



UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

**Managing, organising and sharing fine-grained
data in the spatial design disciplines – An
evaluation of a GeoNode prototype**

by

Cameron Liam Green

Submitted in partial fulfilment for the requirements of the degree

Magister Scientiae (Geoinformatics)

In the Faculty of Natural and Agricultural Sciences

University of Pretoria

Pretoria

13 July 2021

I, Cameron Liam Green declare that the dissertation, which I hereby submit for the degree Magister Scientiae (Geoinformatics) at the University of Pretoria, is my own work and has not previously been submitted by me for a degree at this or any other tertiary institution.

SIGNATURE: .....

DATE: 13 July 2021.....

Abstract

Title: Managing, organising and sharing fine-grained data in the spatial design disciplines – An evaluation of a GeoNode prototype

Student name: Cameron Liam Green, Department of Geography, Geoinformatics and Meteorology, University of Pretoria, South Africa

Supervisor: Dr. Victoria-Justine Rautenbach, Department of Geography, Geoinformatics and Meteorology, University of Pretoria, South Africa

Co-supervisor: Prof. Serena Coetzee, Department of Geography, Geoinformatics and Meteorology, University of Pretoria, South Africa

Degree: MSc Geoinformatics, Faculty of Natural and Agricultural Sciences, Department of Geography, Geoinformatics and Meteorology, University of Pretoria

Geospatial data at a fine-grained scale is commonly collected by researchers and university students in the spatial design disciplines, but is seldom available, accessible and usable for purposes other than the specific study. For example, during an architectural studio, the students store and save their data on their desktop PC or on Google Drive, which they share with other students. However, once the students have completed their design projects, the data is no longer maintained and often deleted. Not only is this way of working with data a challenge for organising, managing and sharing among students, but the data is also lost for any future studies. The question then arises as to how the students and the lecturers can effectively and efficiently organise, manage and share fine-grained data to get the best possible use out of the student collection efforts? First, the functional and non-functional requirements for a solution were specified. Based on a review of literature, there are tools for data collection, software packages for analysing spatial data, and portals for sharing spatial data, but none of these tools are directed at organising and managing spatial data in an educational context. After reviewing possible tools, a prototype for managing, organising and sharing fine-grained data was developed in GeoNode, and its use was evaluated in two architectural studios. While the prototype met all requirements, the students did not necessarily use it as intended and to its fullest potential. Training improved this and should therefore be done before any future deployment of the solution. In further work, the prototype could be improved so as to handle additional data types and to allow lecturers to monitor and guide student activities.

Acknowledgements

I would like to start by thanking my supervisors, *Dr. Victoria Rautenbach* and *Prof. Serena Coetzee* for all the hard work, effort, time, advice, patience and opportunities you have given me over the course of the last two years. Words cannot begin to express how grateful I am for everything you have done for me. Your work ethic and leadership has been inspiring and it was an honour for me to work under you. For that, I am forever grateful.

The financial assistance of the *National Research Foundation (NRF)* and *Department of Science and Technology (DST)* towards this research is hereby acknowledged. Opinions expressed and conclusions arrived at, are those of the author and are not necessarily to be attributed to the *National Research Foundation* and *Department of Science and Technology*.

Additionally, I would like to extend a word of thanks to the *architectural lecturers and students from the University of Pretoria in South Africa and Chalmers University of Technology in Sweden* that provided the feedback that was used throughout this whole process. I would also like to thank the *GeoNode community* as a whole for their guidance and assistance with the development of the GeoNode prototype.

Lastly, I would like to thank my parents, *Steve* and *Debbie*, for all the sacrifices and commitments they made to get me here. Without your support, love and patience throughout my whole academic career I wouldn't have made it this far, everything I do is to make you proud. To my younger brother *Jarryd* who claims he passed his wisdom down to me, thanks. To my *friends* for always pushing me and helping me keep a cool head when times were tough. *Jamie*, thank you for listening to me complain and to *anyone else* that either directly or indirectly helped me with this project, thank you.

*To Irene Grimsell,
my Geography teacher who ignited my passion
and introduced me to the world of geoinformatics.*

Table of Contents

Abstract.....	ii
Acknowledgements.....	iii
Table of Contents.....	v
List of Tables.....	viii
List of Figures.....	ix
Chapter 1 – Introduction	1
1.1. Chapter Overview.....	1
1.2. Background	1
1.3. Problem Statement	3
1.4. Aims and Objectives.....	3
1.5. Research design.....	3
1.5.1. Research Paradigms	3
1.5.2. Research Methodology.....	5
1.5.3. Context.....	7
1.5.4. Ethics.....	8
1.6. Significance of research	9
1.7. Overview of remaining chapters	10
Chapter 2 – Literature Review.....	11
2.1. Chapter Overview.....	11
2.2. Software	11
2.3. Software Engineering	15
2.4. Requirements	16
2.5. Requirements Engineering	18
2.5.1. Requirements Elicitation and Analysis	18
2.5.2. Requirements Specification.....	19
2.5.3. Requirements Validation.....	19
2.6. Web Geographic Information System (GIS) and Cloud-based Mapping.....	21
2.6.1. Overview	21
2.6.2. Thin Client Architecture	24
2.6.3. Thick Client Architecture.....	25
2.6.4. Hybrid Architecture	26
2.6.5. Spatial Cloud Computing Architecture	26
2.7. Geoportals and Spatial Data Infrastructures (SDIs)	27
2.8. Research Data Archives/Libraries	32
2.9. Virtual Research Environments	35
2.10. Learning Management Systems	37
2.11. Content Management Systems.....	39
2.12. F.A.I.R Data Policy.....	41
2.13. Related Studies.....	43

Chapter 3 – The Requirements of the Solution.....	47
3.1. Chapter Overview.....	47
3.2. Context.....	47
3.3. Process to get to the requirements.....	51
3.4. Criteria for Evaluation.....	54
3.5. The Assessment of GeoNode and GeoNetwork.....	58
3.5.1. GeoNetwork and its architecture.....	58
3.5.2. GeoNode and its architecture.....	59
3.5.3. Assessment of GeoNetwork against the requirements.....	60
3.5.4. Assessment of GeoNode against the requirements.....	63
3.6. Summary of Evaluation.....	66
Chapter 4 – Design and Implementation of the Prototype.....	68
4.1. Chapter Overview.....	68
4.2. How GeoNode was installed.....	68
4.3. Prototype – Design and Implementation.....	75
4.3.1. Overview.....	75
4.3.2. User administration in the prototype.....	75
4.3.3. Uploading data in the prototype.....	79
4.3.4. Viewing data in the prototype.....	84
4.3.5. Updating data in the prototype.....	86
4.3.6. Finding data in the prototype.....	88
4.3.7. Making data available in the prototype.....	91
4.3.8. Non-functional Requirements.....	92
4.4. Documentation – Design and Implementation.....	94
Chapter 5 – Evaluation and Discussion of the Prototyped Solution.....	101
5.1. Chapter Overview.....	101
5.2. Test Case One – Hammarkullen, Sweden.....	101
Day 1 – Monday 23 September 2019:.....	103
Day 2 – Tuesday 24 September 2019:.....	103
Day 3 – Wednesday 25 September 2019:.....	104
Day 4 – Thursday 26 September 2019:.....	104
Day 5 – Friday 27 September 2019:.....	105
5.3. Test Case Two – Mamelodi, South Africa.....	105
Day 1 – Monday 27 January 2020:.....	107
Day 2 – Thursday 30 January 2020:.....	107
Day 3 – Friday 07 February 2020:.....	107
5.4. Discussion of Results.....	107
5.4.1. Before starting.....	107
5.4.2. The Test Cases.....	108
5.4.3. The First Test Case.....	109
5.4.4. The Second Test Case.....	110
5.4.5. Lessons Learnt.....	110
5.4.6. Going Forward.....	112
Chapter 6 – Conclusion.....	113
6.1. Chapter Overview.....	113

6.2. Summary of Research Objectives achieved	113
6.2.1. Objective 1 – Literature Study, Understanding Current Practices and Evaluation of Possible Tools:	113
6.2.2. Objective 2 – Develop the context of use and specify the requirements for managing, organizing and sharing of fine-grained data:	114
6.2.3. Objective 3 – Prototype a solution for managing, organizing and sharing of the fine-grained data:.....	115
6.2.4. Objective 4 – Evaluation of the prototype:	115
6.2.5. Objective 5 – Draw conclusions and make recommendations:	115
6.3. Main results obtained from this research	116
6.4. Recommendations for future work	117
References	119
Appendix A	128
Appendix B	129
Appendix C	133

List of Tables

Table 1: Table of licenses for all software used in this research	8
Table 2: Software categories	12
Table 3: WebApp Characteristics	13
Table 4: FAIR data principles according to https://www.go-fair.org/fair-principles/	41
Table 5: Category One from the notes that were drafted in order to understand the process	52
Table 6: Evolution of Category One requirements from the second iteration of functional requirements	53
Table 7: Selected functional requirements from the final iteration of functional requirements	53
Table 8: Functional requirements on which the evaluation was done	55
Table 9: Non-functional requirements on which the evaluation was done	56
Table 10: Overview of the week's activities for Test Case One, 23 - 25 September 2019..	103
Table 11: Overview of the week's activities for Test Case Two, 27 January – 07 February 2020	106

List of Figures

Figure 1: Flow diagram depicting methodology	5
Figure 2: Types of non-functional requirements (Sommerville, 2011).....	17
Figure 3: Thin client architecture (Agrawal and Gupta, 2017)	25
Figure 4: Thick client architecture (Agrawal and Gupta, 2017)	25
Figure 5: Spatial cloud computing architecture adapted by Agrawal and Gupta (2017).....	27
Figure 6: The role of a geoportal in an SDI (Maguire and Longley, 2005)	30
Figure 7: System architecture for an SDI geoportal (Maguire and Longley, 2005).....	31
Figure 8: The University of Pretoria's learning management system, Blackboard (https://clickup.up.ac.za)	38
Figure 9: Data Example: Photograph of the area being mapped	48
Figure 10: Data Example: Sketch of the area being mapped	49
Figure 11: Data Example: Scanned copy of notes from a notebook	49
Figure 12: Data Example: Data being categorised based on years	50
Figure 13: Data Example: Maps rolled up in the back of a cupboard.....	50
Figure 14: Timeline indicating the iteration and evaluation process to get the requirements	51
Figure 15: Simplified architecture of GeoNetwork (Ožana and Horáková, 2008)	59
Figure 16: Architecture of GeoNode (Corti et al., 2019)	60
Figure 17: Home page of the Prototype	74
Figure 18: Registration page.	76
Figure 19: Sign in option on homepage	76
Figure 20: People Users tool (email addresses have been redacted for privacy).....	77
Figure 21: Explore Groups page.	78
Figure 22: Authentication and Authorization Groups tool.	79
Figure 23: Upload Layers page	80
Figure 24: Metadata Wizard for Layers	82
Figure 25: Viewing a layer	85
Figure 26: Editing layer tools	88
Figure 27: Explore Layers page	89
Figure 28: GitHub README.md text file	99
Figure 29: YouTube Playlists for QGIS, Collaborative Data Library, OSM and EpiCollect5..	99
Figure 30: Hammarkullen (Source: OpenStreetMap)	102
Figure 31: Processes followed on Day Two	104
Figure 32: Processes followed on Day Three	104
Figure 33: Processes followed on Day Four	105
Figure 34: Mamelodi East (Source: OpenStreetMap).....	106

Chapter 1 – Introduction

1.1. Chapter Overview

This chapter introduces the background of this project, the problem statement of the research, the aims and objectives that were set out, how the research was designed, the significance of the research, and how the rest of this dissertation will be structured. For this research, time was spent on reading and reviewing relevant literature about content management systems, geoportals, learning management systems, research data archives, virtual research environments, and relevant concepts in software engineering such as functional and non-functional requirements. Time was also spent with the two design studios to become familiar with the processes followed by the architectural design studios at the University of Pretoria and Chalmers University of Technology in their current method of collection, management, and sharing of fine-grained data. Possible tools that might be able to provide an effective solution to this problem will be reviewed and evaluated.

1.2. Background

Urban planning relies on meso-level qualitative and quantitative data that is collected through surveys. Urban planning involves integrating multiple layers of information and data resulting from concurrent and diverse driving forces and stakeholder activities intervening in urban development (Adelfio *et al.*, 2019). However, the absence of fine-grained geospatial data makes such information largely inaccessible for planning and design at neighbourhood or precinct scale, so as to identify areas of development potential or problem hotspots where interventions would be most useful and sustainable. Geospatial data at these meso-level scales are commonly collected by researchers and university students in several disciplines, but are seldom available, accessible, and usable for purposes other than the specific study for which the data has been gathered. The data that is collected for urban planning can also be used in a research environment, but ensuring reproduceable results is of primary concern. The frequency of non-reproduceable research results has raised critical discussions in the science community, led to retractions of published papers, and prompted publishers into setting new policies on the availability of data, methods, and materials presented in research publications (Coetzee *et al.*, 2017). If such data can be collected and stored in a way that will make it trusted, useful and accessible, in a methodology that can be transferred to other educational institutions, it can make a significant contribution to the world of collaborative learning.

Architectural students at the University of Pretoria in South Africa and Chalmers University of Technology in Sweden routinely collect fine-grained visual and qualitative spatial data while working on their design projects, either through transect walks, participatory action research, community mapping, and participatory geographic information systems (PGIS) methodologies. The students from their respective universities either work in groups or by themselves, depending on if they are an undergraduate or a postgraduate, to collect the data using these methods. An architecture design studio is a class in an undergraduate or postgraduate architecture programme in which students receive hands-on instructions in architectural design. Typically, a studio includes distinctive educational techniques such as 'desk crits', referring to when the student's project is critiqued at their desk, and 'juries', which are meetings of students and tutors that openly discuss their projects, and any other potential ideas. This class is usually well equipped with big drafting tables for one or multiple students, pin-up boards for design posters and even a smart board, as architecture is becoming increasingly digitised. These studios collect this geospatial data in order to supplement their design projects and try to create real world scenarios.

This data is usually collected before the design studio begins. The data that the architectural students collect come in many different formats. The architectural students then use this diverse and potentially rich data to enhance their design project by production maps that show a social context, such as safe communal spaces. During the duration of the studio, the architectural students store and save their data either on their desktop PC or on their personal Google Drive which they then share with other architectural students. However, once these architectural students have completed their design project, the data stays on their personal Google Drive, and when they leave university, the data essentially leaves with them. Some students even delete the data once the project is finished, as they claim that they will no longer need it. This can prove to be a lost opportunity if the area is being studied over several years, for example, in a longitudinal study. Over time, this data has the potential to reveal changing spatial patterns, such as a change in land use from informal trading to a taxi rank, or even a change in perception of a place such as towards an open field, which can feel safe at first as it is used by the community as sports field, but then over time can be neglected and come to feel as an unsafe area. The spatial data deals with the questions 'what is happening', 'where' and 'how' and the surveys answers the questions why the growth patterns are evolving the way that they are (Abbott and Douglas, 2003). Steiniger *et al.* (2017) suggested the use of web-based data service that is best thought of as a web-based 'drop box' service for spatial data and documents to solve the challenge of losing data.

Based on the above, the question then arises as to how the students and the lecturers can effectively and efficiently organise, manage, and share such spatial data to get the most value out of the student's collection efforts. Currently, there are multiple tools for data collection, such as KoBoToolbox and FieldPapers, multiple software packages such as QGIS and ArcGIS for analysing spatial data, and portals for sharing spatial data, but none of these tools are directed at organising and managing spatial data in an educational context.

1.3. Problem Statement

University educators, researchers and students in spatial design disciplines such as those in architecture need a solution for managing, organising and sharing fine-grained data collected in local environments during the design studios.

1.4. Aims and Objectives

The aim of this research was to evaluate a solution for university educators and researchers in spatial design disciplines such as those in architecture for managing, organising and sharing fine-grained data collected in local environments during the design studios at the University of Pretoria in South Africa and Chalmers University of Technology in Sweden.

In order to achieve the aim of this research, the following objectives were met:

1. Read and review relevant literature to understand the problem and review possible tools for implementing a solution.
2. Develop the context of use and specify the requirements for a solution for architectural studios at the University of Pretoria and Chalmers University of Technology.
3. Prototype a solution for managing, organising and sharing of the fine-grained data.
4. Evaluate whether the prototype meets the specified requirements.
5. Based on the findings of the evaluation, draw conclusions on possible solutions for managing, organising and sharing fine-grained data and make recommendations.

1.5. Research design

1.5.1. Research Paradigms

Two paradigms characterise the research in the information systems discipline: behavioural science, and design science. The behavioural science paradigm seeks to develop and verify theories that explain or predict human or organisational behaviour. The design science

paradigm seeks to extend the boundaries of human and organisational capabilities by creating new and innovative artifacts (Hevner *et al.*, 2004). Design sciences focuses on the creation of an innovative, purposeful artifact for a special problem domain. The developed artifact is evaluated in order to ensure its utility for the specified problem meaning the problem that the artifact must either solve a problem that has not yet been solved, or provide a more effective solution.

This research falls under the design science paradigm. The design process is a sequence of expert activities that produces an innovative product. This is a build-and-evaluate loop, which is typically iterated a number of times before the final design artifact is generated. This research serves to address an important unsolved problem in a unique and innovative way.

The primary method of this research is prototyping, and the secondary method is a literature survey. Before the prototyping is designed and completed, the literature survey will be completed to gain an understanding of current practices that assisted with knowledge in the fields of this research. A prototype of the solution was designed and implemented as a proof of concept.

1.5.2. Research Methodology

Figure 1 provides an overview of the research methodology that this research followed.

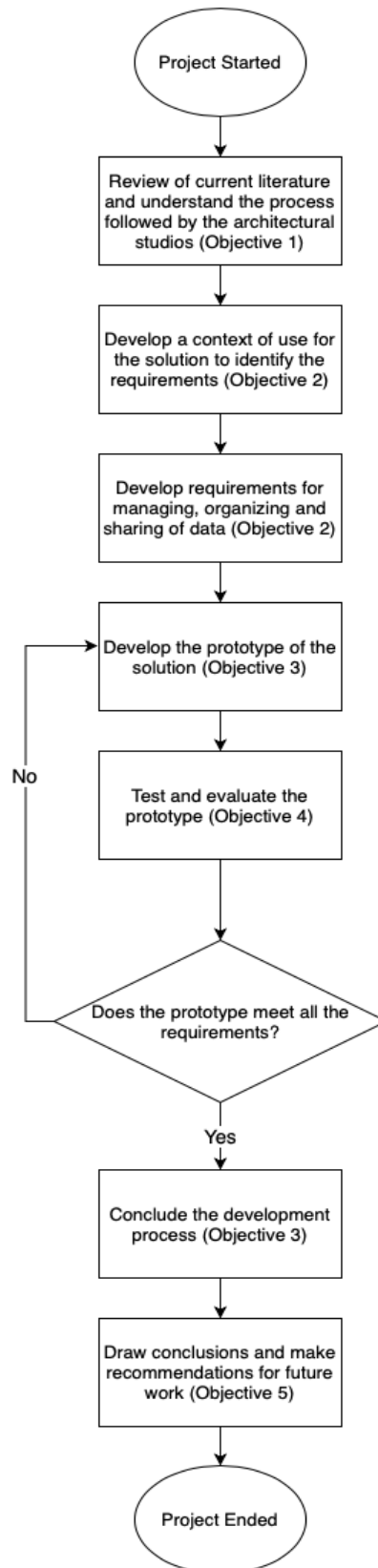


Figure 1: Flow diagram depicting methodology

Relevant literature was read and reviewed (Objective 1) in order to gain an understanding that can aid this research about content management systems, geoportals, learning management systems, research data archives, and virtual research environments. Relevant concepts in software engineering, such as functional and non-functional requirements, were also read up on. Time was spent with the two design studios to become familiar with the processes followed by the architectural design studios at the University of Pretoria and Chalmers University of Technology in their current method of collection, management and sharing of fine-grained data. A lack of understanding for the processes that the architects follow with regards to their data collection; data management; and data storage can slow down the research process and delay the requirements writing process. From here, the requirements for managing, organising and sharing of fine-grained data in architectural studios at the University of Pretoria and Chalmers University of Technology were specified, which met Objective 2. See 1.5.3. Context which outlines the test cases. This was done by studying the process of data collection and storage that the architects followed. The scope of the requirements for this project are limited since the requirements are based only on two architectural design studios, one from the University of Pretoria and the other from Chalmers University of Technology, but they can be used by others and further expanded in future work. It must be noted that some of the requirements are mandatory requirements meaning they have to be met and other requirements are nice-to-haves meaning they are not a requirement that has to be met but it would be advantageous if it is.

A prototype of the solution for managing, organising and sharing of the fine-grained data was developed. This comprised a technical solution as well as any associated documentation that explained to the users of the solution the correct steps for certain procedures. The documentation consists of mixed media methods such as a Wikipedia page and YouTube videos. Doing this met Objective 3. As an already existing solution was used as the prototype, there was already existing documentation available on the configuration and use of the prototype, but this documentation may be outdated, limited, or incomplete.

The technical solution and the associated documentation were demonstrated to the architectural lecturers from both universities to gain feedback and evaluate the technical solution and documentation. Based on the feedback provided by the architectural lecturers, the technical solution and its documentation was improved to allow the users to be better equipped to use the solution. As this project constitutes design science. The build-and-evaluate loop will be iterated a number of times.

Based on all the subsequent findings and evaluations, conclusions were drawn on the overall effectiveness of the methods for managing, organising and sharing fine-grained data and recommendations we made based on the findings which are presented in this dissertation. This satisfies the last and final objective, Objective 5.

1.5.3. Context

This research was part of a larger project between the University of Pretoria and Chalmers University of Technology, funded by the National Research Foundation (NRF) and the Swedish Foundation for International Cooperation in Research and Higher Education (STINT) (Grant Number 112634). The primary investigator for this project is Professor Chrisna Du Plessis, Head of the Department of Architecture at the University of Pretoria. Two design studios, one from the University of Pretoria, and the second design studio is from Chalmers University of Technology, will be participating in this study.

As part of the projects, two test cases were identified. Test Case 1 took place from the 23rd of September 2019 to the 27th of September 2019 in Hammarkullen, Sweden and Test Case 2 took place from the 27th of January 2020 to the 7th of February 2020.

Test Case 1 (23-27 September 2019): The technical solution and its associated documentation was tested in a master's architectural design studio at Chalmers University of Technology in Sweden as part of the NRF/STINT Joint Science and Technology Collaboration. The solution was tested for its ability to store and manage geospatial and non-geospatial data. The solution was demonstrated to the students in the design studio by the researchers with backgrounds in geoinformatics and computer science, after which the students used the solution and documentation to test the usefulness of the solution and documentation. Feedback and suggestions for improvement from the students were noted and taken forward. Based on this feedback, the technical solution and its documentation was improved for Test Case 2 in early 2020.

Test Case 2 (27 January 2020 – 07 February 2020): The solution and its associated documentation was tested in a South African Honours architectural design studio at the University of Pretoria as part of the NRF/STINT Joint Science and Technology Collaboration. During this session of the testing, a more conservative approach was taken. For this design studio, the architectural students who were present for the training in Sweden demonstrated the technical solution and its documentation. The architects need to be able to use the solution without any input from the developers. Feedback and suggestions for improvement from the students were noted and taken forward. Based on this feedback, the solution and its

documentation were improved for overall use. Test Cases 1 and 2 both helped to meet Objective 4.

1.5.4. Ethics

For this project, ethical clearance was granted by the University of Pretoria's Faculty of Natural and Agricultural Sciences Ethics Committee, reference number NAS128/2019. This project involved no direct contact or interviewing of any of the students that used the prototype by the researchers from the University of Pretoria. The ethics clearance was for the development of a platform, where data can be shared with communities and other stakeholders. Architectural students have been using this platform, but they were not interviewed directly or surveyed for feedback. When the researchers were in Sweden, they were invited to a feedback session moderated by Monica Billger, one of the local lecturers from Chalmers University of Technology. All the feedback that was obtained during the session was provided to the researchers by Monica Billger. To populate the platform, a sample of OpenStreetMap data was loaded. OpenStreetMap data is licensed under the Open Database License. The ethics approval letter for this project can be found in Appendix A.

Table 1: Table of licenses for all software used in this research

Data or Software	License	Purpose of Data or Software
OpenStreetMap	Open Database License (http://opendatacommons.org/licenses/odbl/1.0). Any rights in individual contents of the database are licensed under the Database Contents License.	Data collection
KoBoToolbox	Licensed for use under the GNU Affero General Public License v3.0 (https://github.com/kobotoolbox/kpi/blob/master/LICENSE) (https://opensource.org/licenses/AGPL-3.0).	Data collection
EpiCollect5	Licensed for use under the GNU Affero General Public License v3.0 (https://github.com/ImperialCollegeLondon/EpiCollectplus) (https://opensource.org/licenses/AGPL-3.0).	Data collection
FieldPapers	Licensed for use under the GNU General Public License v2.0 (https://github.com/stamen/fieldpapers/blob/master/LICENSE).	Data collection
JOSM	JOSM, and all its integral parts, are released under the GNU General Public License v2 or later (https://josm.openstreetmap.de/browser/trunk/LICENSE).	Data editing
Maptionnaire	General Terms and Conditions: https://maptionnaire.com/terms . Chalmers University of Technology has a license to use Maptionnaire.	Data collection
QGIS	Licensed for use under the GNU Affero General Public License v3.0 (https://www.gnu.org/licenses/gpl-3.0.en.html).	Data editing and visualization
GeoNode	Licensed for use under the GNU Affero General Public License v3.0	Data storage

Data or Software	License	Purpose of Data or Software
	https://github.com/GeoNode/geonode/blob/master/license.txt https://opensource.org/licenses/AGPL-3.0 .	
GeoServer	Licensed for use under the GNU General Public License v2.0 https://github.com/geoserver/geoserver/blob/master/LICENSE.txt .	Data storage
PostgreSQL	PostgreSQL is released under the PostgreSQL License https://www.postgresql.org/about/licence/ https://opensource.org/licenses/postgresql .	Data storage
PostGIS Extension	PostGIS is released under the GNU General Public License v2.0 or later (https://opensource.org/licenses/gpl-2.0.php).	Spatial extension library for PostgreSQL – data storage
Git	Git is released under the GNU General Public License version 2.0 (https://opensource.org/licenses/gpl-2.0.php).	Version control
SQLite3	SQLite is in the public domain and does not require a license.	Data storage
GDAL	GDAL is licensed under an MIT/X style license https://raw.githubusercontent.com/OSGeo/gdal/master/gdal/LICENSE.TXT .	Reading and writing of raster and vector data – data editing
OpenJDK	Licensed for use under the GNU General Public License v2.0 (https://openjdk.java.net/legal/gplv2+ce.html).	Enables Java to run on the virtual machine
Apache Tomcat	All software produced by The Apache Software Foundation or any of its projects or subjects is licensed according to the terms of Apache License, Version 2.0 (http://www.apache.org/licenses/LICENSE-2.0).	Application server designed to execute Java servlets and render web pages
NGINX	Nginx is free and open-source software, released under the terms of the 2-clause BSD license https://opensource.org/licenses/BSD-2-Clause .	A web server designed for web serving, reverse proxying, caching, load balancing

1.6. Significance of research

A significant output from this research was that a solution was evaluated for a problem from a set of high-level functional requirements, which detailed the needs of the architects in their design studios. These high-level functional requirements can then be later used to develop a custom solution to suit the needs of these design studios. These requirements, although unique to the University of Pretoria and Chalmers University of Technology, can be applied to other architectural design studios around the world. Other disciplines such as environmental science, town planning, zoology, and botany can utilise this solution within their programmes to manage, organise and share any data they have collected. This problem was tackled with an international multi-disciplinary research team, with differing backgrounds in architecture, computer science, and geoinformatics, which allowed for different viewpoints and approaches to this complex problem. At the end of this project, the architectural design studios had a working prototype that allowed them to manage, organise and share any of the fine-grained data that had been collected in an educational context.

1.7. Overview of remaining chapters

The remaining chapters of this dissertation are structured as follows:

Chapter 2 – Literature Study. This chapter will highlight relevant practices from recent literature about architectural design studios to understand the problem, cover possible solutions such as content management systems, geoportals, learning management systems, research data archives, and virtual research environments. Relevant concepts in software engineering, such as functional and non-functional requirements are explained. This chapter covers Objective 1.

Chapter 3 – The Requirements of the Solution. This chapter specifies the requirements of the architects and the context in which the architects collect, manage and share their fine-grained data. Two possible tools, namely GeoNode and GeoNetwork, were evaluated to determine which is better for implementing the solution. This chapter covers Objective 2.

Chapter 4 – Design and Implementation of the Prototype. This chapter focuses on the design and implementation of the GeoNode prototype. It describes the design of the prototype by focusing on the functional and non-functional requirements, describes how the GeoNode prototype was developed to meet all the requirements, and includes the documentation that was developed as part of the prototype. This chapter covers Objective 3.

Chapter 5 – Test Cases of the Prototyped Solution. The test cases where the GeoNode prototype and its associated documentation was used are explained in detail and describe whether they meet the specified requirements. The results of the evaluations obtained through these test cases are discussed with reference to the literature in section 5.4.3. This chapter covers Objective 4.

Chapter 6 – Summary and Conclusions. This chapter provides a summary of how each objective was met, what was the most significant finding from each objective and draw conclusions on the effectiveness of the methods for managing, organising and sharing fine-grained data. It also highlights the main results obtained in this research, as well as makes recommendations for future work. This chapter satisfies the last objective, Objective 5.

Chapter 2 – Literature Review

2.1. Chapter Overview

The aim of this literature review is to examine and understand previous research related to the key concepts of this research. The literature review helped to determine and redefine the existing research aims, objectives and methodology. The literature review covers topics such as software, software engineering, requirements, requirements engineering, portals and geoportals, Web GIS or Cloud-based mapping, virtual research environments (VREs) or science gateways, research data archives/libraries, learning management systems (LMEs), content management systems (CMSs) and the F.A.I.R data policy, as well as related work. All these topics play a vital role in this research, and contribute to it in ways that will be noted.

2.2. Software

It is crucial to build software that is ready to meet the requirements of the 21st century. Software and applications have become deeply embedded in our everyday lives, where the need for high-functioning software and applications that meet our requirements remains increasingly important. The requirements that are needed by individuals, businesses and governments are getting ever more complex with each passing year. Individuals, businesses and governments increasingly rely on software for strategic and high-level decision making, as well as day-to-day running. When software fails individuals, businesses, or governments, this leads to anything from minor inconvenience to a catastrophic system failure, where they can lose all their data and business intelligence. While designing and building software, there will always be hurdles that need to be overcome, so it is important to remember Mosher's Law of Software Engineering, which states "Don't worry if it doesn't work right. If everything did, you'd be out of a job."

A textbook definition by Pressman (2005) of software is that software can be broken down into three parts: (1) instructions (computer programmes) that when they are executed, they provide the desired features, functions and performances that the user requested; (2) data structures that enable the programme to sufficiently manipulate information; and (3) descriptive information in both hard copy and virtual forms that describe the operation and use of the programmes. Hence, it can be said that software is a computer programme and its associated documentation that is developed for a particular customer or the general market. It is a set of instructions, data or programmes that are used to run computers and complete specific tasks. Software, as opposed to the hardware, which refers to the physical components of a computer,

is the generic term used to describe the applications, scripts, and programmes that run on a computer.

Software can be grouped into seven broad categories of computer software (Pressman, 2005), namely: system software, application software, engineering/scientific software, embedded software, product-line software, web applications, and artificial intelligence (AI) software. Table 2 expands on each software category further.

Table 2: Software categories according to Pressman (2005)

Software Category	Definition
System	A collection of programmes that are written to facilitate or support other programmes. Systems software are defined by the heavy interaction they have with the computer's hardware, the heavy usage by the computer's multiple users, resource sharing, and the multiple external interfaces of the computer.
Application	A stand-alone programme that solves specific business needs. Software applications process business, or technical data in a way that supports business operations or management decision making. It is used to control business functions and operations in real time.
Engineering/Scientific	"Number crunching" algorithms that range from use in astronomy, vehicle stress analysis, and automated manufacturing. With the improvement of software, engineering/scientific software is moving away from the old number crunching to more modern computer-aided design, system simulation, real-time analysis, and any Geographical Information Systems (GIS).
Embedded	Exists within a system and is used to implement and control features and functions for the end user and for the system itself. Embedded software can either provide basic functions such as keypad control or significant functions, such as dashboard displays.
Product-line	Designed to provide a specific capability for use by different customers. It can focus on a limited market such as inventory control or a mass consumer market such as database management.
Web Applications	Commonly referred to as "WebApps", this network-centric software category spans a wide range of applications. In the simplest of forms, WebApps can be a set of simple linked hypertext files that present data using text or limited graphics but with the evolution of Web 2.0, WebApps are emerging into sophisticated computing environments that can provide stand-alone features as well as the capability to be integrated with corporate databases and day to day business applications.

Software Category	Definition
Artificial Intelligence (AI)	As opposed to engineering/scientific software, AI software makes use of non-numerical algorithms to solve complex problems that are not solvable by normal number crunching. Applications within this category can include, but not limited to: robotics, pattern recognition, artificial neural networks, and even game playing.

For the purpose of this literature review only WebApps will be discussed to provide context and background on WebApps as the nature can define the requirements. Back in 1998, Powell *et al.*, suggested that Web-based systems and applications involve a mixture of print publishing, software development, marketing, computing, internal communications and commands, external relations, art, and technology. WebApps can differ significantly from each other, but they can share similar characteristics such as network intensiveness, concurrency, unpredictable load, performance, availability, data driven, content sensitive, continuous evolution, immediacy, security and aesthetics. Table 3 expands on each of the characteristics and explains each one in detail.

Table 3: WebApp Characteristics (Pressman, 2005)

WebApp Characteristic	Definition
Network Intensiveness	As WebApps reside on a network and must serve the needs of a wide community of clients, the network must enable either world-wide access and communication or limited access and communication.
Concurrency	A large number of users may request access to the selected WebApp at any given time, which means that the steps taken to complete a process may be complete or partial and this cannot affect the final outcome of the product.
Unpredictable Load	The number of users that request access to the WebApp may vary from time to time or day-to-day.
Performance	If the users of the particular WebApp have to wait too long for a service or a product that the WebApp provides, they may decide to no longer use the WebApp and go somewhere else.
Availability	Users of popular WebApps expect 100% availability on the basis of 24/7/365, downtime for maintenance must be kept to a minimum.
Data Driven	The main function of many WebApps is to use software to display text and graphic content to the user. WebApps are also commonly used to access data that exists in databases.
Content Sensitive	The quality and nature of the content of a WebApp is an important factor in

WebApp Characteristic	Definition
	the quality of the particular WebApp.
Continuous Evolution	WebApps evolve continuously over time unlike normal applications, which consist of a series of planned and chronologically spaced releases.
Immediacy	Immediacy in terms of software is required to get software to the open market quickly. WebApps often have a time-to-market that is relatively short, a matter of a few days or weeks.
Security	As WebApps are accessible via open networks, which make it difficult to limit the number of end users who may access the application. In order to protect any sensitive data, strong security measures need to be implemented throughout the WebApp.
Aesthetics	An undeniable part of the appeal of WebApps is the look and feel of the application. This can essentially make or break a WebApp before it is fully utilised.

With these WebApp characteristics in mind, the requirements development process can begin. It is important to understand WebApps so that the correct requirements can be drawn up, which allows the application to be of best quality, and ultimately serve its purpose at the end of the day. Having applications that are not well designed and built impractically, can waste a significant amount of capital over the long-term, if the application will need to be redesigned and rebuilt.

As with any project, funding can potentially be a problem. The lack of funding can hinder the process of acquiring any software package and the maintenance of any software that has been acquired. There are many ways to acquire various software packages, such as shareware, liteware, freeware, public domain software, and open source (Guadamuz, 2009). Shareware is a software package that has all the capabilities of the software, which is distributed on a free or trial basis, with the intention of a license being bought at the end of the trial period, whereas liteware is a type of shareware with some of the capabilities disabled until the full version is bought. Freeware can be downloaded from the internet for free, but with copyright restrictions. Public domain software can also be downloaded from the internet for free and without any copyright restrictions. Open source software refers to software for which the source code is publicly available, and the copyright holder grants users the rights to study, change, and distribute the software to anyone for any purpose.

2.3. Software Engineering

Sommerville (2011) defined software engineering as an engineering discipline concerned with all the aspects of software production from the early stages of system specification through to maintaining the system after it has gone into use or production. Software engineering includes the process, the collection of methods, and an array of tools that allow professionals to build high-quality computer software (Pressman, 2005). Both of these definitions are subjective in nature therefore the Institute of Electrical and Electronics Engineers (IEEE) developed a more accepted and comprehensive definition, where software engineering refers to the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software (Pressman, 2005).

The systematic approach that is used in software engineering is sometimes called a software process. A software process is a sequence of activities, actions, and tasks that leads to the production of a software product. In the context of software engineering, the software process is not a rigid process for how to build computer software, but rather, it is an adaptable approach that enables software developers to pick and choose the appropriate set of work actions and tasks. A generic process framework for software engineering includes five activities, namely: communication, planning, modelling, construction, and deployment (Pressman, 2005).

Communication: It is vitally important for software engineers to communicate and coordinate with the customer. The purpose of this is to understand the customer's objectives and outcomes for the project and to gather the requirements that will help define the software features and functions.

Planning: Any complicated project requires planning, a guide on what steps to follow. In software engineering this guide is called a software project plan, which defines the software engineering work by describing the technical tasks that need to be completed, the risks that are likely to be encountered, the resources that will be needed to complete the project, and the work schedule that will be followed.

Modelling: A model is a "sketch" of something that allows someone to see the bigger picture of what they are building. A software engineer also creates a sketch by creating models to better understand software requirements and the design process that will be used to achieve the requirements.

Construction: This activity combines code generation and the testing phase that is required to discover any errors or bugs in the code.

Deployment: The software is delivered to the customer who then evaluates the delivered software product and then provides feedback based on their evaluation and the process repeats again.

2.4. Requirements

The requirements for a system are the description of what the system ought to be able to do, the services that the system provides, as well as the constraints on its operation. The requirements of the system mirror the needs of the customers for a particular system that serves a certain purpose, such as controlling a device, placing an order, or simply finding some information. The process of determining, analysing, documenting, and checking these constraints and services is called requirements engineering (RE), which will be discussed later in this chapter. If requirements are incorrectly drawn up at the beginning of the project, this can lead to problems during the requirements engineering phase. Sommerville (2011) made a clear separation between the different levels of description. He differentiates between them by using the terms “user requirements” and “system requirements”, respectively. He defined these as follows:

1. *User requirements* are simply statements in a natural language with diagrammes detailing the services the system is expected to provide to users, as well as the constraints under which it must operate.
2. *System requirements* are the more detailed descriptions of the system’s functions, services and operational constraints. The system requirements document should define exactly what is going to be implemented. It can be part of the agreement between the system user/buyer and the software developers.

A more traditional view on software system requirements often classify the requirements into functional and non-functional requirements.

Functional requirements describe how the system ought to react to particular inputs and how the system should behave in particular situations, that is, the functional requirements describe what the system should do. In some cases, the functional requirements may also state what the system should not do. These kinds of requirements depend on the type of software that is being developed, the expected end users of the software, and the general process followed by the organisation determining the requirements. Functional requirements are usually described in an abstract way that can also be understood by the system users. They vary from general requirements covering what the system ought to do in general, to very specific requirements. The functional requirements specification of a system ought to be complete and

consistent. Completeness means that all services required by the user ought to be defined, and consistency means that the requirements should not have contradictory definitions.

Non-functional requirements refer to the constraints on the services or functions that are offered by the system. They include and are not limited to timing constraints, constraints on the development process, constraints imposed by standards and ethics, as well as constraints imposed by the client. Non-functional requirements often apply to the system as a whole, rather than to the individual system features or services. Non-functional requirements are not directly concerned with the specific system services delivered by the system to the end user. Non-functional requirements such as performance, scalability, or security usually specify characteristics of the system as a whole. They are often more critical or important than individual functional requirements. Not correctly meeting the non-functional requirements can lead to the whole system being unusable. Non-functional requirements also have the capabilities of affecting the overall architecture of the system rather than the individual components. Figure 2 shows a classification of non-functional requirements (Sommerville, 2011). It can be seen from Figure 2 that the non-functional requirements may come from either the required characteristics of the software (product requirements), the organisation developing the software (organisational requirements), or from external sources.

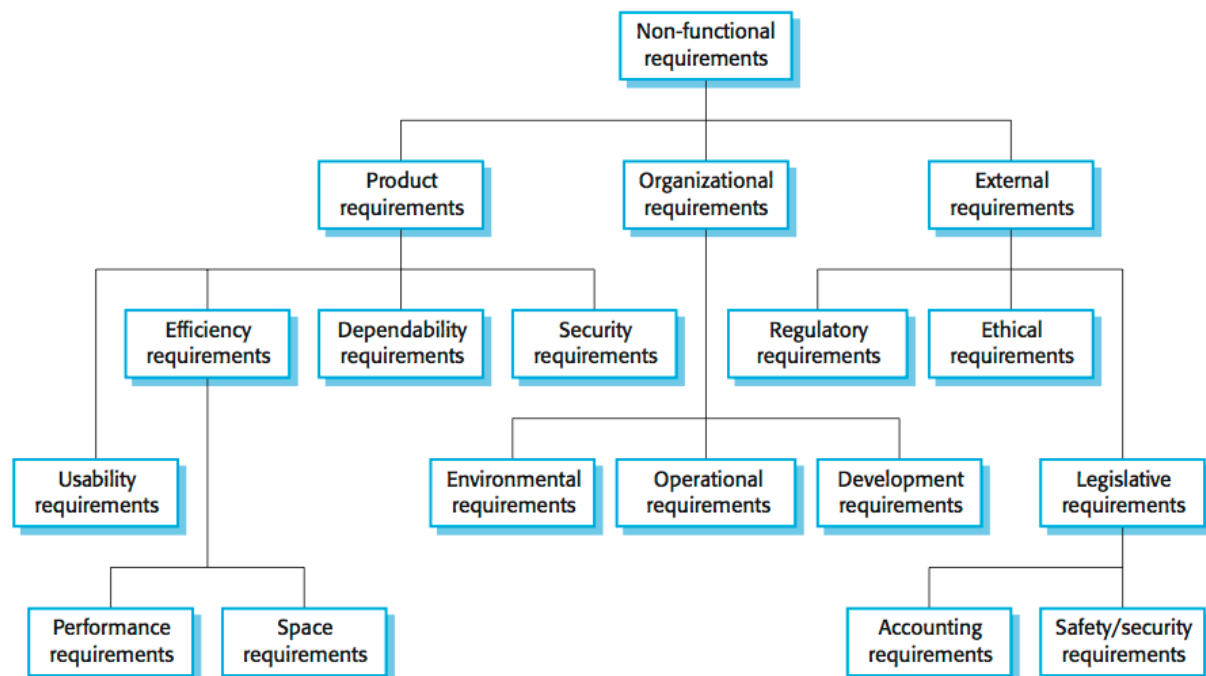


Figure 2: Types of non-functional requirements (Sommerville, 2011)

2.5. Requirements Engineering

Requirements engineering is understood as the broad range of tasks and techniques that lead to the understanding of the requirements. From a software process point of view, requirements engineering is a major software engineering action that begins during the communication phase, and continues into the modelling phase. It must be adapted to the needs of the process, the project, the product, and the people doing the work. Requirements engineering builds a bridge between design and construction. It also provides the appropriate tool for understanding what the customer wants, analysing customer needs, assessing the feasibility of the system, negotiating a reasonable solution to the problem, specifying the solution to the problem clearly, validating the specifications of the system, and managing the requirements as they are transformed into an operational system.

Requirements engineering process includes four high-level activities (Sommerville, 2011). These four activities focus on either assessing whether the system is useful to the business (feasibility study); discovering the requirements (elicitation and analysis); converting these requirements into a standard form (specification); or finally; checking that the requirements actually define the system that the customer wants (validation). Requirements engineering is an iterative process, in which all the activities concerned with the requirements engineering process are incorporated.

2.5.1. Requirements Elicitation and Analysis

Software engineers work with customers and system end-users to determine the application domain, what services the system should provide, the required performance of the system, and hardware constraints. Requirements elicitation and analysis involves a variety of different people in any organisation, and is concerned with the origins of software requirements and how they are collected. It is the first step in the understanding of the problem the software needs to solve. A system stakeholder is anyone who has some form of either direct or indirect influence on the system requirements. A fundamental principle of good requirements elicitation is the process of effective communication between the various stakeholders. Requirements elicitation is sometimes called requirements discovery. Requirements discovery refers to the process of interacting with the stakeholders of the system to discover their unique requirements by gathering information about the required system and existing systems and then determining the user and system requirements from this information. Sources of information during this discovery phase include documentation, system stakeholders, and specifications of similar systems.

2.5.2. Requirements Specification

Requirements specification refers to the process of writing down the user and system requirements in the requirements document. The user and system requirements ought to be clear, unambiguous, easy to understand, complete, and consistent. This can be difficult to achieve, as the stakeholders of the system tend to interpret the requirements in different ways, and there are often conflicts and inconsistencies in the requirements. In software engineering, software requirements specification typically refers to the production of a document that can be reviewed, evaluated, and approved. The user requirements for a system should describe the functional and non-functional requirements, so that they are understandable by system users who do not have a detailed and technical background. System requirements are expanded versions of the user requirements that are used by software engineers as the starting point for the designing of the system. They add detail and explain how the user requirements should be met by the system. They should simply describe the external behaviour of the system and its operational constraints.

2.5.3. Requirements Validation

Requirements validation refers to the process of checking that the requirements actually define the system that the customer actually wants. Requirements validation overlaps with analysis, and is focused on finding the problems with the requirement, which is important because errors in a requirements document can lead to widespread rework when these problems are discovered during development or after the system is in service. The cost of fixing a requirements problem by making a systems change is usually much higher than that of repairing a design or code error. The main reason for this is because a change to the requirements usually means that the system design and how the system is implemented needs to be changed which also leads to the system needing to be re-tested which costs time and money.

During the requirements validation, there are five different types of checks that can be carried out to determine the validity of the requirements:

1. *Validity checks*: a user may think that the system is needed to perform certain functions but after careful consideration, thought and analysis, it may be discovered that additional or different functions are needed.
2. *Consistency checks*: requirements in the document should not conflict with each other. There should not be any contradictory constraints or different descriptions for the same functions.
3. *Completeness checks*: the requirements document should include the requirements that define all the functions of the system and the restrictions anticipated by the system user.

4. *Realism checks*: using prior knowledge of any existing systems and technologies, the requirements should be checked to ensure that they can actually be implemented. This ought to consider the budget of the project, and the schedule for the development of the system.
5. *Verifiability*: to reduce the potential for dispute between the customers and the software developer, system requirements should always be written in such a way as to be verifiable. This means that you should be able to write a set of tests that can demonstrate that the delivered system meets all the requirements set out.

Once the validity of the requirements has been confirmed, there are three main requirements validation techniques that can be used to test the requirements of the system so as to ensure that all the requirements are met by the software development team for the end-client.

1. *Requirement reviews*: the requirements are analysed systematically by a team of reviewers who check for errors and any inconsistencies.
2. *Prototyping*: a working version of the system in question is demonstrated to the end-users and clients for constructive feedback.
3. *Test-case generation*: requirements should always be testable. The tests should be developed as part of the validation process as this will lead to any problems with the requirements. If a test is difficult to design, it usually means that the requirement will be difficult to implement and should be reconsidered.

2.6. Web Geographic Information System (GIS) and Cloud-based Mapping

2.6.1. Overview

Maps have the ability to offer an overview and insight into spatial patterns and relations, they can guide users from A to B, show the arrangement of the landscape, display changes in population distributions, or even show future urban plans. Maps are able to do this because they represent selections of reality. If the maps are well designed, the meaning of their symbology will give the user a link to a part of reality (Kraak, 2004). Traditionally, geographic information systems (GIS) is a system of hardware, software, and procedures that capture, store, edit, manage, analyse, share, and display georeferenced data (Bolstad, 2005). A GIS is used for storing, collecting, retrieving, transforming, and displaying spatial data that is linked to a geographical location. GIS offers the facilities for data management, data manipulation, data capture, analysis, and presentation. GIS is a combination of cartography, statistical analysis, software, hardware, computer science, geography, photogrammetry, remote sensing, surveying, GPS technology, and data, and is usually used as a supporting system for decision making, by offering the best possible decisions through spatial and non-spatial data relations, processing and visualisation, as well as other disciplines concerned with handling and analysing spatially referenced data. Traditional GIS can usually only serve a dedicated user with sophisticated software and hardware with limited access by the public. (Karnatak *et al.*, 2007; Bhat, Shah and Ahmad, 2011; Aly and Makram, 2013; Helmi, Farhan and Nasr, 2018). Maps play an important role in GIS. Not only do they present the final results of a spatial analysis, but they are also critical during the process of geospatial data handling. Today, millions of maps are being made and used by an audience that earlier would not necessarily have thought of creating maps by themselves (Kraak, 2004) but it is a question as to how has this been achieved, as unfortunately everyone does not have equal access to GIS, nor is everyone able to spend the time to use it effectively (Alesheikh, Helali and Behroz, 2002), as a traditional GIS is an expert tool and expensive (Yang *et al.*, 2016).

Web applications can be used to perform major tasks on the web. These applications run on the web server and the results depend on the user's input that is given by the web browser. With the advancement of web technologies, specifically Web 2.0, some of the functionalities of a GIS has moved to online platforms or distributed systems. This is called Web GIS, meaning a GIS that uses some aspect of web technologies (Fu and Sun, 2010). Web GIS is an extension and application of client/server computing, where the geospatial data is accessible in a shareable environment (Karnatak *et al.*, 2007). Web GIS has evolved with the growth of the web and the fast development of the web has made data of GIS enabled environments accessible to the public through the Web GIS (Agrawal and Gupta, 2017). The

integration of GIS with the web has resulted in making GIS available to the public in an easy and efficient way and the increasing popularity as well as the decreasing cost of laptops, smart phones and other mobile devices created a major role of the internet we rely on in our day-to-day activities. This has then resulted in the wide spread use of web GIS (Agrawal and Gupta, 2017). For example, a web GIS can be used to display results of a spatial analyses to the end user or an external processing service could be used to perform an analysis. Similarly, cloud based mapping or GIS have emerged by making use of the flexibility of the cloud environment for data capturing, visualisation, analysis, and sharing (Helmi, Farhan and Nasr, 2018). Web GIS or cloud-based mapping can provide users with a wide range of online tools or services at lower costs than traditional desktop GIS solutions.

Cloud computing is evolving as a key computing platform for sharing resources that include infrastructures, software, applications and business processes (Bhat, Shah and Ahmad, 2011). Bhat, Shah and Ahmad (2011) and Agrawal and Gupta (2017) used the National Institute of Standards and Technology (NIST) to define Cloud Computing (CC) as follows:

Cloud Computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g. networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

Cloud computing can be seen as a new style of computing, in which dynamically scalable and often virtualised resources are provided as services over the internet with many experts expecting that Cloud Computing will reshape information technology (IT) processes and the IT marketplace (Aly and Makram, 2013). Cloud Computing is interpreted as having five key characteristics (i.e. on-demand self-service, ubiquitous network access, location-independent resource pooling, rapid elasticity and pay-per-use). Three delivery models (i.e. Software as a service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS)) and four deployment models (i.e. private, community, public, and hybrid) (Bhat, Shah and Ahmad, 2011). As a result of integrating GIS with the concept Cloud Computing (CC), Cloud GIS was proposed as a valuable approach to give a broad range of services to users across the world.

Currently, there are various web GIS or cloud-based mapping solutions, such as Esri Online (<https://www.arcgis.com/index.html>), Carto (<https://carto.com>), Mango map (<https://mangomap.com>) or Mapable (<http://www.mapable.co.za/>), that allow non-experts to create customisable web maps using their own data. These tools are generally easy to use and provides a wide range of add-ons, such as various styles or base layers. When a user

creates a map using these tools, the data needs to be loaded into the database of the service provider and the copyright of the data is thereafter no longer clear (e.g. does the data this exclusively belong to the user or can the service provider distribute or use this data?). Although the user can add predefined additional layers (i.e. supplied by the service provider) to their map, the user would not be able to see all the metadata of the geospatial data that is currently loaded on the entire online tools. For example, to determine whether other user maybe has data on a specific topic or region. For educational. projects, the lecturer is not able to track student contributions and activity using the existing functionalities of these tools. An example of a web GIS that focuses on statistics, both basic and advanced statistics, is the multi-layer GISualization [sic] framework that was developed by Adelfio *et al.* (2019). This framework is built on five layers of data, representing different methods for data collection as well as different grades of complexity, richness and interpretation.

Cloud GIS is a next generation Web GIS solution with advanced capabilities of collecting, processing, analysing, and publishing geospatial and database data. The main goal is to provide the power of desktop GIS solutions on a web-based platform and bring GIS closer to a wide audience. GIS and Cloud Computing will provide full featured GIS capabilities over the web in any browser with powerful data processing capabilities. The web browsers provide tools working with maps, including spatial analysis. As Cloud Computing is emerging as a new technology, it is introducing new opportunities for the world of GIS, as clients or users do not need to install updates that are delivered over the network as Cloud GIS users share a single geographical data storage. Cloud GIS also provides lower training costs for non-expert GIS users (Aly and Makram, 2013).

The use of service-oriented architecture in GIS is becoming increasingly widespread. This approach helps to hide the technical details of the datasets in question by exposing them through standards, implementation-neutral web interfaces, potentially making them available to wider audiences. In 2010, out of all the specifications proposed by the Open Geospatial Consortium (OGC), the Web Map Service (WMS) was the most widely used standard for sharing spatial data over the web. One common way to increase the performances of Web Map Services is to pre-generate imagery, limiting the possible geographic extent of the images to a finite set of tiles, which are then served as static images. This reduces the flexibility of a full WMS to a tiled WMS which reduces server load and increases scalability (Blower, 2010). As map data is becoming richer and GIS providers are striving to present multi-layered data in a variety of projections and map zoom levels, traditional Web mapping techniques become too inflexible (Boulos *et al.*, 2010). The network infrastructure and hardware specification for Internet GIS provide high-speed communication channels for publishing and accessing

geographic information through the computer network. Internet GIS is a means for GIS users to exchange GIS data, conduct GIS analysis, and present GIS output in the form of maps, the world of internet GIS can also provide simple visualisations, spatial query tools, or geospatial analysis functionality (Karnatak *et al.*, 2007). The open source movement has had a significant impact in the influence of Web GIS to the users of different platforms. The world of Free and Open Source Software (FOSS) has made the software available to all, free of cost (Agrawal and Gupta, 2017).

Non-spatial web applications usually contain only a web server, but in the case of web GIS, there is an additional server called the data or map server for spatial data. This server handles all the geospatial data, provides geospatial data compatible services such as WMS and Web Feature Service (WFS), and is able to perform GIS functionalities such as editing and routing. The client server architecture follows a traditional network architecture and has different approaches such as thin client, thick client, and hybrid architecture (Agrawal and Gupta, 2017), and are explained as follows by Agrawal and Gupta (2017).

2.6.2. Thin Client Architecture

This architecture has minimal resource requirements (see Figure 3) from the client side as most of the processing is done on the server side. When the client makes a request to the server, the server generates a response which in a simple form may be a map that is generated using the database of geospatial data. This spatial response comes in a web-friendly file format so that it is able to be rendered by the web browser of the client. The client cannot directly call the GIS server; therefore, it requires some interpretation by an interpreter such as the Common Gateway Interface (CGI) or some other gateway script.

This architecture has several advantages as there is no responsibility on the client side. The client just needs resources to send the HTTP request to the server and display the server produced result. It is a low-cost solution as there is not much capital investment needed from the client side. Unfortunately, the main disadvantage of this architecture is the limited functionality on the client's side. This means that a new request is made by the client each time a basic map operation such as zooming, panning, and querying is performed. This increases the number of interactions with the server by the client as the client's capabilities are not fully used. A thin client cannot support vector data.

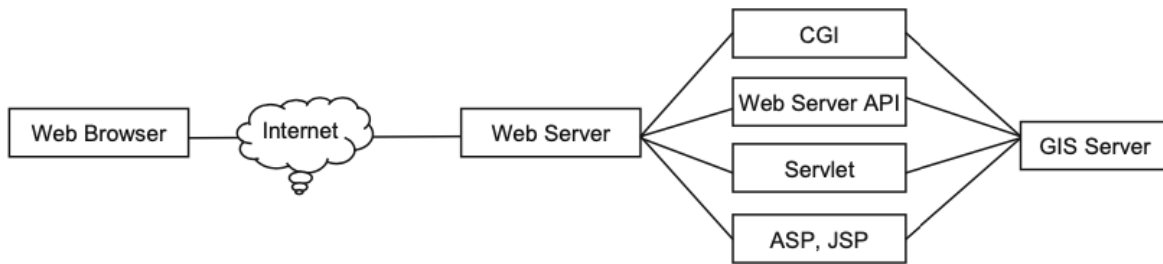


Figure 3: Thin client architecture (Agrawal and Gupta, 2017)

2.6.3. Thick Client Architecture

In this architecture (see Figure 4), the client is treated as a more powerful user, as the browser's capabilities are amplified by plug-ins, applets, ActiveX or JavaScript. This allows the processing to be performed on the server's side, as well as the client's side. Plug-ins are executables that operate on a specific data type; however, they must be installed on the client's side beforehand. Raw data is provided from the server and can be processed on the client's side by various operations, for example, the raw data can be modified by a filtering process, the filtered data can then be mapped and then this can be rendered.

The advantage of a thick client is the running of the client-side system, even when there is low connectivity with the server. This is because the raw data is provided by the server that can be used for different purposes. This also means that the client does not need to perform small requests to the server for small actions. This in turn lessens the load on the server as some of the processing is performed on the client's side. As operations are performed on the client's side, the raw data and code increase the download time and consume more of the client's machines resources.

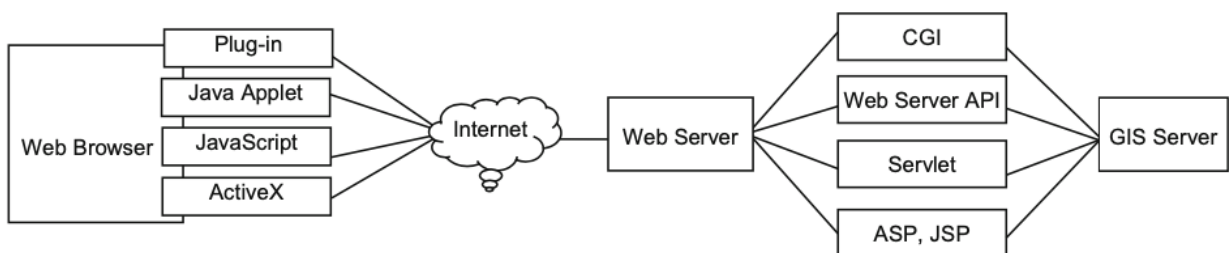


Figure 4: Thick client architecture (Agrawal and Gupta, 2017)

2.6.4. Hybrid Architecture

This architecture is a combination of the thin and thick client architectures. Some of the tasks that are related to data manipulation are performed on the server's side, while other tasks that are related to user interaction, such as panning and zooming, are performed on the client's side. It uses a combination of client and server-side technologies.

2.6.5. Spatial Cloud Computing Architecture

A thick or thin can make a request that is related to projections, visualisations or geoprocessing. A request from the user requires processing power from the host or a physical computing node that is in the Cloud. In this Cloud, there is an extra layer of virtualisation that affects the execution and hosting of the cloud-based application service. Cloud service providers provide the requested resources to the client using various cloud functionalities such as load balancing, CPU allocation, memory allocation, storage allocation and virtual machine provisioning. In a high-performance computing environment, GPU clusters provide capabilities to perform visualisations and analysis of spatial data. Virtualisation assists in resource sharing as they remove the limitations of the physical machine and achieving greater flexibility. The data uploaded to the cloud can be accessed more easily by the users with the help of the internet. Users can use this computing power the same way they use electricity and water, they pay according to their usage. However, it should be noted that the desired architecture of any Web GIS depends on the desired functionality. Figure 5 shows the cloud computing architecture according to Agrawal and Gupta (2017).

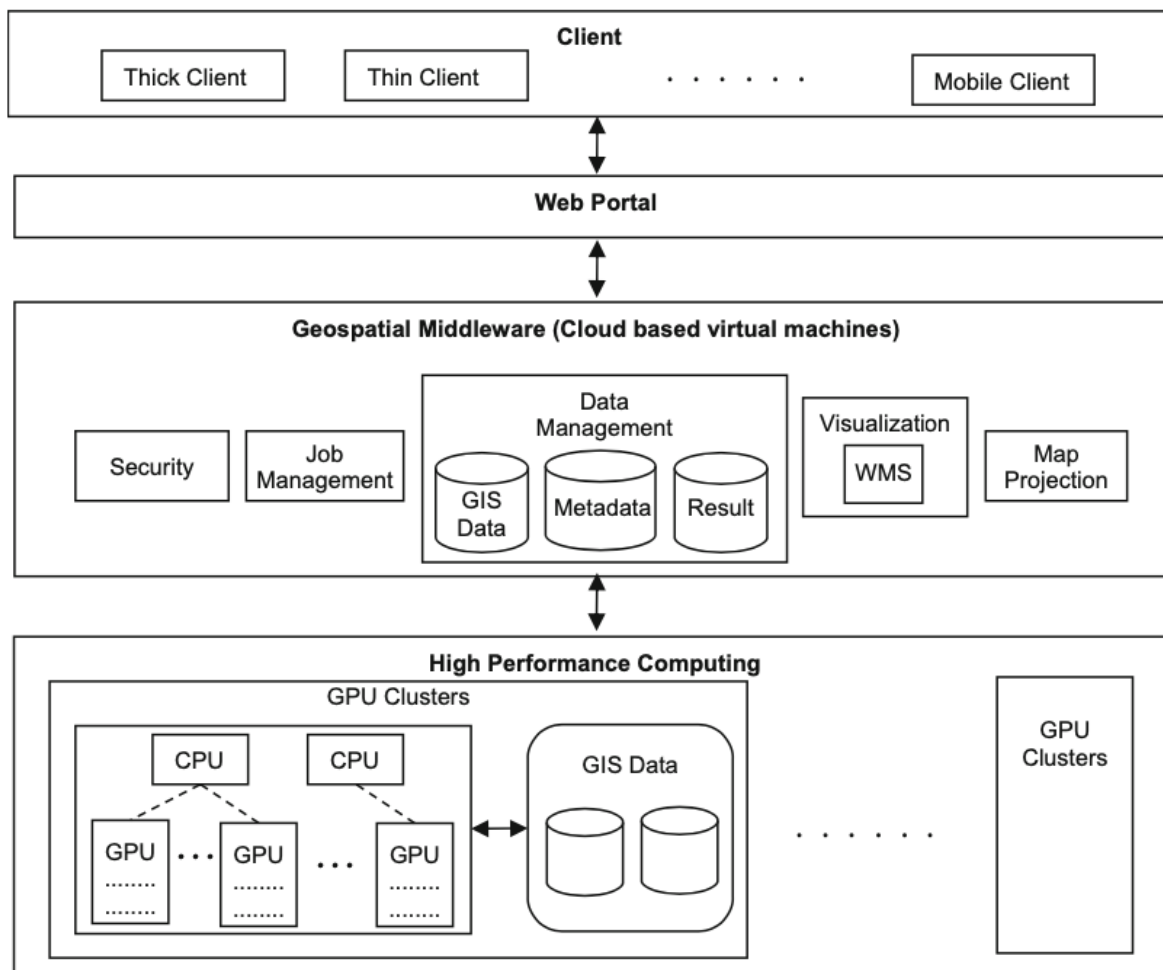


Figure 5: Spatial cloud computing architecture adapted by Agrawal and Gupta (2017)

There are continuous changes in the world of web GIS architecture in order to adapt to the changing needs and technological advances. There is a modern-day shift towards peer-to-peer computing in the form of the evolution of Internet of Things (IoT), where there is a need to perform a certain level of computing on the network instead of transferring all computing capabilities to the cloud (Agrawal and Gupta, 2017).

2.7. Geoportals and Spatial Data Infrastructures (SDIs)

Humans have always exchanged geographic information, but the practise has grown in recent years with the popularity of the Internet and with the growth of geographic information technology. There has also been a need for the sharing of this information from the dawn of time from hunter-gatherers sharing knowledge about resource rich location, to 16th century sailors who shared knowledge of the sea routes to richer lands, to the 19th century explorers of the polar regions to 20th century military commanders, who share information on troop movement and enemy locations (Goodchild, Fu and Rich, 2007). Goodchild, Fu and Rich (2007) also identified that the accumulation and sharing of information about the earth is

central to the field of geography. With the rapid expansion of technology, can it be ensured that all have access to this geographic information?

Interoperability is becoming essential for today's geographic information systems (GIS). Geographic information is usually available as datasets stored in traditional databases and accessible via a GIS (Gunay, Akcay and Altan, 2014). The amount of spatial data and geographic information have increased at an exponential rate due to rapid advances in space and other Earth observation technologies, as well as photogrammetric and surveying methods. The era of big Earth data has now arrived, and challenges have been encountered to manage this huge amount of spatial data in a systematic and logical way and at the same time, to effectively and efficiently interact with the end users (Guo *et al.*, 2017; Jiang *et al.*, 2019). Due to this increasing number of published geospatial resources available on the Web, the search and discovery of useful and up-to-date resources have become a difficult task. Most generally, search and discovery of geospatial resources, such as geospatial data and services, are performed through geoportals (Dareshiri, Farnaghi and Sahelgozin, 2019).

Over the past few decades, the concept of a geoportal has emerged as a significant solution for accessing and sharing spatial data and geographic information and they play a central role in enabling users to access and manage huge volumes of spatial data and geographic information through the internet (Jiang *et al.*, 2019). Jiang *et al.* (2019) performed an analysis of Google searches of the keyword 'geoportal', which revealed the search count was over four million hits in 2017; much higher when compared to previous literature of Goodchild, Fu and Rich (2007) indicating only 182 000 hits in 2007.

A portal is a web site that acts as door or a gateway (Jiang *et al.*, 2019) to a collection of information resources that include data sets, tutorials, tools, and an organised collection of links to many other web sites (Maguire and Longley, 2005). To further define a portal, Maguire and Longley (2005) said a portal is a web environment that allows an organisation or a community of information users and providers to aggregate and share content. According to Tait (2005), a geoportal is a website that presents an entry point to geographic content on the web, or a website where geographic content can be discovered. A geoportal is a specialised kind of portal that provides complex functionality through a simple user interface for a user in a specific application domain (Rautenbach, Coetzee and Iwaniak, 2013). A geoportal can be further defined as a point of access to spatial data and geo-information. It is able to provide a geospatial data inventory linking to an inclusive collection of spatial data, geographic information, online services and data processing tools. The geoportal is designed to provide information about geospatial data and the data owners, but not the data itself. The data as

well as the related metadata information remains with the data providers leaving full control of the data to the information provider (Mehdi *et al.*, 2014). A geoportal can be considered an entry point and the human-to-machine interface of the data and information management system (Jiang *et al.*, 2019). According to Mehdi *et al.* (2014) there are two main types of geoportals, viz. “discovery geoportals” and “data delivery geoportals”. The only difference between the two is the ability to access real data in a data delivery geoportal but this is setup by the owner of the data (Mehdi *et al.*, 2014). This is similar to the two groups that Maguire and Longley (2005) identified as they said that a geoportal can be divided into either a “catalogue geoportal” or an “application geoportal”. Catalogue geoportals are concerned with organising and managing access to geographic information, whereas application geoportals provide online, dynamic geographic web services, such as routing services. Besides the term geoportal, the term spatial web portals is also widely used for defining a web-based gateway to access and manipulate geospatial data via the internet.

A geoportal has the ability to dramatically expand the availability of location-based data to expert and non-expert users for the reviewing, editing and analysis of this data. Task-oriented, intuitive applications are available to the end users via the internet, without any software having to be installed on the client’s computer. As a geoportal is a type of web portal used to find and access geographic information via the internet, they are important for effective use of GIS and are a key element of a Spatial Data Infrastructure (SDI) (Karabegovic and Ponjavic, 2012). A prominent feature of all SDI geoportals is a catalogue service for publishing and accessing metadata (Maguire and Longley, 2005). Geographic information providers, such as governments and companies, use geoportals to publish geospatial metadata of their geographic information. Geographic information consumers, either professional or occasional, then use these geoportals to search and access the information they need. Karabegovic and Ponjavic (2012) go on to say that geoportals serve an increasingly important role in the sharing of geographic information, and can avoid duplicated efforts, inconsistencies, delays, confusion and wasted resources. Spatial data can be searchable if the correct and required information, so-called metadata, are attached to it. This metadata helps people who use geospatial data to find the data they need, and to determine how best they can use the data. Metadata is a key factor in supporting the discovery, evaluation, and application of geospatial data beyond the scope of the original project (Mehdi *et al.*, 2014). Geoportals and SDIs have made a major contribution to simplifying access to geographic information, and in doing so, they have helped to encourage and assist people who want to use geographic concepts, databases, techniques, and models in their own work (Maguire and Longley, 2005).

The role of a geoportal in an SDI is an important one. Typically, in a geoportal, a database is populated with metadata records of published geographic information services. Users then execute queries against the database either from a lightweight web client, or a heavier-weight desktop GIS client, on the basis that they have permission to do so, and possess an internet connection. This allows the users to discover what services are available that meet the criteria such as topic, geographic extent, and time period. The services can then be used directly by the client in their lightweight web client or a heavier-weight desktop GIS client. Figure 6 provides a visual representation of what this means and how it is performed.

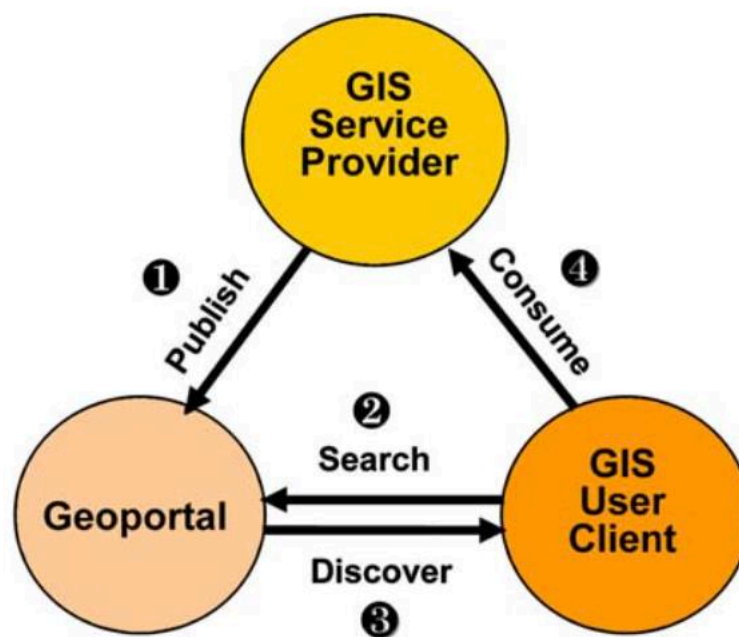


Figure 6: The role of a geoportal in an SDI (Maguire and Longley, 2005)

Figure 7 provides a visual of the system architecture for an SDI geoportal. Geoportals are built using the World Wide Web (WWW) infrastructure technology, and either commercial or open source off the shelf GIS software. Network communication between clients and web servers use Hypertext Transmission Protocol (HTTP) meaning that technically, a geoportal is a master website connected to a web server, which contains a database of metadata information about geographic data and services. The data services are exposed as web services that are self-describing web applications that are transmitted over an HTTP connection (Maguire and Longley, 2005; Tait, 2005; Mehdi *et al.*, 2014). Additionally, the geographic industry has published their own geographic web services which work in conjunction with these other standards. Organisations that are involved in the publishing of the geographic web standards include the Open GIS Consortium (OGC, <https://www.ogc.org>) and the International Standards Organisation (ISO, <https://www.iso.org/home.html>). Typical geographic web service functionality include map rendering, feature streaming, data projection, geographic

and attribute-based queries, address geocoding, place name searches, metadata query and management, data extraction, and so on (Tait, 2005). GeoNetwork is a popular examples of geoportal application.

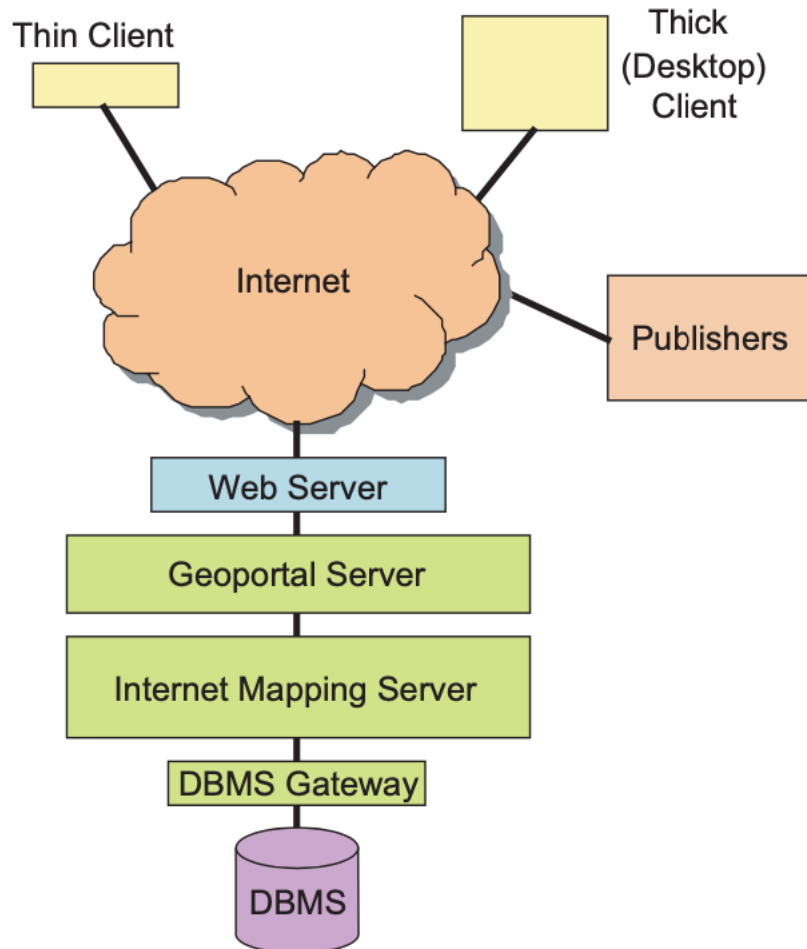


Figure 7: System architecture for an SDI geoportal (Maguire and Longley, 2005)

Geoportals are used in various professional fields such as: tourism; dissemination of information on geological and cultural heritage; information management in land use systems; decision support systems; monitoring of technogenic systems; rational use of land and resources; the analysis of ecological territories; emergency prevention and disaster reduction, traffic flow management and road traffic management, to name a few (Yamashkin and Zanozin, 2019). Two of the most common examples of modern-day geoportals include the European Community geoportal, Infrastructure for Spatial Information in the European Community (INSPIRE) and UNSDI, the United Nations Spatial Data Infrastructure. As a standard, Spatial Data Infrastructures (SDIs) facilitate the sharing of public geospatial data within a country and are maintained by regional and local governments for sharing data at different administrative levels (Coetzee *et al.*, 2011). SDIs are particularly valuable in terms of solving interoperability problems at different levels. INSPIRE defines the directives for 34

spatial data themes among European countries as it also meets the need for sharing, access and exchange of geographic data among member states by implementing a common data specification (Gunay, Akcay and Altan, 2014). DataFirst is an example of an African data portal, as it is a research data services dedicated to giving open access to data from South Africa and other African countries. This open data portal gives researchers access to high-quality disaggregated data from African countries. Currently, there are 470 datasets present in this portal.

2.8. Research Data Archives/Libraries

Today's scientific activities require collaboration among parties that are widely distributed. Collaboration is often a cross-discipline practice and requires access to a variety of data and to specialised tools that support the analysis and processing of this data. Libraries and archives have always been the primary organisations entrusted to manage human knowledge and culture. When the advances in computer science allowed the dealing with the digital representation of documents, libraries were involved in using the potential of this digital revolution, thus "Digital Libraries" soon became the term to indicate the digital counterpart of traditional libraries (Candela, Castelli and Pagano, 2011). Like all processes in the world of archiving and libraries, the archiving of research data is limited by the constant changing environment of modern information technology and is therefore undergoing rapid changes (Doorn and Tjalsma, 2007). The daily work of many research communities is characterised by an increasing amount and complexity of data. This makes it increasingly difficult to manage, access and utilise data to ultimately gain scientific insights (Grunzke *et al.*, 2016). Besides research data being complex, they are often expensive, and time-consuming to replicate (Ng'eno and Mutula, 2018). Research datasets include all manner of objects that can range from web pages to sensor data to publications or multi-media material, experimental data, to tools that manipulate this data, and computing and storage resources; and it can originate from every domain and be used in diverse contexts (Candela, Castelli and Pagano, 2011; Ribeiro *et al.*, 2018). As knowledge evolves, newer methods and approaches are developed and influenced by researchers from different disciplines, resulting in inter-disciplinary methods and approaches (Coetzee *et al.*, 2011). Having computer software that allows interactive displays and visual context can save users both time and effort which then means an increase in productivity (McCarthy and Graniero, 2006).

Digital Libraries have greatly evolved since their first appearance and today they have become complex networked systems that are able to support communications and collaborations among different scientific communities, especially with the recent movement towards the

notion of e-Infrastructures. This movement makes it possible to support the dynamic construction and maintenance of digital libraries, which can, in this context, be called Virtual Research Environments (VREs) (Candela, Castelli and Pagano, 2011, 2013). Digital Libraries deal with “digital objects” such as documents, images, videos, programmes, and any other kind of multimedia that the users might find useful (Candela, Castelli and Pagano, 2011). Digital libraries could also become a major means to support the entire life cycle of scientific production, which consists not only of the retrieval of relevant information, but also in the analysis of this information and the production of new content that is then published and distributed for use by others (Candela, Castelli and Pagano, 2011). Web archiving initiatives have started to deal with the problem of making the selection and describing of the data a more light-weight and semi-automated task. The main purpose of digital research archives and repositories is not to preserve materials, but rather, to provide access to what is preserved (Doorn and Tjalsma, 2007). Research data archives face the same problem, where some collections of the datasets will be much better curated for than other datasets (Ribeiro *et al.*, 2018). Research data archives are built on the premise that science is not an individual endeavour, but rather, researchers aim to achieve provisional results for others to criticise and build upon. However, for this continued critique to happen, researchers are required to make it transparent and understandable the manner in which they came to their conclusions. Sharing of this research data plays a major role in achieving the reproducibility of research (Kinder-Kurlanda *et al.*, 2017). Unfortunately, there are reservations to the sharing of this research data as researchers fear of opening up their research to attack, legal limitations due to intellectual property and data protection issues, as well as the effort that is required to prepare the data for reproducibility and reuse. Specialised data archives play an important role in the sharing of this research data, however, disciplinary boundaries are becoming less important when it comes to the sharing of research data as an increasing number of researchers are using mixed methods to utilise data from different types and from various sources (Borgman, 2012; Kinder-Kurlanda *et al.*, 2017). Archiving resources from the academic world has always been an activity that takes place at a distance and the University’s archives are usually maintained by the universities themselves, often in the libraries (Doorn and Tjalsma, 2007). Research data are valuable resources that need to be properly managed by research institutes and besides research data being sophisticated and complex, they are expensive, and time-consuming to replace (Ng’eno and Mutula, 2018).

Ng’eno and Mutula (2018) defines research data management (RDM) as the collection, organisation, validation, and preservation of the data for analysis, discovery, sharing, reuse, and transformation. Research data management consists of different activities and processes associated with data creation, storage, security, preservation, retrieval, reuse and sharing

taking into account the technical capabilities, ethical considerations, legal issues, human resource capability and government frameworks. Research data management brings benefits to researchers and research institutes such as the ability to share research data, which minimises the need for repeated work. The data that is collected allows for comparison and co-analysis with data from multiple sources, leading to new and powerful insights. Long-term preservation of data allows for the data to be validated and this increases the credibility and transparency of the research data used and the project itself (Ng'eno and Mutula, 2018), to name a few. Metadata ought to help users to assess the quality, the fitness of use, of geospatial data, which in turn reduces the risk of data misuse (Devillers, Bédard and Jeansoulin, 2005) and plays a role in making data available for the long-term (Grunzke *et al.*, 2016). The ease with which digital research data can be stored, disseminated, and made accessible online means that many institutions ought to strive to share research data to enhance the impact and visibility of the research (Ng'eno and Mutula, 2018).

Wright, Brunner, and Nebel (2018) identified that research data archives need to satisfy at least three processes, namely, storing files, organising and sharing of files. The main purpose of an archiving system is to store files in a way that they are accessible to the target users over a long period of time, which supports Doorn and Tjalsma's (2007) statement of that the main purpose of digital research archives and repositories is not to preserve materials, but rather, to provide access to what is preserved. A reliable hardware infrastructure must be present and a description of the data, which makes searching and retrieving as easy as possible. Additional to the metadata describing file types, document names, authors, keywords, and other data type related information. Documents can also be linked to one another using descriptive links such as *is documented by* and *is related to*. With this type of linking, a network of documents is created, providing easy access to related data of any given file. When an archive grows, it becomes important to be able to organise documents in a more sophisticated manner such as linking documents by keywords and authors than simply archiving them and linking related documents to each other. Lastly, one requirement from funding agencies is to share and publish data. Currently, this requires the accessing party to have the appropriate permissions and login rights to the system. To manage who can view which documents, access management based on user permissions ought to be implemented.

South Africa is the leading country in a mass of African countries in embracing RDM (Van Deventer and Pienaar, 2015; Ng'eno and Mutula, 2018). The University of Cape Town (UCT) has established an e-Research centre to work and partner with researchers in finding an IT solution for their research work while the University of South Africa (UNISA) has completed an investigation into RDM as part of the plan to establish data management. The University

of Pretoria, Stellenbosch, and Witwatersrand are at different planning and implementation stages (Van Wyk and Der Walt, 2014; Van Deventer and Pienaar, 2015). South Africa has a number of data repositories that have been established to manage research data such as the South African National Park, the National Health Information Repository and Data Warehouse, and Data Intensive Research Initiative of South Africa (DIRISA) with DIRISA aimed at promoting RDM in the country. The National Research Foundation (NRF) has been involved in many initiatives to allow the sharing of research outputs, datasets research support and knowledge networking databases, which contribute to knowledge generation for the support and promotion of research development (Ng'eno and Mutula, 2018).

2.9. Virtual Research Environments

In today's circumstances, scientific activities require a large amount of collaboration amongst all those involved that are widely geographically distributed and cross-discipline. Being cross-disciplined means there is a demand for access to a variety of data and specialised tools that support the analysis and processing of this data. Scientific workflows are attracting considerable attention in the scientific community (De Roure, Goble and Stevens, 2007). Such collaborations rely heavily on a wide range of different and constantly evolving application resources (Candela, Castelli and Pagano, 2011), and that is where Virtual Research Environments (VREs) are of relevance. VREs have been proposed to close the gap between service providers and the scientific communities. Very often these portals are built to serve the needs of a specific community (Assante *et al.*, 2016). VREs have been used by a number of universities and research institutions as technological frameworks to facilitate the collaborative nature of research projects. However, of the VREs that have been built thus far, have tended to be either precise configurations for specific research projects, or systems having very generic functions (Van Wyk, Bothma and Holmner, 2020).

After an extensive literature study prepared by Van Wyk, Bothma and Holmner (2020), this study determined that internationally there is no one formal definition of a Virtual Research Environment (VRE), but instead that there are a number of ways that it can be defined, such as a framework into which tools, services, and resources can be plugged, a digital infrastructure and services that enable research to take place (Fraser, 2005); a virtual working environment, created on demand, in which communities of research can effectively and efficiently conduct their research activities (Thanos, 2013); an enabler of research within the virtual, multi-disciplinary and multi-organisational context and an intricate part of eResearch (Van Deventer, M., Pienaar, H., Morris, J. and Ngcete, 2009); a technological framework providing access to data, tools and services, which in turn give access to a wider research

infrastructure (Carusi and Reimer, 2010); a specific class of usage of e-infrastructure, as well as similar intangibles (Dunn, 2009); and a vague layer of expertise and the best practices, standards, tools, collections and collaborative environments that can be broadly shared across communities of enquiry (Unsworth, 2006). Following the literature about VREs, a VRE can be described as a set of web applications, online tools, systems, and processes inter-operating to facilitate or enhance the research process within and without institutional boundaries; it enhances collaborative research activities beyond geographical barriers (Carusi and Reimer, 2010). Barker *et al.* (2019) later defines VREs as various kinds of community-developed digital interfaces to advanced technologies that support research that are used in a wide variety of scientific domains, from high-energy physics, to social sciences, to the analysis of climate change consequences (Gordov *et al.*, 2016). Depending on the context, the environments can be referred to as either VREs, Science Gateways, Collaboratories, Digital Libraries/Archives, Collaborative e-Research Communities, Collaborative Virtual Environment, Virtual Organisation, Virtual Research Community or Inhabited Information Spaces (Carusi and Reimer, 2010; Candela, Castelli and Pagano, 2013).

Carusi and Reimer (2010) identified various challenges to VREs such as sustainability; funding; business models and community support; as well as other various barriers to the use of VREs, such as: the lack of support; unsuitability for research practice; the reliability of technology; the number of active users; the legal and ethical issues of VREs; and the data within the VRE as well as the interdisciplinarity and different ways of working between researchers. At the same time, they highlighted some of the desirables of a VRE being the awareness raising, an international VRE forum, national and international integration of resources, authentication and single sign-on, usability, and integration of Web 2.0 technologies within the VRE and common standards.

There are benefits to using a VRE such as student collaboration over a distance, exchange of information and data amongst the students, access to skills, knowledge, research data, and cooperative writing of academic material such as dissertations or journal articles. For a VRE to be successful, there needs to be clear ownership of all the data present, mutually agreed project plan and clearly defined objectives and responsibilities. It is also important that the software and technology in the VRE be easy and simple to use and it must be expressed to the students how the VRE can simplify their workflow and allows to distribute work. VREs provide innovative and dynamic research environments that support scientists to access data, software and other resources. By tailoring digital environments to the community's needs, VREs perform a key role in integrating elements of the e-infrastructure landscape, providing online access to software, data, collaboration tools and high-performance computing, which

increases research impact. A popular example of a VRE is CANARIE (<https://www.canarie.ca>) as it allows Canadian researchers to access research data, tools and other collaborators (Barker *et al.*, 2019). Another example of a popular VRE is the ^{my}Experiment VRE (<https://www.myexperiment.org/home>), which provides a personalised environment which enables users to share, re-use and repurpose experiments (De Roure and Goble, 2007).

2.10. Learning Management Systems

Technology has influenced all aspects of society, none more so than the field of education. The manner in which universities perform administration, lecturers teach, and students learn are shaped greatly by technological advancements (Gautreau, 2011). Wang (2002) and Gautreau (2011) both indicated that the World-Wide Web provides unique opportunities for educators to create engaging and authentic learning contexts and activities for learners. Online teaching and learning have been around for years with the number of courses that are offered online at a distance increasing rapidly. Many of the classroom-based higher education programmes that meet in person during the course of a semester use online technologies, even more so during this current COVID-19 pandemic. Students and lecturers use distance education technologies to share files, present content, communicate with each other between classes and carry out other teaching and learning activities (Beatty and Ulasewicz, 2006).

Learning Management Systems (LMS), also referred to as a Course Management System (CMS) or a Virtual Learning Environment (VLE) (Rhode *et al.*, 2017) are technological learning environments that support online course delivery as they offer wide-ranging synchronous and asynchronous services that support collaborative learning (Rachel and Parthasarathy, 2016). Gautreau (2011) referenced Laster (2010), who previously defined a learning management system as a self-contained web page with embedded instructional electronic tools such as a discussion board, files, grade book, emails, announcements, assessments and other multimedia elements that permit faculty to organise academic content and engage students in their learning as it provides access to student-centred teaching approaches, increased accessibility, assessment and evaluation features, and improved management of course content, administrative tasks and organising content (Simpson *et al.*, 1999; Mullinix and McCurry, 2003; Laster, 2010; Almarashdeh, 2016; Rhode *et al.*, 2017).

Recognised benefits of using an LMS is the ability to instruct online using a wide range of modalities to meet learners' diverse needs (Mullinix and McCurry, 2003), the distribution of reading materials and handouts are completed through electronic documents and files, which reduces the amount of additional tasks (Pittinsky, 2004), enables lecturers with little technical

skill to deliver instructions to students at a distance and it enables secure online collaborations among lecturers and students (Rhode *et al.*, 2017). Some lecturers feel that current learning management systems are too limited in functionality and have proposed a next generation LMS which is often referred to as a “next generation digital learning environment” (NGDLE) (Brown, Dehoney and Millichap, 2015).

Moodle (<https://moodle.org>) and Blackboard (<https://www.blackboard.com>) are both popular examples of course management systems. Moodle is the leading open source learning management system software package. It is a free and open source software package designed to help educators create effective online learning communities. It can be downloaded on a computer that acts as a localhost or to a webhost and can scale up to suit up to 40 000 students (Beatty and Ulasewicz, 2006). Some of the key features or online tools that Moodle encompasses include assignment, forum, chats, quiz, blogs and Wiki, glossary and Rich Site Summary (RSS) feed (Rachel and Parthasarathy, 2016). Blackboard is the leading commercial learning management system software package available. It is a software application for institutions, where their primary focus is teaching and learning. Blackboard is intuitive and easy-to-use, with capabilities in three major areas, viz.: instruction, communication, and assessment (Beatty and Ulasewicz, 2006).

Figure 8: The University of Pretoria's learning management system, Blackboard (<https://clickup.up.ac.za>)

2.11. Content Management Systems

Content management is usually viewed as a digital concept in this day and age, but it has been around for as long as content has been. For as long as humans have been creating content, they have been searching for solutions to manage it. The Library of Alexandria was an early attempt at managing content. It preserved content in the form of papyrus scrolls and codices, and controlled access to them. Effectively, librarians were the first content managers. The need for content management didn't begin with the World Wide Web, but instead, it simply shifted into fast-forward when the Web was born in the early 1990s. It was at this moment, the ability to create and share content became more accessible than before. Content is created by humans, for humans (Barker, 2016).

Barker (2016) defined content as information that is produced through an editorial process and is intended for human consumption via publication. Content is created and managed, then it is published and delivered. Content has value in the future, where it is used for years and can continue to provide value for an organisation or individual far into the future. Barker (2016) considers content is an investment in the future, not a record of the past.

There are many different definitions of a Content Management System (CMS) but there is one universally accepted definition of a content management system as “a system that lets users apply management principles to content” (Patel, Rathod and Parikh, 2011). A content management system is a software package that provides some level of automation for the tasks required to effectively manage content. It is usually a server based, multiuser software that interacts with content that is stored in a repository with this repository; either located on the same server or in a separate storage facility elsewhere (Barker, 2016). A more recent definition of a content management system by Cabot (2018) is that a CMS is any system that facilitates the creation and publication of digital content. Martinez-Caro *et al.* (2018) extended this definition by saying that a content management system is a software platform that is generally used when a website is needed that requires different user roles, but where at the same time, there is a lack of web programming knowledge. A content management system performs content control functions such as permissions, state management and workflow, versioning, dependency management and, search and organisation; it allows for content reuse, automation and aggregation; and increases efficiency. However, a content management system does not create content, does not format content and does not provide governance as to how to store and manage data (Barker, 2016).

There are four main types of content management systems that are currently available, namely: Web Content Management (WCM), Enterprise Content Management (ECM), Digital

Asset Management (DAM) and Records Management (RM) (Barker, 2016). A web content management system is focused on the management on content intended for mass delivery via a website. An enterprise content management is focused on the management of general business content not intended for mass delivery and it excels in collaboration, access control and file management. A digital asset management system focuses on the management and manipulation of rich digital assets such as images, videos and audio for usage in other media and it excels in metadata management. A records management system focuses on the management of transactional information and other records that are created as a by-product of business operations and excels at access control. A digital asset management system can be further divided into two more categories, namely: Learning Management System (LMS) and Portals. Only some of what an LMS can do is specific and unique to an LMS. Many different web content management systems have add-ons and extensions that claim to turn them into an LMS (Barker, 2016).

The three most popular open source content management systems are Joomla!, Drupal and WordPress (Patel, Rathod and Parikh, 2011; Martinez-Caro *et al.*, 2018). Joomla! (<https://www.joomla.org>) is one of the most powerful open source content management systems, with core features such as user management, media manager, banner management, contact management, polls, search, web link management, content management, newsfeed management, template management, integrated help system, and powerful extensibility. According to W³Techs a total of 2.3% websites currently use Joomla! Drupal (<https://www.drupal.org>) is a platform for building robust, flexible websites and allows users to update their web pages without any technical knowledge with core features that permit to administrate, build, collaborate, connect, create, design and display, extend and organise, and find. According to W³Techs a total of 1.5% websites currently use Drupal. The third open source content management system is WordPress (<https://wordpress.com>). WordPress was initially designed as a blogging platform, but recently, it has changed into a more useful content management system with core features such as custom post types, themes, cross-blog communication tools, spam protection, full user registration, password protected posts, easy importing and menu management. According to W³Techs and Cabot (2018), a total of 38,4% websites currently use WordPress making WordPress the most popular open source content management system on the market.

2.12. F.A.I.R Data Policy

Providing other researchers with access to the data that is collected encourages knowledge discovery and improves research transparency as knowledge discovery is one of the biggest challenges of data-intensive sciences. To tackle this challenge, a consortium of scientists and organisations authored a publication in March of 2016 titled “FAIR Guiding Principles for scientific data management and stewardship” to optimise data sharing and reuse by humans and machines. The FAIR principles describe how research outputs ought to be organised so they can be more easily accessed, understood, exchanged, and reused. Major funding bodies, including the European Commission, promote FAIR data to maximise the integrity and impact of their research investment. Good data management is not a goal, but rather, is the key conduit leading to knowledge discovery and innovation, and to subsequent data and knowledge integration and reuse by the community after the data publication process. Unfortunately, the existing digital ecosystem surrounding scholarly data publication prevents users from extracting maximum benefit from our research investments (Wilkinson *et al.*, 2016).

The four FAIR foundational principles are findability, accessibility, interoperability, and reusability. Each one of these foundational principles are then broken down into another set of guidelines which further describe the foundational principle. More of this can be seen in Table 4.

Table 4: FAIR data principles according to <https://www.go-fair.org/fair-principles/>

Four Basics of FAIR	Explanation
Findable	The first step in (re)using data is to find the data. Metadata and data should be easy to find for both humans and computers. Machine-readable metadata is essential for the automatic discovery of datasets and services available.
Accessible	Once the user finds the required data, they need to know how they can access the data, this process may possibly include authentication and authorisation.
Interoperable	The data usually need to be integrated with other data. In addition, the data need to interoperate with applications or workflows for analysis, storage, and processing.
Reusable	The ultimate goal of FAIR is to optimise the reuse of data. To achieve this, metadata and data should be well-described so that they can be replicated and/or combined in different settings or scenarios.

Findable

To ensure that the data is findable by both humans and machines, it needs to meet the following four criteria:

F1: (Meta)data is assigned a globally unique and persistent identifier.

F2: Data is described with rich metadata as defined by R1.

F3: Metadata clearly and explicitly includes the identifier of the data it is describing.

F4: (Meta)data is registered or indexed in a searchable resource.

Accessible

To ensure that the data is accessible by both humans and machines, it needs to meet the following four criteria:

A1: (Meta)data is retrievable by its identifier using a standardised communications protocol.

A1.1: The protocol is open, free and universally implementable.

A1.2: The protocol allows for an authentication and authorisation procedure, where necessary such as a login portal.

A2: Metadata is accessible, even when the data is no longer available.

Interoperable

To ensure that the data is interoperable by both humans and machines, it needs to meet the following three criteria:

I1: (Meta)data uses a formal, accessible, shared, and broadly applicable language for knowledge representation.

I2: (Meta)data uses vocabularies that follow FAIR principles.

I3: (Meta)data includes qualified references to other (meta)data.

Reusable

To ensure that the data is reusable by both humans and machines, it needs to meet the following four criteria:

R1: Meta(data) is richly described with a plurality of accurate and relevant attributes.

R1.1: (Meta)data is released with a clear and accessible data usage license.

R1.2: (Meta)data is associated with a detailed provenance (record of ownership).

R 1.3: (Meta)data needs to meet domain-relevant community standards.

These principles refer to three types of entities: data or any digital object, metadata (information about that digital object), and infrastructure. These principles are also only guidelines, and not a fixed standard that can be enforced.

Incorporating these guiding principles into the solution proposed will be an important milestone, as the whole basis of this solution is about making data available to other researchers through data discovery and sharing. Another key factor of why the FAIR principles are important to this project are in the field of metadata and data management for the purpose of longevity.

2.13. Related Studies

After an extensive review of existing literature, a few relevant projects were identified, each solving their problems in a different but relatable way. The literature review highlights the similarities of the works as well as identifying research gaps.

Steiniger *et al.* (2017), presented a case study of the geographic data and document repository of the Chilean research Centre for Sustainable Urban Development (CEDEUS), the CEDEUS Observatory. Besides the infrastructure to host and distribute data, communication tools are an important component of such a data repository service. In this case study, they analysed which things worked well and which things did not work well during the three years of operation. Steiniger *et al.* (2017) identified three components to consider when implementing a geographic data repository for research based on a kind of spatial data infrastructure (SDI). These are: 1) the repository user; 2) the context of implementation; and 3) a spatial data infrastructure as an organisational and technical solution for the implementation of a research data repository. When considering the repository user, one needs to take into account the users' knowledge about geographic data, mapping programmes, and spatial data formats. They also need to consider what data and documents do the users have; the formats in which they are stored; who are they going to share their data with; what activities the data repository user ought to be able to perform, i.e. what functionalities should the web-based platform provide; and the ways in which the repository user ought to be able to contact the repository management team if the need for questions should arise. The context of implementation refers to the platform architecture, the available funds for implantation and operations, technical standards and governance. Platform architecture refers to whether it is a centralised architecture with one central database, or is it distributed, where each section has its own database. The available funds for implantation and operations are the funds available for building and running the research data repository as this will constrain the team size that creates and maintains the repository, the software that can be used due to license costs, and relevant technical equipment, such as servers which are needed to host databases. Funds can also affect the data that is collected and stored in the repository. Technical standards refer to the uploading and downloading of data and documents via services such as the File-

Transfer-Protocol (FTP) service. The content of the repository should be searchable based on keywords or geographical regions which will lead to the utilisation of metadata standards for digital library catalogues such as Dublin Core and ISO 19115, as well as the Open Geospatial Consortium (OGC) standards developed for spatial data infrastructures. Governance means that agreements are required to control and manage the repository as well as the data in it. Two agreements were identified, viz.: 1) rules and protocols about data access and data exchange; and 2) how to address the governance of the repository, for instance, who decides on the focus and direction of the repository development. For this data portal, they used the free and open source software called GeoNode. Components of GeoNode include GeoServer which is used to manage data sources and render maps, the database application Postgres with the PostGIS extension to store vector data, a web map client based on OpenLayers and GeoExt, the spatial data catalogue software PyCSW and Django as the web-framework to provide data and user management functionality via a web browser. They identified that users who have been trained in desktop GIS experienced no problems with GIS data and using GeoNode. The use of GeoNode by an untrained GIS user was limited, as they were able to search and download data but exploring and uploading spatial data required some prior knowledge. The authors encouraged the users to attend a GIS 101 crash course prior to using GeoNode as the basics of geospatial data were taught.

Paes Leme *et al.* (2007) proposed a software architecture for automated geographic metadata generation, a feature with which every geographic catalogue application should be equipped. The authors go on to describe their GeoCatalog tool which is an implementation of the architecture that demonstrates the viability of their approach. The core of their proposal is the metadata *Harvest Service*, which directly retrieves metadata records from a set of external resources. This service extends the OGC Catalogue Service reference *harvestResource* operation, which acts as an automated pre-processing step to metadata cataloguing. The *harvestResource* operation receives as input parameters a source (URL location) of data containers, a *resourceFormat* that specifies a container type, a *responseHandler* that specifies an URL where the operation response should be forwarded to, a *timeInterval* used to refresh the captured metadata, and a *resourceType* that identifies the type of repository to be crawled (Paes Leme *et al.*, 2007). That is to say, if the repository is an FTP location, the URI would refer to a file containing specific connection information such as user, password and root directory. If the repository is a file directory, the URI would just contain a directory name.

Dareshiri, Farnaghi and Sahelgozin (2019) proposed an approach that can improve the functionalities of the conventional geoportals by adding recommendation capabilities to the performance of geoportals. The geoportals with recommendation capabilities are able to

analyse users' activities, and recommend geospatial resources to users based on their requirements and preferences. The authors developed a hybrid recommendation mechanism based on the singular value decomposition (SVD) method to provide the required prediction and recommendation capabilities. Spatial attributes of geospatial resources, along with the users' characteristics are used to improve the quality of the recommendations. The results showed that the recommender geoportal can relatively well predict users requirements and preferences and provide them with useful geospatial resources (Dareshiri, Farnaghi and Sahelgozin, 2019). This is similar to the work done by Iwanaik *et al.* (2011), where the authors present the concept of an intelligent geoportal, seen as a gateway to a spatial data infrastructure intended to fulfil the needs of any particular user by providing a user-friendly interface that hides the details about different data sources and processing. Universities and other research organisations typically have well-established libraries and digital catalogues for scientific literature, but catalogues for geospatial data are rare (Coetzee *et al.*, 2011) and due to this fact, the authors proposed an Academic Spatial Data Infrastructure which is an SDI for research and education.

Yang *et al.* (2016) developed a cloud-based online geospatial information sharing and geoprocessing platform to facilitate collaborative education and research called *GeoSquare*. *GeoSquare* was designed to solve four key problems: 1) geospatial information resources (GIRs) registration and multi-view query functions allow users to publish and discover GIRs more effectively; 2) online geoprocessing and real-time performance status checking to help users process data and conduct analysis without pre-installation of cumbersome professional tools such as desktop GIS on their own machines; 3) a service chain planning function that enables experts to contribute and share their knowledge with community members through workflow modelling; and finally, 4) user inventory management allows registered users to collect and manage their own GIRs, monitor their performance status, and track their own geoprocessing histories. To enhance the flexibility and capability of *GeoSquare*, distributed storage and cloud computing technologies were employed. To support interactive teaching and training, *GeoSquare* adopts the rich internet application (RIA) technology to create a user-friendly graphical user interface (GUI). The results showed that *GeoSquare* can integrate and promote collaboration between dispersed GIRs, computing resources and people. Consequently, educators and researchers were able to share and exchange resources in an efficient and balanced way (Yang *et al.*, 2016).

gCube (<https://www.gcube-system.org/>) is a software system that promotes seamless access to research datasets such as data, services, and computing across the boundaries of institutions, disciplines, and providers to favour collaborative-oriented research tasks (Assante

et al., 2016). *gCube* owns services that include the services of a spatial data infrastructure (SDI) by relying on state of the art technologies and standards. It offers standard-based services for data discovery, such as a catalogue, storage and access to a group of repositories, and a visualisation such as a map. The catalogue service enables the discovery of geospatial data that is located in dedicated repositories by relying on GeoNetwork. For data storage and access, *gCube* offers a group of repositories based on GeoServer. Put simply, the infrastructure hosts a number of repositories and a GIS Publisher Service that enables a seamless publication of geospatial data while guaranteeing load balancing, failure management, and automatic metadata generation. It relies on an open set of back-end technologies for the actual storage and retrieval of the data. Due to this fact, the GIS Publisher Service is designed with a plug-in-oriented approach, where each plug-in interacts with a given back-end technology. For data visualisation, the infrastructure offers GeoExplorer and GIS Viewer, both of these applications are developed by ESRI, two components dedicated to support the browsing and visualisation of geospatial data. In particular, the GeoExplorer is a web application that allows users to navigate, organise, search and discover layers from the catalogue via the Catalogue Service for the Web (CSW) protocol. The GIS Viewer is a web application that allows users to interactively explore, manipulate and analyse geospatial data (Assante *et al.*, 2016).

Chapter 3 – The Requirements of the Solution

3.1. Chapter Overview

This chapter focused on the context in which architectural students collect, manage, and share all their spatial and non-spatial data. This context will provide the insights used to draw up the functional and non-functional requirements. The context (3.2) is a detailed scenario of the processes that are followed by the architectural students. Following the context, the process that was followed to identify and expand on the requirements will be described in detail in 3.3, as well as the way in which the workshops played a vital role in this process. Once all the functional and non-functional requirements were identified (3.4), they were then written up in a way that describes the category under which they fall, the nature of the requirement, and a description of the requirement. Lastly, the chapter ends with an evaluation of GeoNetwork and GeoNode in section 3.5.

3.2. Context

Postgraduate architectural students the University of Pretoria in South Africa and Chalmers University of Technology in Sweden routinely collect fine-grained visual and qualitative spatial data while working on their designs projects, either through transect walks, participatory action research, and community mapping and participatory geographic information systems (PGIS) methodologies. These architectural students attend what is called an architectural studio and that is usually where they do their work and design their models. An architecture studio is a class in an undergraduate or postgraduate architecture program in which students receive hands-on instructions in architectural design. Typically, a studio class includes distinctive educational techniques such as ‘desk crits’, where the student’s project is critiqued at their desk, and ‘juries’ involving meetings of students and tutors to openly discuss their projects and any other potential ideas. This class is usually well equipped with big drafting tables for one or multiple students, pin-up boards for design posters, and even a smart board as architecture is becoming more and more digital.

In previous years, the architectural students have collected data and then used this data to produce information rich maps about the area they are studying. The architectural students try to use data collection tools such as EpiCollect5 in Mamelodi, South Africa and Maptionnaire in Hammarkullen, Sweden in order to as accurately as possible collect the data, but they struggle to use the tools as they are unfamiliar with them and also struggle to differentiate when a tool is suitable for their needs and when a tool is not suitable for their needs. The architectural students collect a vast amount of diverse and rich data, which they then use in their design projects.

The data that the architectural students collect comes in many different forms. For example, the student(s) may record a conversation between themselves and a resident of the area, which leads to the output of this data collection method being a voice recording file. Photographs they have taken of the area shown in Figure 9 and coordinates of certain places they deem relevant to their research. Sketches are another example (see Figure 10). Architectural students draw sketches of the area they are studying which they then use to create a context of the area. Other pieces of information include the notes that they had handwritten in their journals, which they may scan from time to time (see Figure 11). The architectural students then supplement the data they have collected with data from official sources such as the City of Tshwane (CoT), such as contour lines and building footprints. Architectural students wish to link the sketches they have drawn of an area to a particular set of coordinates, but they have no means to do so. Some of the data that the architectural students collect can be sensitive in nature, which therefore means that only the architectural student(s) that collected the data should be able to view and access it.



Figure 9: Data Example: Photograph of the area being mapped



Figure 10: Data Example: Sketch of the area being mapped

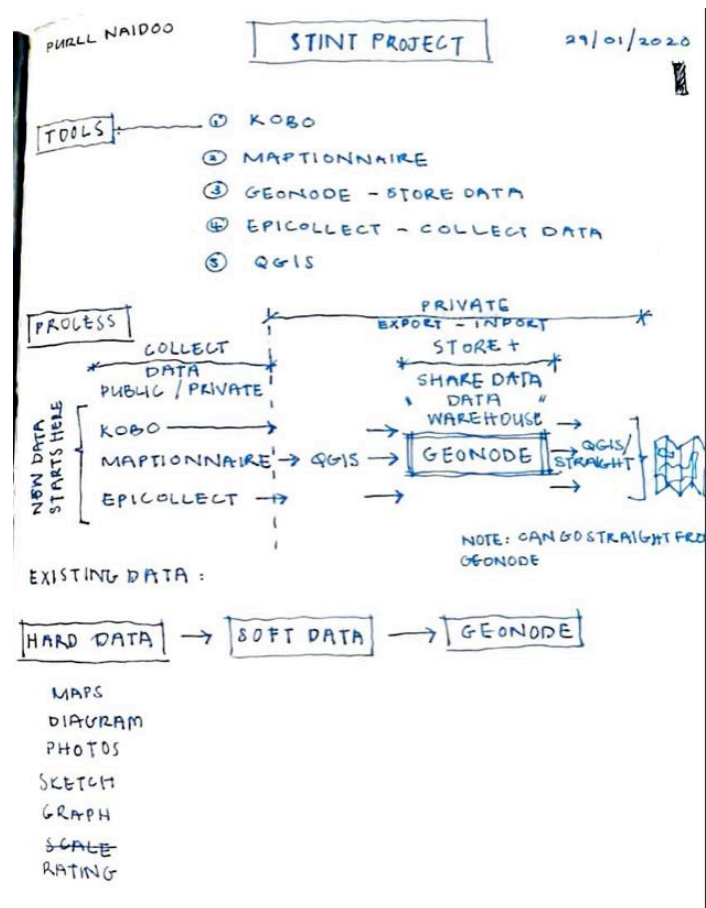


Figure 11: Data Example: Scanned copy of notes from a notebook

The architectural students then use this diverse and potentially rich data to enhance their design project by producing maps that present a social context, such as safe communal places for people to use. During an architectural studio, the students store and save their data on

their desktop PC or on Google Drive, which they share with other students. However, once the students have completed their design projects, the data is no longer maintained and often deleted. Not only is this way of working with data a challenge for organising, managing and sharing among students, the data is also lost for any future studies.

The data that does get saved, is saved poorly. The data is either incorrectly or poorly labelled, misplaced or sometimes, incomplete. The lecturers of the design studios attempt to save as much of this data as possible. Unfortunately, more of the analogue output is saved rather than the raw data. Such an example can be seen in Figure 12 and Figure 13. Other pieces of data and information such as flash drives with digital data like voice recordings and photographs, as well as analogue data such as sketches and notes get put into a folder, which is then put in the same cupboard. The lecturers of the design studios have attempted to at least organise the data into groups based on the year the data was collected and used.



Figure 13: Data Example: Data being categorised based on years



Figure 12: Data Example: Maps rolled up in the back of a cupboard

The following year, when a new group of architectural students are working in the same area and wish to centre their design project around a similar social context that has already been done, the architectural student(s) wish to review all the existing data in this area. The lecturer and students know that it has been previously mapped, and data was collected, so they wish to use this data to understand a change of social context over time. However, the lecturer

cannot remember who collected the data, or in what year the data was collected. This means that they need to search through all the folders and rolled up maps to find it. It may occur that the data has been corrupted, or lost over time. This leads to the data having to be collected again, which is a time-consuming process.

There is no efficient way to search through the analogue data and piles of paper. For any data that they do find, it may not be known who collected the data, why they collected the data, what purpose the data is supposed to serve, how they collected the data, and when they collected the data. Knowing this metadata can enhance the data and actually help determine whether the data is fit for use. It is worth noting that the data is usually represented in a more visual way and it is not necessarily a true representation as they tend to alter the raw data in Photoshop. The new architectural students then opt not to use the older data as they feel it doesn't suit their needs. They then collect new data and store it in the same way which then continues the cycle of incorrectly stored, managed and duplicated data.

3.3. Process to get to the requirements

The requirements for this solution were collected through a series of workshops in Sweden and South Africa during 2018 and 2019. During a 2018 workshop, the foundation was laid with regards to how the studios are run, how the data that is collected, as well as the process used to collect the data, and it was clear from this that a solution is needed that suited both a South African and Swedish context.

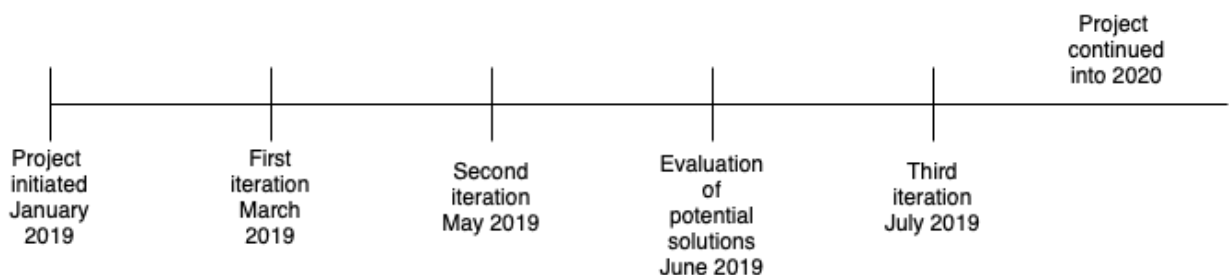


Figure 14: Timeline indicating the iteration and evaluation process to get the requirements

Figure 14 shows the iteration and evaluation process to get the requirements. The first iteration of identifying the requirements was performed followed by the second iteration. Once the second iteration was deemed to have detailed information, the evaluation of two potential solutions was made. After the evaluation of two potential solutions was complete, the third and final iteration of requirements was completed as well. With the evaluation and the requirements complete, the project then continued.

During the workshop in early 2019 held in Hammarkullen, Sweden, the first draft of the requirements that needed to be met were drawn up. The requirements were, rather, written as notes to understand the process that the students follow, and what the lecturers of the studios consider their requirements to be. The author of this dissertation was not present at this workshop, but did attend a virtual meeting during the workshop. The notes were grouped into categories that the lecturers identified as important. There were jotted down ideas that were later used to guide the requirements process.

Table 5: Category One from the notes that were drafted in order to understand the process

Uploading Data	Data is uploaded to the cloud, a server.
	Upload spatial data such as points, lines and polygons.
	Upload non-spatial data such as notes and images.
	Organize data into folder/groups.
	Upload data collected in the field.

The ‘requirements’ in Table 5 were taken from the notes that were drawn up from the early workshop in 2019 and only indicate a small portion of the actual first draft of the requirements. The complete full list of the requirements can be found in Appendix B. The notes that are presented in Table 5 were all grouped under the category of ‘Uploading Data’ as they all applied to the uploading of the data. The lecturers spoke about their need to store and manage the data on a cloud rather than on a local disk, which can be easily lost, and how they can upload any data that was collected in the field. The requirements indicate to a solution that is able to handle both spatial and non-spatial data uploaded. This data should be able to be uploaded into folders or groups for privacy concerns as well as for better organization.

Based on the needs listed in the first iteration, the first formal draft of the functional requirements was written. The functional requirements were at first written in order of need. It was these requirements that were then used to create the next iteration of the functional requirements that are presented in Table 6.

Table 6: Evolution of Category One requirements from the second iteration of functional requirements

Functional Requirement	Description
Uploading of non-spatial data	The solution should be able to store various kinds of data formats such as voice notes, images, .pdf documents, text files that are non-spatial in nature (but can be linked to a location via the metadata) and uploaded by the user.
Uploading of spatial data	The solution should be able to store spatial data formats such as vector and raster files that are uploaded by the user.
Groups	The users of the system will be working in project groups and each member of the group will need access to the data of their group and not be allowed to view the data of the other groups.

Table 6 shows an updated set of the requirements that were presented in Table 5. From early on it was evident that the solution needed to be able to handle data that is spatial and non-spatial in nature. As this was identified as an important requirement by the research team, it was kept in the evolution of the requirements from the first iteration to the second iteration. In the iteration process, the requirements were refined and expanded upon such that they were more clear and self-explanatory. It was decided that breaking up the uploading category from the first iteration would be best, so that the requirements were clear and concise. The full set of complete requirements from the second iteration can be found in Appendix C.

The requirements were then later refined, regrouped, and redefined, with more clear and precise definitions in the third iteration. The final requirements were split into six distinctive categories that were derived from the notes that were based on the needs of the lecturers, as well as the second iteration of the functional requirements. The full set of final requirements can be found in Chapter 4 Prototype – Design and Implementation through sections 4.3.2 to 4.3.8. It can be seen that the notes were a basic outline, with no explanation, where the second iteration did however provide an explanation.

Table 7: Selected functional requirements from the final iteration of functional requirements

Functional Requirement	Description
Filter based on location in the solution	User can filter for the spatial and non-spatial data based on location.
Filter based on metadata in the solution	User can filter for the spatial and non-spatial data based on the existing metadata.

Search based on metadata in the solution	User can search for a spatial and non-spatial data based on the existing metadata.
Upload a non-spatial data	User uploads a non-spatial data (e.g. .pdf, .docx, .xlsx, .zip, .jpg, and .txt) and add metadata.
Upload a spatial data	User uploads a spatial data (i.e. vector as a shapefile and raster as GeoTiffs) and add metadata.
User group assignment	User can be assigned to groups or can assign themselves to groups.

A search refers to starting from a blank page and populating it with results that meet the criteria, with a filter that hides the results that do not fit or match the criteria. Search functionalities were identified as an important requirement in the first iteration, where, as the iterations went along, search functionalities evolved to become search based on metadata in the solution, which is described by allowing the user to search for a document based on existing metadata. Filtering the data was further broken down into filtering based on metadata in the solution, and filtering based on location in the solution. This can speed up the filtering process, and allow for more accurate matches. Groups became user group assignment, which means that users can assign themselves to groups or they can be assigned to a group. Table 7 shows how the same set of requirements that are from Table 5 and Table 6 evolved over the multiple iterations.

In the next section, Criteria for Evaluation, the evaluation criteria on which GeoNode and GeoNetwork were evaluated, are presented. The evaluation criteria is based on the preliminary requirements that are presented above. The final functional requirements are presented in 4.3. Prototype – Design and Implementation.

3.4. Criteria for Evaluation

Functional requirements describe how the system ought to react to particular inputs and how the system should behave in particular situations, basically, the functional requirements describe what the system ought to do. In some cases, the functional requirements may also state what the system should not do.

Non-functional requirements are the constraints on the services or functions that are offered by the solution. They include, and not limited to, timing constraints, constraints on the development process, constraints imposed by standards and ethics, as well as constraints

imposed by the client. Non-functional requirements often apply to the system as a whole, rather than to the individual system features or services. Non-functional requirements are not directly concerned with the specific system services delivered by the system to the end user (Pressman, 2005; Sommerville, 2011).

Before developing the non-functional requirements, the principles of FAIR (findability, accessibility, interoperability, and reusability) data were read up on and understood so they could be incorporated into the non-functional requirements to make sure the solution complies to the FAIR data policy. More can be read about the FAIR data policy in 2.12.

Based on Table 8 and Table 9, the evaluation of GeoNode and GeoNetwork was undertaken, as presented in the next section (3.5). This evaluation was used to determine whether GeoNode or GeoNetwork would be used as a prototype for the potential solution and then the final formal functional requirements were also drawn up based on Table 9. In Table 8 and Table 9, a user is classified as the person that would be using the solution such as the architectural students and the administrator refers to architectural lecturers and the Information Technology (IT) Department that is within the Department of Architecture that will be responsible for the maintenance of the solution.

Table 8: Functional requirements on which the evaluation was done

Functional Requirement	Description
F1. Data Storage of non-spatial data	The system should be able to store various kinds of data formats, such as voice notes, images, .pdf documents, text files that are non-geographic in nature (but can be linked to a location via the metadata) and uploaded by the students.
F2. Data Storage of spatial data	The system should be able to store spatial data formats such as vector and raster files that are uploaded by the students.
F3. Search Functionalities	The system will allow users such as the students and lecturers to search the metadata of the data for specific attributes such as date collected, who collected it, what extent it was collected at and any other attributes that were collected.
F4. Filtering the data	Users must be able to filter the data according to an attribute of their choosing, i.e. the user wants to view all the photos that are stored in the system and then they can use the search feature to filter the data even further.
F5. Map Production	The users of the system must be able to export the data of their choosing into a simple map that can be saved as a PDF document for later photoshopping.
F6. Groups	The users of the system will be working in project groups and each member of the group will need access to the data of their group and not be allowed to view the

Functional Requirement	Description
	data of the other groups.
F7. Login Types	The users of the system should have different rights, roles and privileges. Students can only view, add data and have access to their group's data, lecturers can only view, add data and have access to all the groups data while the system administrator has rights to add, view and delete all the data that is stored in the system, they maintain and monitor the system.
F8. Viewing	The users of the system should be able to view the data in a simple manner with an OSM base map.
F9. Metadata	The users of the system must be able to add metadata to the system. Metadata that can be added include, who collected the data, what purpose the data was collected for, at what scale the data was collected and so forth.

Table 9: Non-functional requirements on which the evaluation was done

Non-functional Requirement	Description
NF1. Accessibility	A user of the solution ought to be able to easily access the solution online.
NF2. Data Integrity	The solution should be able to maintain the integrity of the data that is stored within the solution. In maintaining the integrity of the data, the solution should perform automated checks to ensure no corrupt data is uploaded to the solution.
NF3. Documentation of code	Clear and precise documents detailing the development process and the code that was used to develop the solution so that it can be open-source and understood by system admins. This is in case once the initial researchers and system engineers leave, the successors can understand the solution and be aware how to fix any issues that relate to the code.
NF4. Ease of Use	The solution should be easy to use for users, especially those users that have a non-GIS background. The functions and capabilities will not be GIS focused in a way that only working GIS professionals can use the solution. This is also related to the User Interface (UI) as it should not be complicated and easy to understand.
NF5. Interoperability	The solution needs to cross-platform in order to render correctly on all operating systems and browsers so that the users of the solution will be able to use the solution to its full effect. The solution needs to use OGC web services in order to correctly render and share spatial data.
NF6. Longevity	Systems longevity for an information system such as this one can be defined as the objective that is met if it is possible to manage the system over an extended period of time, so that the system continues to meet its intended purpose for as long as possible, even when the initial research project is complete. The

Non-functional Requirement	Description
	increasing availability of online resources means that data need to be created with longevity in mind. For this project, it is important to keep the system running even when the project funds have run out and the research team moves onto other projects. Both the potential solutions evaluated in The Assessment of GeoNode and GeoNetwork are supported by OSGeo, which has a strong developer and user base.
NF7. Online Capabilities	Online capabilities, such as the cloud, allow for working partners from different countries to collaborate.
NF8. Privacy	Data privacy is important, as personal questions with personal data will be collected by the users. The personal data that is collected should not be visible to those who do not or should not have access to it. User privacy is also important, the users must have the option to hide or display any personal data that they wish to. Privacy relates to being able to hide sensitive data from users who are not allowed to view it.
NF9. Reliability	The solution should experience as little down time as possible while keeping the rate of failure as low as possible as the architects will be collecting data at random times throughout the years.
NF10. Reusability	The solution should be able to be used multiple times in different use cases. The solution should also be able to be used over multiple years.
NF11. Robustness	Robustness refers to the time for the solution to restart after a failure and that probability of data corruption is low. If the solution were to crash or needed to be rebooted, the data should not be corrupted.
NF12. Scalability	Scalability refers to the ability of the solution to process or handle the growing number of requests or users, its potential to accommodate the growth. As the number of users that are going to use the solution vary from year to year, it is important that the solution is able to handle such a varying amount of users.
NF13. Security	The solution should be secure and safe as personal data and information will be collected and stored. If personal data leaks, this can cause ethical issues for the researchers as the users can sometimes collect sensitive data. Security relates to keeping the sensitive data secure.
NF14. Speed	The solution should be fast. The processed transactions per second should be high, the user/event response time should be as little as possible. Again, there might be a large number of users using the system at one time therefore an acceptable response time is essential.
NF15. User Documentation	Clear and precise documents on the correct management, use and maintenance of the solution. It is important to have a clear guide on how to use and maintain the solution. This can assist the long-term acceptance of the solution.

3.5. The Assessment of GeoNode and GeoNetwork

Based on the requirements that were presented in the previous section, two open-source solutions were reviewed. These were GeoNetwork (<https://geonetwork-opensource.org>) and GeoNode (<http://geonode.org>). As this research was a continuation of the research presented by Haynes, Coetzee and Rautenbach (2019), they identified and only considered open source tools as other tools would affect the manner in which the applications may be used and also how any derived data may be distributed.

3.5.1. GeoNetwork and its architecture

GeoNetwork is a catalogue application to manage spatially referenced resources. It provides powerful metadata editing and search functions as well as an interactive web map viewer. GeoNetwork provides users with an easy to use web like interface to allow them to search for geospatial data across multiple catalogues. The search function allows users to perform a full-text search as well as searches on keywords, resource types, and map scale. It has an interactive map viewer that is based on OpenLayers, which allows users to create, print, and share maps with others. Data within GeoNetwork can be described using the online metadata editing tools according to the ISO19115/119/110 standards. Based on user profiles such as reviewer or editor, a dashboard provides easy access to their information and tasks. Users can upload data in the forms of graphics, documents, PDF files, and spatial files. The administration console can provide access to the system configuration. Administrators can easily manage user and group accounts from this administration console.

As can be seen from Figure 15, GeoNetwork is a standard-based, free and open source catalogue application that is written in Java and it can be used as a server or desktop application. Its architecture is based on Service Oriented Architecture (SOA) and Geospatial portal reference architecture. It is comprised of four significant technologies; Java server pages (JSP), a database connection, Jeeves system and Lucene search (Ožana and Horáková, 2008).

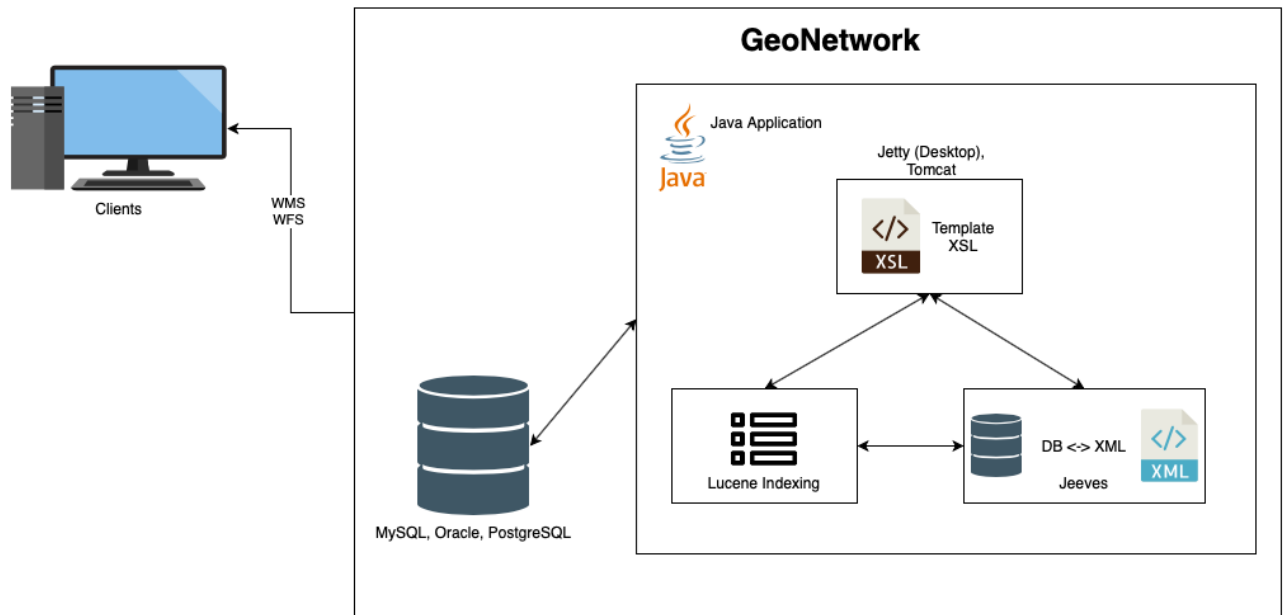


Figure 15: Simplified architecture of GeoNetwork (Ožana and Horáková, 2008)

3.5.2. GeoNode and its architecture

GeoNode is a geospatial content management system, which is a platform for the management and publication of geospatial and non-geospatial data. GeoNode uses an easy-to-use interface, which allows non-specialised users to share data and create interactive maps. It allows users to upload vector data, raster data, and other non-spatial documents. Once a user has finished uploading their data, the user can fill in the metadata in order to make it available via OGC protocols such as Web Map Service (WMS) and Web Feature Service (WFS). GeoNode lets users search for data geographically, or via a keyword search. Data can be edited and styled, and the metadata can be updated. Users have the option to associate documents with the spatial layers. GeoNode comes with user-friendly tools for styling and creating maps. These tools make it easy for users to assemble a web-based map that can be embedded in a web page or the map can be exported as a PDF document. For system administrators, GeoNode provides a customised version of the default Django Administration and Management Console, which provides users with an interface for viewing the entire set of modules and data that make up a GeoNode instance and allow administrators to perform bulk operations. The console can be used to monitor basic usage by the users and any user-provided content.

As described in Figure 16, GeoNode is a web application developed in Django, a Python web framework which controls a spatial data server such as GeoServer, a user interface, a spatial cache data server, a spatial database such as PostgreSQL and PostGIS, a catalogue such

as pycsw, or that can be altered to use GeoNetwork. Vector datasets are stored in the spatial database, while the raster datasets are stored in the file system. Map tiles are returned to the user using standards like WMS-C or WMS. Vector datasets are returned and can also be edited by the client using standards such as WFS and WFS-T (Corti *et al.*, 2019).

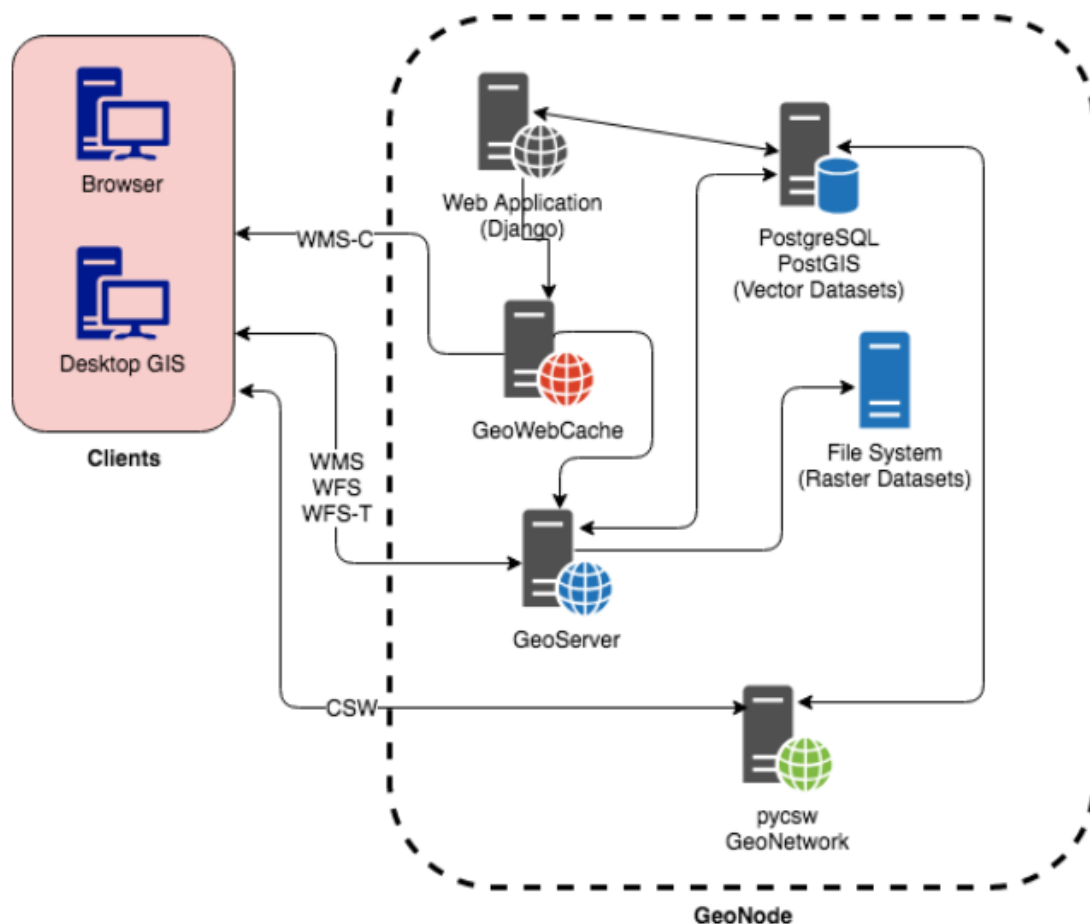


Figure 16: Architecture of GeoNode (Corti *et al.*, 2019)

3.5.3. Assessment of GeoNetwork against the requirements

F1. Data Storage of non-spatial data: As GeoNetwork is a catalogue of data resources rather than a repository, it does not store non-spatial data within GeoNetwork itself, but rather links to a GeoServer where the data is stored. In other instances, users can upload the URL to the data, such as a document, and then add the metadata. They can upload graphics, documents and PDF files.

F2. Data Storage of spatial data: As GeoNetwork is a catalogue of data rather than a repository it, does not store spatial data within GeoNetwork itself, but rather links to a GeoServer or other repositories where the data is stored. If users wish to store spatial data, they need to upload it to GeoServer first and then it is linked to GeoNetwork from there.

F3. Search Functionalities: GeoNetwork allows users to perform a full text search, a group search (a search based on user groups within GeoNetwork) and spatial filtering to choose data in specific geographic areas as well as an advanced search.

F4. Filtering the data: Data can be filtered on its spatial extent meaning that results that do not match the filter are hidden.

F5. Map Production: GeoNetwork can be used as a mapping application to create maps that can be printed or shared with others. These maps can also be saved in GeoNetwork as data themselves.

F6. Groups: GeoNetwork has the ability to create groups. A user can be part of one or more groups, and these groups can then have access to the data to which they are permitted access.

F7. Login Types: There are five login types or user roles within GeoNetwork, viz.: administrator profile; user administrator profile; content reviewer profile; editor profile; and registered user profile. The administrator has special privileges that give access to all available functions, where the user administrator is only the administrator of their own group; the content reviewer is the only person allowed to give final clearance on the metadata publication; the editor works on metadata; and the registered user has more access privileges than non-authenticated guest users.

F8. Viewing: The map viewer provides a way for users to interactively assess the relevance of a resource discovered in GeoNetwork. Data is not automatically added to a base map; the user is required to add the data to a base map to visualise it.

F9. Metadata: When using metadata to describe data within GeoNetwork, the user first needs to choose a standard to use such as Dublin core, ISO 19115/119/139 and ISO 19110. Metadata can also be edited or updated once it has been added and then downloaded should the user wish to do so.

NF1. Accessibility: Depending on a user's status within GeoNetwork, they have the ability to access and discover data within GeoNetwork as they can access GeoNetwork via an internet-enabled device.

NF2. Data Integrity: GeoNetwork can perform automated processes to check the integrity of the data thus ensuring that no data integrity is lost.

NF3. Documentation of code: The code of GeoNetwork is well documented on GitHub (<http://geonetwork-opensource.org/>). There is existing documentation on the method for installing GeoNetwork.

NF4. Ease of Use: GeoNetwork is often used in Spatial Data Infrastructures (SDI) across the world such as The Webservice-Energy.org Global Atlas (<http://geocatalog.webservice-energy.org/geonetwork/srv/eng/main.home>) for solar and wind energy data. The interface is similar to that of a web interface but finding and viewing data can be complicated for non-technical users.

NF5. Interoperability: GeoNetwork is a cross-platform application that renders correctly on all operating systems and browsers. GeoNetwork also uses OGC web services such as Web Feature Service (WFS) and Web Map Tile Service (WMTS) to correctly render and share spatial data as well as support ISO19115/119/110 standards that are used for spatial metadata standards.

NF6. Longevity: As there is currently a stable version of GeoNetwork ready for deployment, GeoNetwork will be stable which aids in its longevity. GeoNetwork is an OSGeo project, which has a strong developer and user base.

NF7. Online Capabilities: GeoNetwork is an online platform that supports multi-person use as it allows users to collaborate with other professionals. GeoNetwork can work on a server which is preferred for this solution.

NF8. Privacy: GeoNetwork only hosts the metadata to the actual data so that allows users, who have the appropriate permissions, to view the metadata of the actual data, but it is the responsibility of the data host to impose restrictions on the viewing, use and downloading of the data.

NF9 Reliability: as there is currently a stable version of GeoNetwork ready for deployment, GeoNetwork will experience little down time when it is running which aids in its reliability. GeoNetwork is also maintained around the clock by a team of volunteers as it is an OSGeo project.

NF10. Reusability: as an instance of GeoNetwork can be set up once and as long as that instance is operational, users can reuse the same instance over multiple years and in different use cases.

NF11. Robustness: as GeoNetwork is stable, it is highly robust, and downtime between reboots is minimal.

NF12. Scalability: as GeoNetwork can be accessed via an internet-enabled device and can be hosted on a powerful server, it is possible to adjust to the number of users using it at.

NF13. Security: the server GeoNetwork is installed on can be encrypted making it secure and user access to the data can be restricted.

NF14. Speed: the more powerful server that is used to host GeoNetwork, the quicker it will be for processing any requests.

NF15. User Documentation: The documentation of use for GeoNetwork is publicly available under the GeoNetwork Users Guide (<https://geonetwork-opensource.org/manuals/trunk/en/user-guide/index.html>). The documentation is well maintained and up to date with the versions of GeoNetwork.

3.5.4. Assessment of GeoNode against the requirements

F1. Data Storage of non-spatial data: GeoNode has the ability to store non-spatial data such as portable document format (PDF) files, MS Word documents, MS Excel sheets, images (.png and .jpeg), as well as ZIP files. This is done through a simple user-friendly interface. In GeoNode, non-spatial data is referred to as “Documents”.

F2. Data Storage of spatial data: GeoNode has the ability to store spatial data files such as shapefiles (.shp) and GeoTiffs (.Tiffs). This is done through a simple user-friendly interface and the data gets stored in a PostGIS database. In GeoNode, spatial data is referred to as “Layers”.

F3. Search Functionalities: GeoNode has a user-friendly search function that allows users to search through all the files using keywords, metadata, data owners, and spatial extent.

F4. Filtering the data: users of GeoNode have the option to filter data before and after a search for specific data. Once a search is completed in GeoNode, users have the ability to further filter their search based on other requirements such as keywords, owners of the data, and spatial extent.

F5. Map Production: using existing spatial layers that are stored in GeoNode, users have the ability to create a simple map with a title, legend, and scale bar. This map can then be exported as a PDF document for use within the field. These maps can also be saved in GeoNode as maps.

F6. Groups: GeoNode has the capability of assigning users to groups and then assigning rights and privileges to those groups to allow them to view and edit data based on their rights. A user can be part of one or more groups.

F7. Login Types: GeoNode in a sense has four types of users, viz.: an anonymous user; a registered user; a staff user; and an administrative user. An anonymous user and a registered user can only search, view, and download data permitted, a staff user can designate whether the user can log into this admin site and has the same ability to upload search, view, and download data permitted, and an administrative user has access to the admin site and can control the entire GeoNode as well as having the same ability to upload search, view, and download data permitted.

F8. Viewing: any layers that are stored in GeoNode can be viewed by those who are permitted to do so, on an interactive OpenStreetMap base map. Panning, zooming, and point query functionalities are also available. Users can only view documents that they are permitted to view by having to download the document first.

F9. Metadata: when a layer or document is uploaded to GeoNode, users are given the opportunity to add metadata to the layer or document they are uploading. Once a layer or document is uploaded, users have the ability to edit the metadata or update it.

NF1. Accessibility: depending on a user's status within GeoNode, they have the ability to access and discover data within GeoNode as they can access GeoNode via an internet-enabled device.

NF2. Data Integrity: GeoNode performs automated checks on the spatial data that is being uploaded. For example, if a user is uploading a shapefile and a supporting file such as the

projection metadata (.prj) file is missing, GeoNode will alert the user and prevent the shapefile from being uploaded.

NF3. Documentation of code: GeoNode's code is documented on GitHub (<https://github.com/GeoNode/geonode>) with clear instructions on the correct use and method of installation for the software.

NF4. Ease of Use: GeoNode is designed so that non-technical users can use GeoNode to the fullest of its capabilities. It has a user-friendly interface with step-by-step instructions along the way.

NF5. Interoperability: GeoNode is a cross-platform application that renders correctly on all operating systems and browsers. GeoNode also uses OGC web services such as Web Feature Service (WFS) and Web Map Service (WMS) to correctly render and share spatial data as well as support ISO19115/19139:2007 standards that are used for spatial metadata standards.

NF6. Longevity: as there is currently a stable version of GeoNode ready for deployment, GeoNode will be stable, which aids in its longevity GeoNode is an OSGeo project which has a strong developer and user base.

NF7. Online Capabilities: GeoNode is an online platform that supports multi-person use as it allows users to collaborate with other professionals. GeoNode can work on a server, which is preferred for this solution.

NF8. Privacy: as personal data is collected by the users; it is important that the data that is stored in GeoNode is secure and private. Certain data can be restricted from being viewed and downloaded by other users. The server GeoNode is installed on can be encrypted making it secure and user access to the data can be restricted. Users can hide or display any personal data that is stored in their profile.

NF9. Reliability: as there is currently a stable version of GeoNode ready for deployment, GeoNode may be considered stable, experience less down time which therefore aids in the reliability of it. GeoNode is also maintained around the clock by a team of volunteers as it is an OSGeo project.

NF10. Reusability: as an instance of GeoNode can be set up once and as long as that instance is operational, users can reuse the same instance over multiple years and in different use cases.

NF11. Robustness: as long as GeoNode is installed on a stable server, it is very robust and downtime between reboots is minimal. If a Docker install was performed, there is a separate Docker container with the data in a PostgreSQL database which can be easily backed up if need be.

NF12. Scalability: as GeoNode can be accessed via an internet-enabled device and can be hosted on a powerful server, it has the ability to adjust to the number of users using it at any given time as one can adjust the hardware according to the expected amount of parallel users.

NF13. Security: the server GeoNode is installed on can be encrypted making it secure and user access to the data can be restricted.

NF14. Speed: the more powerful the server that is used to host GeoNode as well as the network speed and connection of the user, the quicker it will be for uploading layers and documents and processing requests.

NF15. User Documentation: the documentation of use for GeoNode is publicly available under the GeoNode User's Guide (<https://docs.geonode.org/en/2.10/usage/index.html>). The documentation is well kept and up to date with the versions of GeoNode.

3.6. Summary of Evaluation

Based on the evaluations above, it was decided that GeoNode was the most suitable solution that met the requirements by providing an integrated solution for data hosting and management. GeoNode allows users to view, manage, edit the raw data as well as the metadata and share the data with other users while at the same time, offering a layer of security and privacy for the users. The extensive existing documentation also counted in its favour, but more of this will be discussed in later chapters. GeoNetwork lacks the ability to edit spatial data files.

The final evaluation came down to the aim of the solution. GeoNetwork is a catalogue of data and the purpose of it is not to load data, but to rather index the data and GeoNode is a repository because users can store, manage, share, edit, and download data. This was the reason GeoNode was selected over GeoNetwork even though they in theory both have the

same functions and particularly similar capabilities, but the aim of each application differs. The aim of this solution was to store, manage, share, edit, and download data, which is what GeoNode is better suited for.

For this project, a repository is better suited to the needs of the architects, and therefore, GeoNode is the tool this project used.

Chapter 4 – Design and Implementation of the Prototype

4.1. Chapter Overview

This chapter focused on the design and the implementation of the prototyped solution. The solution was designed to meet the requirements both functional and non-functional in nature. This chapter describes the process that was followed to design and implement the prototyped solution and its associated documentation. The prototyped solution is based on a GeoNode instance which was configured to meet the requirements. Some screenshots of the prototyped solution are included in this chapter to demonstrate the capabilities of the prototyped solution.

4.2. How GeoNode was installed

4.2.1. Minimum System Requirements

Once the appropriate server was selected and set up to meet all the system requirements for GeoNode to operate successfully, the GeoNode prototype was installed. The system requirements for the server were:

- Ubuntu or CentOS operating system version 18.04 or later
- 8 Gigabyte (GB) of Random-Access Memory (RAM)
- 2.2GHz processor with two cores
- 10 Gigabyte (GB) software disk usage
- Additional disk space for data storage (50 Gigabyte (GB) or more)
- 64-bit hardware

4.2.2. Installation Process

At the beginning of the project, the GeoNode installation process was unfamiliar and new, which made the process difficult, where it took longer than it should have. When the installation process was started, version 2.10 of GeoNode was out and the documentation was outdated. Once the GeoNode prototype was running in debug mode (see Run GeoNode in Debug mode for more information), the researcher attended a GeoNode workshop at the FOSS4G conference. The workshop was hosted by the GeoNode lead developer. Attending this workshop was beneficial as some hints and tips were gained that helped produce a functional GeoNode. Once the documentation was updated, the installation process became easier from this point onwards, and the installation process was completed. A recommendation to overcome this barrier is to follow the GeoNode Docker installation instructions. These instructions are less complicated, require less understanding, and as the GeoNode image is already built, it requires less time to get GeoNode up and running. This is a more less-technical user friendly approach. The documentation can be found here:

<https://docs.geonode.org/en/2.10/usage/index.html>. In older versions of GeoNode such as 2.0x, users had option to install GeoNode using the command “apt-get install geonode” but that was disabled in newer versions of GeoNode such as 2.10 which was the one installed for this research.

4.2.2.1. Install the dependencies

Once the Ubuntu server was set up that met the requirements needed, the dependencies needed to be installed first. To do that, the system packages were upgraded, packages such as Python 2.7 (<https://www.python.org/download/releases/2.7>), Git (<https://git-scm.com/downloads>), SQLite3 (<https://www.sqlite.org/download.html>), Geospatial Data Abstraction Library (GDAL, <https://gdal.org/download.html>), OpenJDK (<https://openjdk.java.net/install/index.html>) were installed, and a dedicated super user, GeoNode, was created that would run installation commands in the pseudo-group. All these commands were run using nano and not VIM. As Python 2.7 is depreciated, the source code of GeoNode has been updated to include Python 3.0 so as to account for this.

4.2.2.2. GeoNode Installation in Debug mode

The next step in the GeoNode installation process was to install GeoNode in its most basic form (the same as running it in Debug mode), where it does not utilise any external servers like Apache Tomcat (<https://tomcat.apache.org/download-80.cgi>), PostgreSQL (<https://www.postgresql.org/download>), or HTTPD (<https://httpd.apache.org/download.cgi#apache24>), and it runs locally against a filesystem-based SQLite database. The first thing that happened is a Python Virtual Environment was prepared, because all the commands from here on were run in this Virtual Environment. Once the Virtual Environment was active, the GeoNode core base folder was created and GeoNode Core was cloned into this folder from GeoNode’s GitHub page (<https://github.com/GeoNode/geonode>). After the GeoNode Core was successfully cloned, some of the settings were configured or changed at this stage of the installation process. Before GeoNode could be run in Debug or Development mode for the first time, the site host name had to be changed in the settings.py file. As it stood, the only allowed site host name was localhost and because the GeoNode instance was being run and installed on a public server, the site host name had to be changed to the public URL that was being used. This public URL was <http://geocatalogue.co.za>. If this setting was not changed, then GeoNode would not be accessible. The next setting that needed to be configured was the email setting. As the email enabled setting was set to be true (‘EMAIL_ENABLE’, ‘True’), the email settings needed to be configured. The Django email host was changed to ‘smtp.gmail.com’, the Django

email port was set to '587', the Django email host user was changed to gistuks@gmail.com, the Django email host password were added, the Django email use TLS setting was set to 'True', and the default from email was changed to cameron.green@tuks.co.za. These changes were then saved and GeoNode could now be run in Debug mode.

4.2.2.3. Run GeoNode in Debug mode

Running GeoNode in Debug mode for the first time runs both, GeoNode and GeoServer, locally after having prepared the SQLite database. To prepare the SQLite database, paver was used. Paver is a python-based software project scripting tool. It is designed to handle the dependency tracking requirements of a programme and help with the repetitive tasks such as the downloading of features.

- The GeoNode SQLite database had to be first setup with the command 'paver setup', then it had to be synced using 'paver sync' and finally it was started using the command 'paver start'.
- Once the server had finished the initialisation of GeoNode, the console displayed 'GeoNode is now available', where GeoNode could be opened up ran on <http://geocatalogue.co.za:8000/> and GeoServer ran on <http://geocatalogue.co.za:8000/geoserver/>. This set up is only beneficial when run in Debug mode, but it is not suitable for production as GeoNode and GeoServer are currently in debug mode.
- GeoNode was now running locally and users could sign in using the username 'admin' and password 'admin'.
- If GeoNode was not running locally on <http://geocatalogue.co.za:8000/> and the error 'Site Host Not Allowed' appeared, this meant the changes that were made were not reflected and needed to be reviewed again.
- A test user was then required to register to GeoNode so as verify whether the email settings were also changed correctly.
- If both parties received emails regarding the test user registering, then the settings were changed correctly. If not, and the registration page did not load or the email host got an email saying that it was unable to send the email to the test user, the settings had to be reviewed again.
- Once GeoNode passed both these tests, the next step of the installation process, the PostGIS database setup could be completed.

4.2.2.4. PostGIS database setup

System administrators can choose to use their own database system, but the default and recommended database system is PostgreSQL with the PostGIS extension. To install and configure the PostgreSQL database system, two databases, viz. geonode and geonode_data, were created, and both databases belong to the superuser geonode. This is the default configuration, but system administrators can use any database or role they wish to use but the connection parameters need to be correctly configured. Next, the databases and permissions needed to be configured. First, the geonode user had to be created and GeoNode was going to use this user to access the database. Once the two databases were created with the owner, geonode, the PostGIS extensions were created. The final step was to change the user access policies for the local connections from 'peer' to 'trust'. This changes the security settings of the PostgreSQL database to allow any connections from outside the PostgreSQL server. Local is for Unix domain socket connections only. Once the user access policies were changed, the PostgreSQL service had to be restarted to make the changes effective.

4.2.2.5. Install GeoServer

Installing GeoServer was the next step in the installation process. When the 'paver start' command was run, the script ran a Jetty Servlet Java container GeoServer with its default settings. In order to execute the next steps in the installation process, GeoNode and GeoServer paver services had to be stopped, which is done by running the command 'paver stop'. Running GeoServer in the Jetty Servlet Java container is not the optimal way to run GeoServer, as it is a fundamental component of GeoNode and GeoServer needs to be running in a stable and reliable manner. In order to ensure that GeoServer was run in a stable and reliable manner, an Apache Tomcat 8 Servlet Java container was installed, which was started by default on the internal port '8080'. Several operations to optimise GeoServer were run after the Apache Tomcat 8 Servlet Java container was installed. The first optimisation was to correctly setup the Java VM Options, like the available heap memory and the garbage collector options. The second optimisation was to externalise the GeoServer and GeoWebcache catalogues in order to allow for further updates without the risk of deleting any existing datasets. It is at this point that if the server has not enough RAM or more RAM allocated to it, you can change the RAM assigned to GeoServer and GeoNode. The default options reserve 4GB of RAM to GeoServer and 2GB of RAM for GeoNode. If there is not enough RAM available, the values can be lowered but the performances of the services will be impacted. In this case, 4GB of RAM was allocated to GeoServer and 4GB of RAM were allocated to GeoNode. Once all the changes have been made, the Servlet Container needs to be restarted in order to take effect. The logs should appear without any errors, but if errors do appear, system administrators should check the geoserver.log file to try and gain an understanding of

what went wrong. GeoServer was now up and running at <http://geocatalogue.co.za:8000/geoserver/>. The new running GeoServer was tested with the GeoNode paver service in Debug mode. The paver reset command no longer cleans up GeoServer meaning that any data in GeoServer will remain in GeoServer even if GeoNode was reset.

4.2.2.6. Web Server

Up until now, GeoNode has been run in Debug mode from the command line, using the paver utilities which is not the best way to run GeoNode in a production environment. The first step in moving GeoNode over to being ready to be deployed to production was to install and configure NGINX. Before this was done, the GeoNode paver services were stopped. Once stopped, the NGINX services were installed. Next, GeoNode and GeoServer were served using NGINX. To make this happen, some GeoNode environment variables were changed. The site host name, Django email host user, Django email host password, Django email host, Django email port, Django email use TLS, and default >"From:"< email environments were all changed to match the environments set in GeoNode Installation. Once this was done, the GeoNode NGINX configuration had to be enabled and then the services were restarted. GeoNode's static data and GeoServer's OAuth2 settings were refreshed. Once these steps were successfully completed, GeoNode ran on <http://geocatalogue.co.za/>, and GeoServer ran on <http://geocatalogue.co.za/geoserver/>. The next step in ensuring that GeoNode was ready for production was to update the settings in order to use the PostgreSQL database that was set up previously. To do that, the Tomcat services were stopped, GeoNode was initialized, the UWSGI settings were updated, geonode.settings was migrated to geonode.local_settings then the UWSGI settings were restarted and the OAuth2 settings were updated to now include the new geonode.local_settings. The last step in updating the settings in order to use the PostgreSQL database was to change the GeoNode IP address from '[localhost](#)' to '[geocatalogue.co.za](#)'.

Now that the GeoNode instance was being set up to be run on a public IP, the settings needed to be changed to reflect this. This is a four-step process. The first step is to update the NGINX configuration in order to serve the new domain name, geocatalogue.co.za. The server name had to be changed from '[localhost](#)' to '[geocatalogue.co.za](#)'. Next, the UWSGI configuration was updated in order to serve the new domain name, the hostname was changed from '[localhost](#)' to '[geocatalogue.co.za](#)'. The third step was to update the OAuth2 configuration in order to reflect the new hostname. GeoNode's IP address was changed from '[localhost](#)' to '[geocatalogue.co.za](#)'. The last and fourth step in the process of updating the settings to setup GeoNode to be run on a public server was to update the existing GeoNode links in order to

reflect the new hostname. Here, the GeoNode IP was also changed from 'localhost' to 'geocatalogue.co.za'. The final step was to install and enable a HTTPS secured connection through the Let's Encrypt provider for the GeoNode instance. To have done this, the Let's Encrypt Certbot had to be installed, NGINX config had to be reloaded and the firewall had to be configured to make sure it does not deny access to HTTP. Once the server was encrypted and secure, all the redirect URI's had to be updated accordingly so as to reflect this. The GeoNode OAuth2 redirect URI in the GeoNode Admin Dashboard had to be updated and changed from '<http://geocatalogue.co.za/>' to '<https://geocatalogue.co.za/>'. Next, the GeoServer Proxy Base URL was updated, and this was done in the GeoServer Admin GUI. Again, '<http://geocatalogue.co.za/geoserver/>' was changed to '<https://geocatalogue.co.za/geoserver/>'. While in the GeoServer Admin GUI, the GeoServer Role Base URL was changed from '<http://geocatalogue.co.za/>' to '<https://geocatalogue.co.za/>'. The second last step in this process was to update the GeoServer OAuth2 Service Parameters in the GeoServer Admin GUI. All the 'http://' connections had to be changed to 'https://'. The final step was to update the UWSGI configuration file by changing 'http://' connections to 'https://', three more environmental variables were added and finally the service was restarted.

To ensure that the GeoNode prototype would be accessible to all the students at the University of Pretoria and Chalmers University of Technology, the URL '<https://geocatalogue.co.za/>' had to be whitelisted on the University's network. Once the URL was whitelisted, the prototype was fully functional, working, and available for everyone to use. The prototype was now ready to be used in the Test Cases. The GeoNode prototype was now running in a stable and reliable manner that was ready for production.

The home page of the prototype (Figure 17) has all the links to all the features and capabilities of the prototype. From here, users can easily search for any content stored within the prototype and have an overview of the system as a whole as they are able to view the number of documents, layers, users and maps that are within the prototype. The name of the project, *Stitching the City: From Micro-views to Macro Data*, is visible to the users. This page is the first thing users see of the prototype, so it needs to be visually appealing and welcoming to first time users.

<https://geocatalogue.co.za/> will be available until the end of June 2021 and then will no longer be accessible via this URL. To access a working instance of a GeoNode instance, please refer to the GeoNode master demo instance, <https://master.demo.geonode.org>.

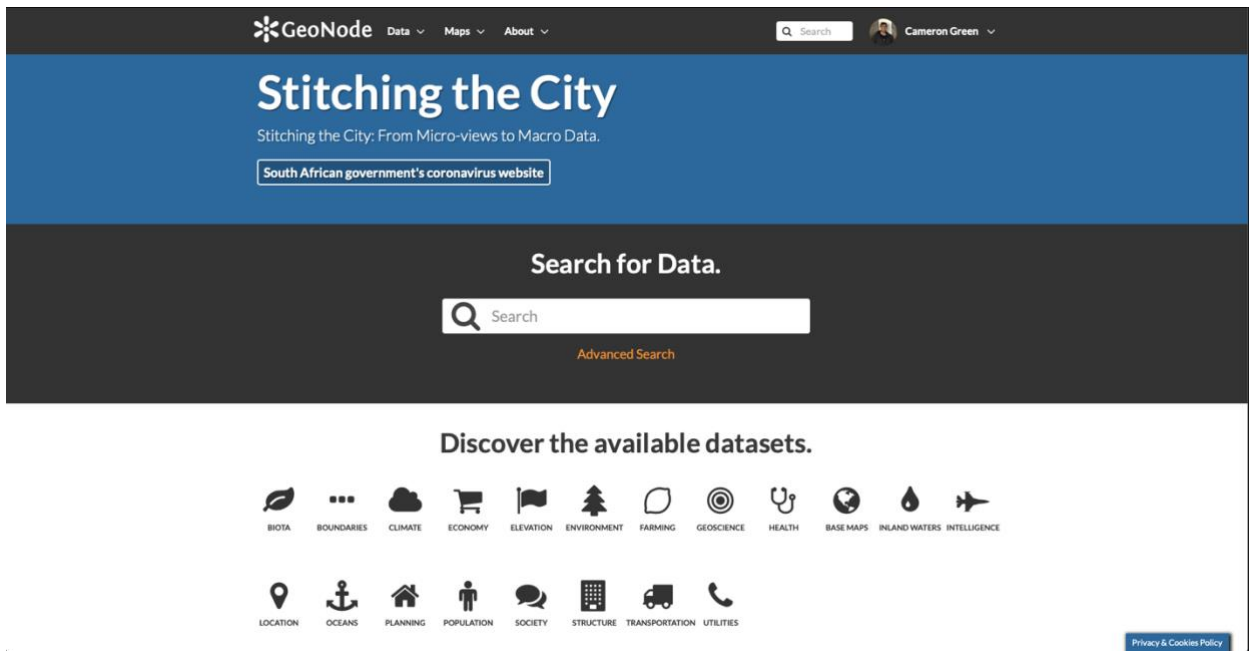


Figure 17: Home page of the Prototype

4.3. Prototype – Design and Implementation

4.3.1. Overview

Once GeoNode was identified as the suitable solution for the prototype, the prototype was then designed. The high-level functional requirements that were identified in the previous chapter are described and how GeoNode in specific meets these requirements, as well as the non-functional requirements which satisfy the implementation.

The final functional requirements of the prototype have been divided into six main categories, viz.: user administration; uploading data; viewing data; updating data; finding data; and making data available.

4.3.2. User administration in the prototype

GeoNode has the capabilities to handle user administration. Users need to be able to register to the prototype in order to have limited or full access and to be able to use the functionalities granted to them by their user status. In order to register, users need to have a valid and working email address, and a unique username and password associated to them. Within the user roles, two roles for the users exist, user and administrator. An administrator is a user with the administrator role. All registered users have the ability to upload and view spatial and non-spatial data files and modify or delete spatial and non-spatial data files that they own. Only administrators can change these permissions. Users will be able to assign themselves to a single group or multiple groups or the administrator can assign them a group or groups. If a non-administrative user creates a group within the solution, they become administrator of that group but not of the GeoNode instance. Members of this group can view data that their group is permitted to view and any other data within the solution that is publicly available. Only once a user has registered to the prototyped, they are able to upload and modify spatial and non-spatial data files.

FR 1: User registration

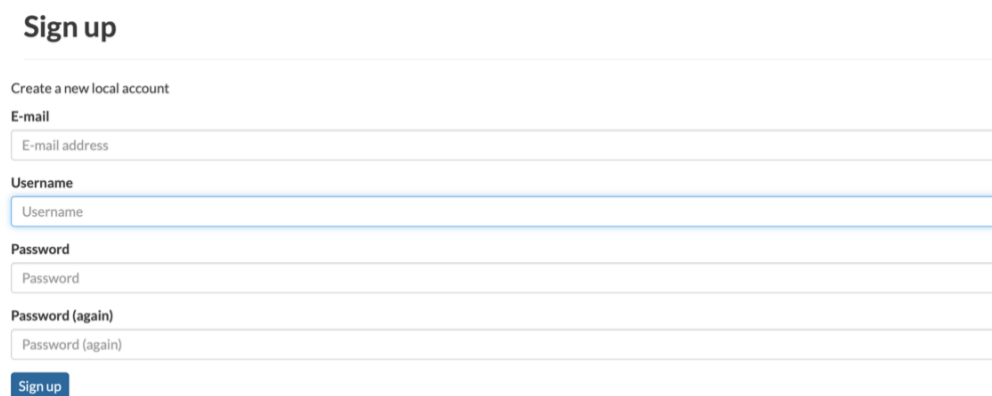
Requirement:

A user shall be able to register to the prototype using a username and password of their choice and their working email address.

In order for a user to register, they need to provide a valid and working email address. Once a user has been registered, the prototype records the username, password, and email address which the users will then use to log into the prototype.

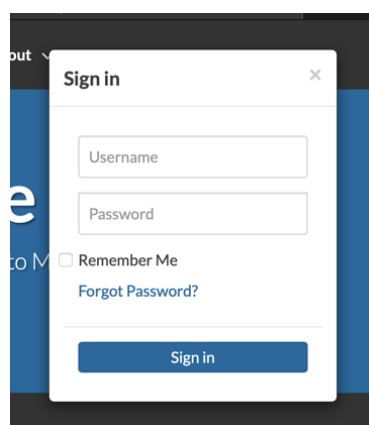
To be able to register to the prototype, users need to have a valid working email address, which they enter under the email address field. Users get to select their own unique and appropriate username and password that they will use to log into the prototype once their account has been approved. After the user details are filled in, users click submit and then the system administrator receives an email notification of the new account request which they can either approve or reject. After the administrator has approved the user, the user receives an email notification stating their account is now active and requests them to log in using their unique username and password.

To sign in, users are required to use their unique username and password that is assigned to them when registering. This step will only work once the account has been approved by the system administrator. Users have the option to reset their password in the event they have forgotten what their password was. Figure 18 and Figure 19 are the Sign up and Sign in pages, respectively. To sign up, the user enters their details and then to sign in, they only have to use their unique username.



The image shows a web form titled "Sign up" for creating a new local account. It includes four input fields: "E-mail address", "Username", "Password", and "Password (again)". A blue "Sign up" button is located at the bottom left of the form.

Figure 18: Registration page.



The image shows a mobile-style "Sign in" modal window. It contains input fields for "Username" and "Password", a "Remember Me" checkbox, a "Forgot Password?" link, and a blue "Sign in" button at the bottom.

Figure 19: Sign in option on homepage

FR 2: User authentication

Requirement:

The prototype shall be able to authenticate a user against the list of users in the user database and allow access.

Using their username and password, users can sign into the prototype. The prototype authenticates these details against the list of registered users in the system's database and access is either granted or denied.

On the Django Administration side, this tool provides the system administrators with the ability to edit all the users that are registered to the solution (Figure 20). Here, system administrators can search and filter for selected users. In the overview, the system administrators can see the username, email address, first and last name of each user, whether or not the account is active, and whether the user has staff access to the administrative console of the prototype. Again, system administrators can perform bulk operations, such as deleting selected or inactive users. System administrators can also manually add and approve a user to the prototype. This tool also provides system administrators with the ability to edit each user and the details related to each user. System administrators can also view which resources in the prototype the user owns, and when they last logged in to the solution.

The screenshot shows the Django Administration interface for the 'Users' section. At the top, there is a navigation bar with 'Django administration' on the left and 'WELCOME, CAMERON VIEW SITE / CHANGE PASSWORD / LOG OUT' on the right. Below the navigation bar, there is a breadcrumb trail: 'Home > People > Users'. The main content area is titled 'Select user to change' and features a search bar with a 'Search' button. Below the search bar, there is an 'Action:' dropdown menu with a 'Go' button and a selection count of '0 of 33 selected'. The main part of the interface is a table with columns for 'USERNAME', 'EMAIL ADDRESS', 'FIRST NAME', 'LAST NAME', 'STAFF STATUS', and 'ACTIVE'. The 'EMAIL ADDRESS' column contains redacted information. The 'STAFF STATUS' column has red circles, and the 'ACTIVE' column has green circles. To the right of the table is a 'FILTER' sidebar with sections for 'By staff status', 'By superuser status', 'By active', and 'By groups'. The 'By groups' section lists 'All', 'anonymous', 'registered-members', 'test-group-1', 'test-group-2', and '-'. At the top right of the main content area, there is an 'ADD USER +' button.

USERNAME	EMAIL ADDRESS	FIRST NAME	LAST NAME	STAFF STATUS	ACTIVE
<input type="checkbox"/> AimeeSimeon				○	●
<input type="checkbox"/> AnonymousUser				○	●
<input type="checkbox"/> ClydeFoelp				○	○
<input type="checkbox"/> CoreyFag				○	○
<input type="checkbox"/> DanielDum				○	○
<input type="checkbox"/> Davidsoda				○	○
<input type="checkbox"/> GeorgeOdoff				○	○
<input type="checkbox"/> Ju@18				○	●
<input type="checkbox"/> Martine				○	●
<input type="checkbox"/> Momedubois				○	●
<input type="checkbox"/> NinaJvR				○	●
<input type="checkbox"/> Patrickbrica				○	○
<input type="checkbox"/> Purlins				○	●
<input type="checkbox"/> Robertzep				○	○
<input type="checkbox"/> RobinEskilsson				○	●
<input type="checkbox"/> ...				○	○

Figure 20: People Users tool (email addresses have been redacted for privacy)

FR 3: User group assignment

Requirement:

The user shall be assigned to a group which allows for certain rights and privileges. Groups can be created in the prototype by registered users and then users can assign themselves to these groups or the prototype administrator can assign a user to a group.

On the Groups home page (Figure 21), registered users and system administrators can create groups and assign other registered users and system administrators to a group. An overview of each group is also provided such as who is the manager of the group and what permissions the group has in terms of interacting with layers and documents, how many registered users are in the group, as well as a description of the group. A thumbnail of their own choosing can also be set for each group by the group manager.

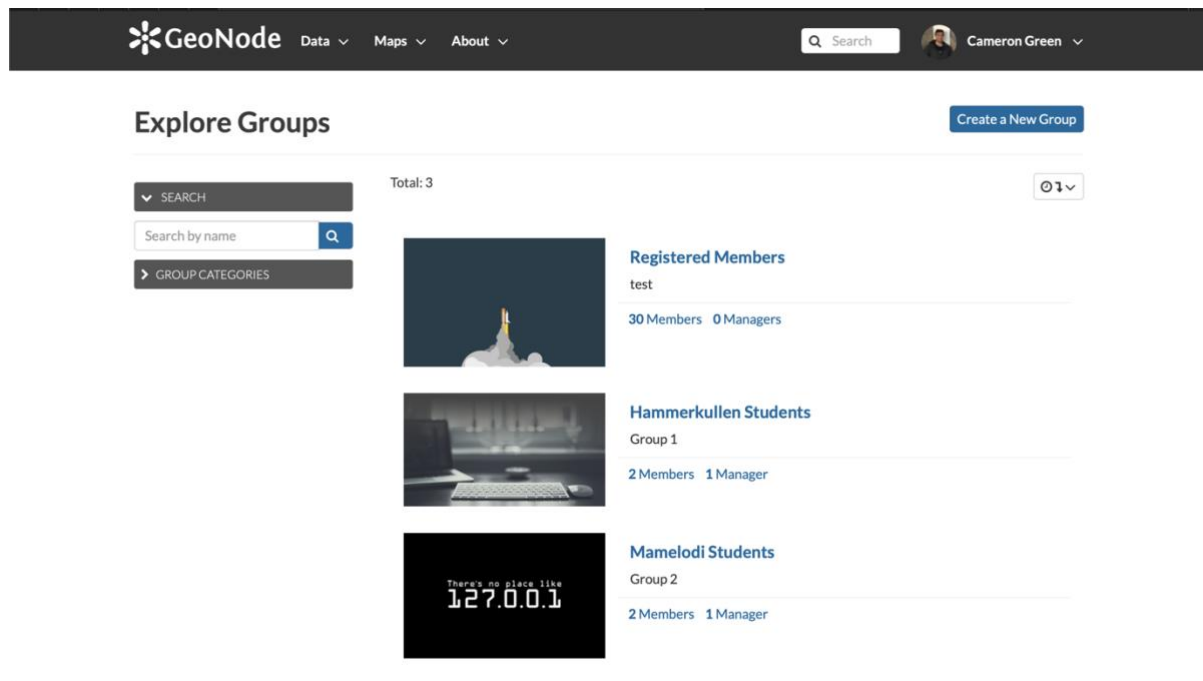


Figure 21: Explore Groups page.

The Authentication and Authorisation Group tool in Figure 22 provides system administrators with the ability to control the groups to a limited extent. System administrators can manually add a group here or edited already existing groups. When the system administrator decides to edit a group, they can only edit the group name and what permissions the group has within the prototype. System administrators cannot edit group members.

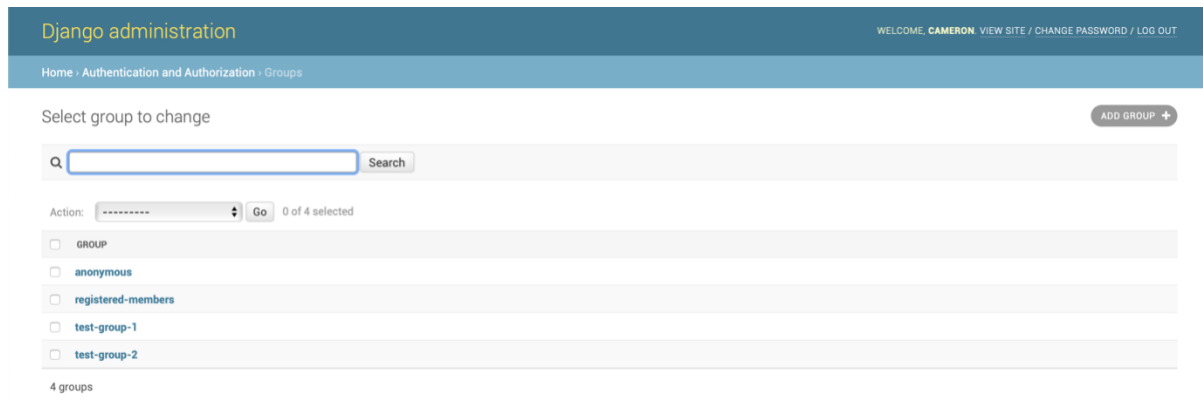


Figure 22: Authentication and Authorization Groups tool.

FR 4: Assign user administrator rights

Requirement:

The administrator shall be able to perform all functionalities that users can perform, manage user status and monitor all user activity.

The prototype administrator can assign the administrator rights and responsibilities to any other registered user of their choosing. These users then go from being a registered user with limited rights and responsibilities, to a prototype administrator with many more rights and responsibilities.

Using the Django Administration People tool, Figure 20, system administrators can change the user status of other users. System administrators can give a user 'Staff Status' or 'Super User Status' meaning different levels of rights and responsibilities. These new system administrators can then in turn, make other users system administrators.

4.3.3. Uploading data in the prototype

The users of the prototype need to be able to upload all their data. Due to all the various data formats that are collected, the prototype needs to be capable of handling all those formats. The user needs to be able to upload their spatial data files such as vector files (i.e. shapefiles) and their raster files (i.e. GeoTiffs) and then associate the appropriate metadata to the spatial data files. The user also needs to be able to upload their non-spatial data files such as portable document format (PDF) files, MS Word documents, MS Excel sheets, images (.png and .jpeg) as well as ZIP files and then associate the appropriate metadata to the non-spatial data files. Users may also wish to link spatial data files with one or more non-spatial data files, as one provides content to the other. For example, a spatial data file. of GPS coordinates and its attributes. To protect the spatial and non-spatial data files in the prototype, permission is

granted by the owner of the spatial and non-spatial data files to other users to allow them to view and edit the spatial and non-spatial data files and the associated metadata.

FR 5: Upload a spatial layer to the prototype

Requirement:

A user shall be able to upload a spatial layer to the prototype.

In order to upload a spatial layer to the prototype, the user needs to have an approved account with the prototype. Once a layer has been uploaded, the layer is visible to those who have permission to view it and the user who uploaded the layer is regarded as the owner of the layer.

On the Upload Layers page in Figure 23, registered users have the ability to upload a single layer or multiple layers of their choosing. Two options are available for the uploading of layers, users can either drag layers from their local system and drop them in the upload box or they can search for the layers through the file manager of their operating system. The prototype warns users if the layer they are uploading is supported or if it is missing associated files such as the .prj file. The registered user who is uploading the layer can set permissions to other registered users or groups for the layer as to who can view, edit, and download the layer being uploaded. Once a layer has successfully been uploaded, the page then loads the Metadata Wizard for editing the metadata.

The screenshot shows the 'Upload Layers' interface. At the top, there's a navigation bar with the GeoNode logo and links for Data, Maps, and About. A search bar and a user profile for Cameron Green are also present. The main heading is 'Upload Layers' with an 'Explore Layers' button. The central area contains a large dashed box with a cloud and arrow icon and the text 'Drop files here'. Below this, there's a 'Choose Files' button and a section for 'Files to be uploaded' with a charset dropdown set to 'UTF-8/Unicode' and 'Upload files' and 'Clear' buttons. On the right, a 'Permissions' panel is visible, allowing users to set permissions for 'Who can view it?', 'Who can download it?', and 'Who can change metadata for it?'. Each permission section has a dropdown menu, a checked 'Anyone' option, and input fields for 'The following users:' and 'The following groups:'.

Figure 23: Upload Layers page

FR 6: Upload a non-spatial document to the prototype

Requirement:

A user shall be able to upload a non-spatial document to the prototype.

In order to upload a document to the prototype, the user needs to have an approved account with the prototype. Once a document has been uploaded, the document is visible to those who have permission to view it, and the user who uploaded the document is regarded as the owner of the document.

On the Upload Documents page, registered users have the ability to upload a single document at a time. The prototype clearly warns registered users what file types are allowed to be saved in the prototype. The registered user who is uploading the document can set permissions to other registered users or groups for the document as to who can view, edit, and download the document being uploaded. Registered users can also link the document to a layer or a map, giving it a spatial context. Once a document has successfully been uploaded, the page then loads the Metadata Wizard for editing the metadata.

FR 7: Add metadata to the spatial layer in the prototype

Requirement:

A user shall be able to upload metadata of any of the spatial data files that are associated to them in the prototype.

During the uploading of a layer, a user is able to add metadata to the data file by manually filling in the metadata fields or they have the option to upload the metadata. Only users who have permission to edit the metadata may edit the metadata fields.

The Metadata Wizard that can be seen in Figure 24 gives registered users the capability to add or edit any metadata to layers, documents, and maps. If a registered user is editing the metadata a layer, the Metadata Wizard is a four-step process.

The first Metadata Wizard page is the Basic Metadata page. Here, registered users are required to provide details such as the title, an abstract, keywords, date added, what category and group the layer, document, or map belongs to. They can also edit the thumbnail if they wish to do so. The second Metadata Wizard page is the Location and Licenses page. Here registered users are required to provide the language of the resource, the license, what region or country the data is in, a data quality statement, and whether there are any restrictions on the use of the data. The third and last for documents and maps Metadata Wizard page is the Optional Metadata page. Registered users can add any additional metadata such as the

edition of the resource, the purpose of the resource, any supplemental information, the maintenance frequency, and the parties (registered users) responsible for the data and its maintenance. The fourth Metadata Wizard page for layers is the Dataset Attributes page. Here registered users can view the attributes, edit their names and reorder them.

The Advanced Edit is exactly the same as the Metadata Wizard but instead of it being a three or four-step process, it is just one continuous page of metadata fields.

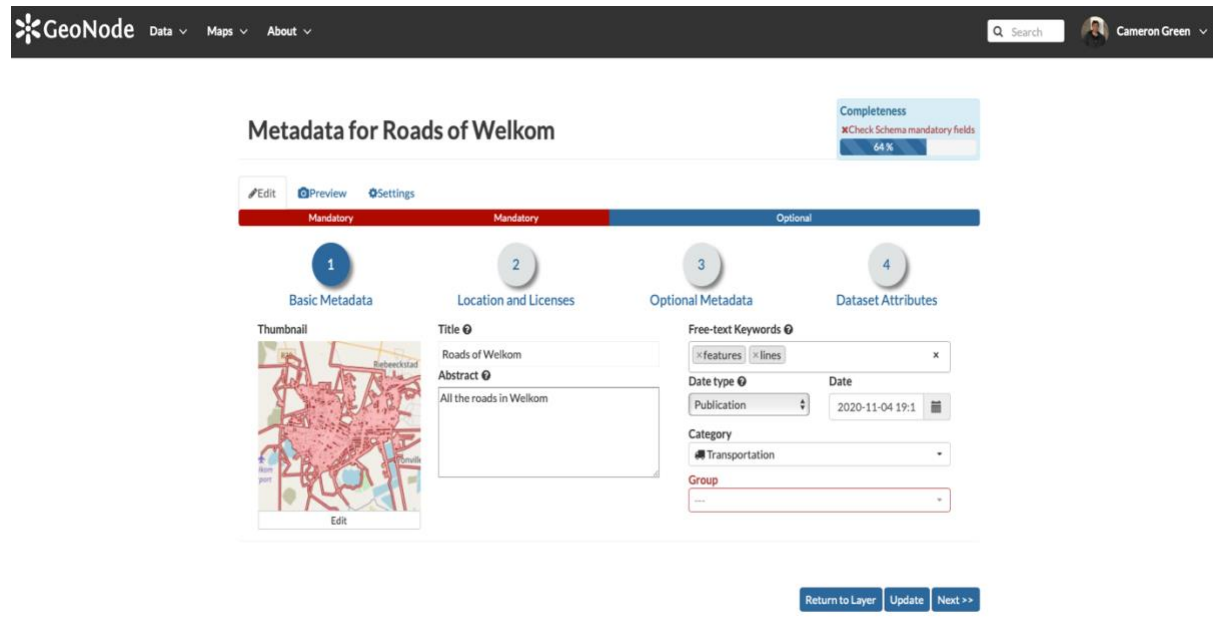


Figure 24: Metadata Wizard for Layers

FR 8: Add metadata to the non-spatial document in the prototype

Requirement:

A user shall be able to upload metadata of any of the non-spatial document that are associated to them in the prototype.

During the uploading of a document, a user is able to add metadata to the data file by manually filling in the metadata fields or they have the option to upload the metadata. Only users who have permission to edit the metadata may edit the metadata fields.

Adding metadata to a non-spatial document is the same process as with adding metadata to a spatial layer in the prototype. When a registered user is editing the metadata of a document or map, it is a three-step process. If users wish to edit the metadata in more detail, they can use the Advanced Edit tool. The Advanced Edit is exactly the same as the Metadata Wizard

but instead of it being a three-step process for spatial data or four-step process for non-spatial, it is just one continuous page of metadata fields.

FR 9: Associate a spatial layer file to a non-spatial document in the prototype

Requirement:

A user shall be able to associate spatial layer and non-spatial document together to supplement the data.

During the uploading of a document, users can associate the document with a layer. Once this is done, the link between the document and layer is now visible, and users can navigate between the two.

In the prototype, registered users have the ability to associate a spatial layer with a non-spatial document, which means that when a user views the spatial layer, they can see which non-spatial document is linked with it, and vice versa. This also applies to maps as users can see which layers and documents are linked to the map.

FR 10: Associate a spatial layer and non-spatial document to a group in the prototype

Requirement:

The user shall be able to associate spatial layer and/or non-spatial document with a group or groups.

When a layer or document is being uploaded to the prototype, the uploader can assign the layer or document to a group which means that the selected group can view, edit, and download the selected layer or document.

Registered users have the ability to associate a spatial layer or a non-spatial document with a group meaning that only users who are registered to that particular group can view, edit, and download the spatial layer or non-spatial document.

FR 11: Change permissions of a spatial layer in the prototype

Requirement:

A user shall be able to change the permissions of a spatial layer with which they are associated.

The owner of the layer may wish to add other researchers to the layer therefore the original permissions need to be updated. To do this, the owner can change the permissions allowing the new researchers to view, edit, and download the selected layer.

Registered users who have the appropriate permissions to edit the permissions of a spatial layer may do so. They can assign the permissions to other registered users so that they can view, edit and download the spatial layer while at the same time keeping the spatial layer private from other users.

FR 12: Change permissions of a non-spatial document in the prototype

Requirement:

A user shall be able to change the permissions of a non-spatial data file with which they are associated.

The owner of the document may wish to add other researchers to the document therefore the original permissions need to be updated. To do this, the owner can change the permissions allowing the new researchers to view, edit and download the selected document.

Registered users who have the appropriate permissions to edit the permissions of a non-spatial layer may do so. They can assign the permissions to other registered users so that they can view, edit, and download the non-spatial layer, while at the same time, keeping the spatial layer private from other users.

4.3.4. Viewing data in the prototype

Users of the prototype must be able to view the spatial and non-spatial documents that are uploaded to the prototype as well as any metadata that is associated to the spatial and non-spatial documents in the prototype. Depending on the permissions of the document, only certain users can view the document. If no permissions are set to the document, then registered and unregistered users can view the documents but if the permissions are set, then only certain registered users or groups can view the documents and metadata.

FR 13: Viewing spatial layers in the prototype

Requirement:

Users of the prototype shall be able to view spatial layers.

To view a layer in the prototype, the user requires permission to view the layer. When the layer is viewed by the user, the layer appears on an OpenStreetMap base map.

Users can view spatial layers that they have the appropriate permissions to view. Users can view these layers on top of an OpenStreetMap base map such as in Figure 25 and users can view the metadata of the spatial layer. Viewing the spatial layer in the prototype allows users to view the spatial layer without having to download it.

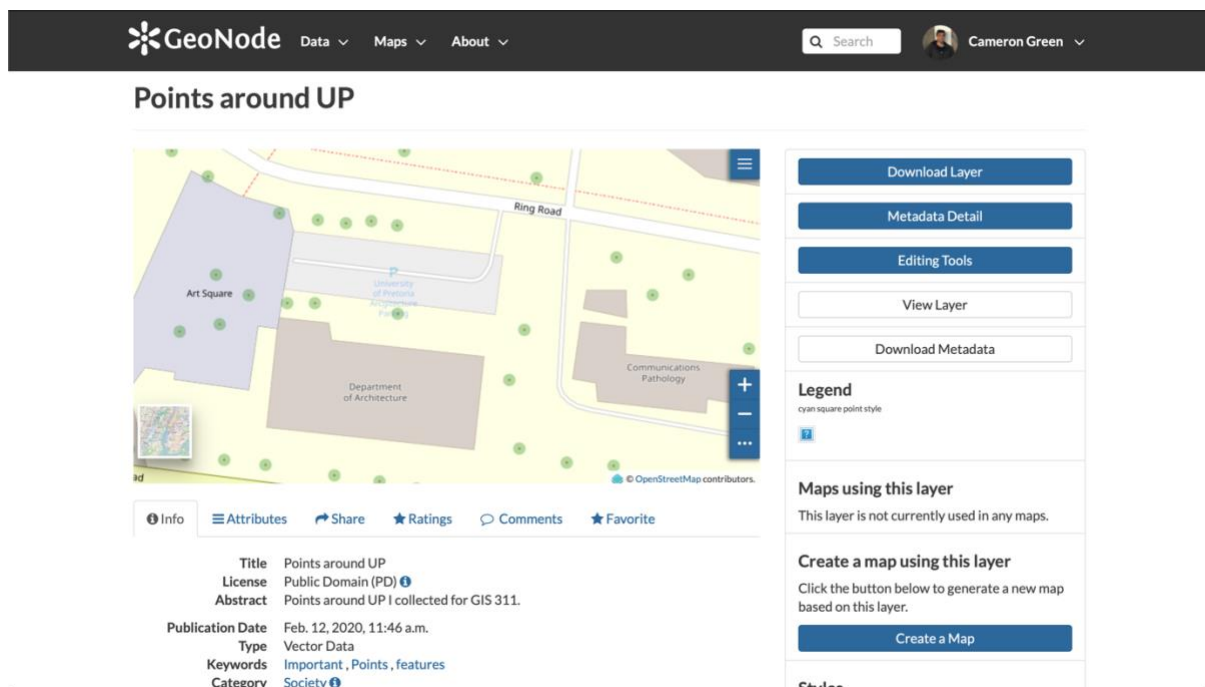


Figure 25: Viewing a layer

FR 14: Viewing non-spatial documents in the prototype

Requirement:

Users of the solution shall be able to view non-spatial documents.

To view a document in the prototype, the user needs permission to view the document.

When the document is viewed by the user, depending on the document type, the document is either displayed, or may be downloaded to be viewed.

Users can view non-spatial documents that they have the appropriate permissions to view. Depending on the non-spatial document type, the non-spatial document can be either viewed

within the prototype, or it has to be downloaded first. This differs from how spatial layers are viewed. Users can also see the metadata of the non-spatial document.

FR 15: Viewing metadata in the prototype

Requirement:

Users of the prototype shall be able to view the metadata of the spatial layer and non-spatial documents.

When a layer or document is uploaded to the prototype, the user adds metadata. Depending on the permissions of the layer or document, the user has the ability to view the metadata of the layer or document.

Users who have the appropriate permissions to view the metadata of a spatial layer or a non-spatial document may do so. The metadata is either created automatically by the prototype such as the upload date and owner of the spatial layer or non-spatial document, or the owner has to manually input some metadata such as the keywords, title, and abstract.

4.3.5. Updating data in the prototype

Users of the prototype will want to update the spatial layers and non-spatial documents that are currently in the prototype. They may wish to update the metadata of a document after getting more details about the spatial layer or non-spatial document such as keywords that relate to the document or they wish to update the permissions of a spatial layer or non-spatial document in order to allow for another user to view the document or download it. Data that was collected in previous years may be outdated, and users of the prototype may wish to update this data with a newer version by uploading a new spatial layer or non-spatial document.

FR 16: Updating metadata in the prototype

Requirement:

Users of the prototype shall be able to update the metadata of the spatial layer and non-spatial documents.

When metadata is uploaded, it may become outdated, and the user may wish to update it. The user, if permission is granted, can update the metadata by manually filling in the metadata fields or uploading a metadata file.

Updating the metadata of a spatial layer or a non-spatial document is an important step in maintaining the data integrity. Users who have the permission to edit the metadata of spatial layer or a non-spatial document may do so if they determine that the metadata requires updating. Users update metadata by making use of the Metadata Wizard (Figure 24). For example, if a user does not complete all the metadata when uploading a spatial layer or a non-spatial document, another user can update and complete the metadata.

FR 17: Updating permissions of spatial layers and non-spatial documents in the prototype

Requirement:

Users of the prototype shall be able to update the permissions of the spatial layer and non-spatial documents.

The owner of the layer or document can update the permissions of a layer or document and assign new permissions to other registered users or groups.

As a spatial layer or a non-spatial document can be restricted in terms of who can view, edit or download the spatial layer or a non-spatial document, these permissions can be updated should the owner wish to do so. The owner can restrict who can view, edit and download the spatial layer or a non-spatial document.

FR 18: Updating spatial layers and non-spatial documents with newer versions in the prototype

Requirement:

Users of the prototype shall be able to upload a new version of already existing spatial layer and non-spatial documents.

The owner of the layer or document can update the existing layer or document with a newer version of the same layer or document by either uploading as a separate layer or document or using the replace function.

When a layer, document, or map is selected by a user that is permitted to view the layer, document or map, the user will be directed to the next page. For layers, users are provided with more details about the layer as well as an OpenStreetMap (OSM) base map with the layer on it, which enables users to visualise the layer on a map as well as being able to see which maps currently use this layer and what styles this layer is using. For documents, depending on the document type, users can view the document without having to download it or they can use the preview window, and see which layers or maps are linked to this particular

document. For maps, the map is displayed on an OpenStreetMap (OSM) base map with the layer or layers that are used to create the map. These pages also provide extra metadata details, such as the license, keywords, region, type of data, spatial extent, and the permissions of the layer, document or map. Extra tools are provided to registered users such as a download option, metadata details option, edit option, download the metadata option, or view map option. A legend is provided on the layers and maps page as to explain to users the layers that are being used in the map, making the understanding of the layers and map easier. Registered users are also able to provide comments about the layer, document or map which other users, registered or not, will be able to see.

When a registered user wishes to edit a layer, document or map they are permitted to access, they may do so by clicking the 'Editing Tools' button (Figure 25), which brings up the Edit options that are presented in Figure 26. These are more detailed editing tools when compared to the Metadata Wizard. Here, registered users can edit the metadata through the Metadata Wizard or the Advanced edit option or they can upload metadata. Registered users can edit, upload and manage the styles the layer is using, a thumbnail image for the layer can be set or uploaded and finally, the layer can be replaced with a more recent or updated layer, the data in the layer could be edited using the GeoNode prototype interface, or it can be removed all together. Registered users can set or upload a thumbnail image for the document, where the document can be replaced with a more recent or updated document, or it can be removed all together. Registered users can set or upload a thumbnail image for the map and the map can be edited in the create map interface or it can be removed all together.

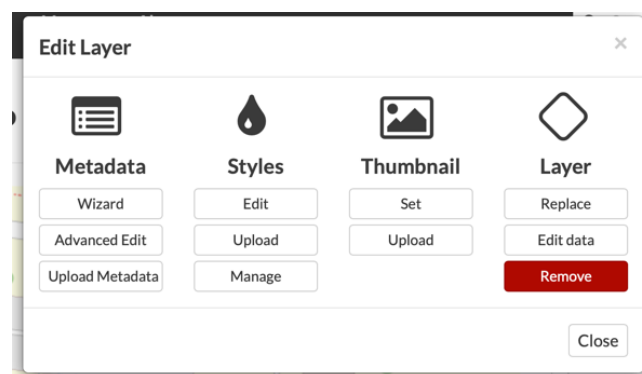


Figure 26: Editing layer tools

4.3.6. Finding data in the prototype

Users of the prototype will want to search for spatial or non-spatial documents in the prototype. They will not want to browse pages and pages for the spatial or non-spatial document they are looking for, but rather use a search or filter option which can speed up the process. In this

context, a search refers to starting from a blank page and populating it with results that meet the criteria, with a filter that hides the results that do not fit or match the criteria. Users can search a filter and filter a search. As metadata exists within the prototype, users may wish to search for documents based on this metadata, such as who uploaded the document and what keywords were used to describe the document. Users can also filter the documents based on the metadata. As there is a spatial context to the spatial documents that are in the prototype, users can filter the spatial documents based on a location.

There are three pages for exploring the data within the GeoNode Prototype. On the Explore page, users have the ability to view all the layers in the prototype they are permitted to view. The Explore Layers (Figure 27), Documents and Maps pages are identical to one another. On the Explore page, users have the ability to view all the layers, documents, and maps in the prototype they are permitted to view. Users have the ability to search for a layer, document and map and then filter the results based on criteria important to them. Users can search for a layer, document and map based on keywords, plain text, categories, owners, and groups. Users also have the option to filter the layer, document and map based on the spatial extent, using an interactive map. The Explore page provides a basic overview of the layers, documents, and maps, such as the name of the layer, document and map, the owner, an abstract, which category the layer, document and map belongs to, and the date added. If registered users do not find a layer, document and map they are looking for, they have the ability to upload a layer or document or create a map.

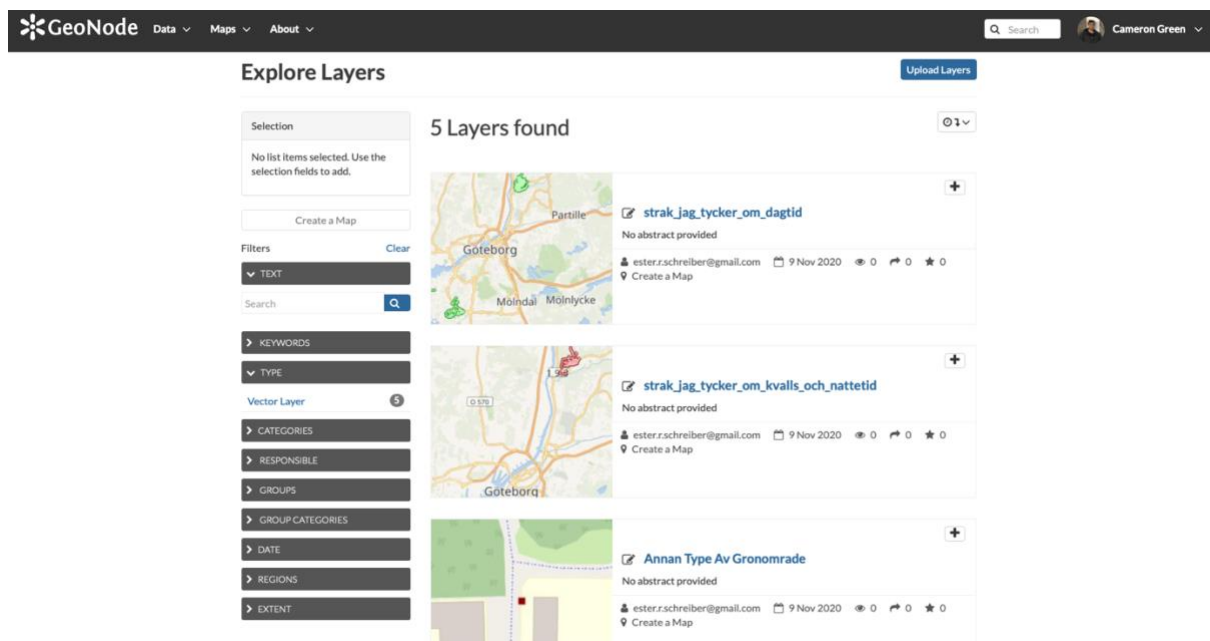


Figure 27: Explore Layers page

FR 19: Search based on metadata of a spatial layer or a non-spatial document in the prototype

Requirement:

The user shall be able to search for spatial layers and non-spatial documents in the prototype based on the metadata of the documents.

A user can search for a layer or document based on the metadata that was uploaded during the upload process. The prototype will return all the layers or documents that meet all the search criteria.

If a user wishes to search for a specific spatial layer or non-spatial document, they can use the search functionality that the prototype offers. Users can search based on many different criteria such as keywords and plain text. Users can also search filtered data.

FR 20: Filter based on metadata of a spatial layer or a non-spatial document in the prototype

Requirement:

Users shall be able to filter the spatial layers and non-spatial documents in the prototype based on the metadata of the documents.

A user can filter all the layers or documents based on the metadata that was uploaded during the upload process. The prototype will return all the layers or documents that meet all the filter criteria.

If a user wishes to filter the spatial layers or non-spatial documents, they can do so. Users can filter based on what category it belong to, owner of the data, what group is it linked to, date and many more.

FR 21: Filter based on location of a spatial layer or a non-spatial document in the prototype

Requirement:

Users shall be able to filter the spatial layers and non-spatial documents in the prototype based on the location of the documents.

A user can filter all the layers or documents based on the spatial extent of the layer or document as documents can have spatial extents. The prototype will return all the layers or documents that are within the spatial extent.

The prototype offers an extra level of filtering to users as users are allowed to filter spatial layers or non-spatial documents based on the region or extent of the spatial layer or non-spatial document. This is done by using an OpenStreetMap base map.

4.3.7. Making data available in the prototype

Users of the prototype may wish to share any of the spatial and non-spatial documents that are stored in the prototype. Users may wish to download or export any of the data in the prototype for their own use in their project. The spatial and non-spatial documents can be exported in the various formats that the prototype supports such as portable document format (PDF) format, MS Word documents, MS Excel sheets, images (.png and .jpeg), shapefiles, GeoJSON and GeoTiffs. More technical users may wish to share the spatial documents via web services such as Open Geospatial Consortium (OGC) web services. Documents can also be shared via the uniform resource identifier (URI).

FR 22: Share the data via the uniform resource identifier (URI)

Requirement:

The users of the prototype shall be able to share the spatial layers and non-spatial documents via their URI.

Users can share data to other registered users by using the URI. Users who have the URI can access the data via the URI.

Sharing data that is in the prototype is an important requirement that needs to be met. Users that intend to share either a spatial layer or non-spatial document may do so by using the URI and sharing that with another user. Each URI is unique to a specific spatial layer or non-spatial document.

FR 23: Export the data in the various formats from the prototype

Requirement:

The user shall be able to export the spatial layers and non-spatial documents in the prototype in the various formats.

Users can export a layer or document, if they are permitted to do so, in various formats depending on the layer or document.

After finding the spatial layers or non-spatial documents they are looking for, users may export the spatial layers or non-spatial documents if they have the appropriate permissions to do so.

Spatial layers can be exported in various formats such as shapefile, GeoJSON, Excel, CSV, GML, or as an image, while non-spatial documents can only be exported in the format in which it was uploaded.

FR 24: Expose the data via OGC web services

Requirement:

The user shall be able to expose the spatial layers in the solution via OGC web services. Users who have permission, can expose the layers via OGC web services for more technical users.

Users that wish to expose spatial layers via OGC web services may do so. An example of this is that technical users can connect QGIS to the prototype and either use the spatial layer as a Web Feature Service (WFS), where they can edit the raw data or as a Web Map Service (WMS) serving the spatial layer as an image.

4.3.8. Non-functional Requirements

The non-functional requirements of this project were met by the GeoNode prototype. Meeting these non-functional requirements were important to ensure the success of the prototype.

NF1. Accessibility

The prototype was built on a public server that is accessible by the students and is hosted by DigitalOcean (<https://www.digitalocean.com>). A public URL, <https://geocatalogue.co.za>, was also used to enhance the accessibility of the prototype allowing for the public sharing of the prototype and its data.

NF2. Data Integrity

Data that is stored in the prototype is saved in a PostGIS database, which maintains the integrity of the data and follows metadata standards, such as ISO 19115. The prototype performs automated checks so as to ensure no corrupt data is uploaded to the solution. For example, if a user is uploading a shapefile and a supporting file such as the projection metadata (.prj) file is missing, GeoNode will alert the user and prevent the shapefile from being uploaded.

NF3. Documentation of code

The prototyped code is well documented on GitHub (<https://github.com/GeoNode/geonode>), with clear instructions on the correct method of installation for the GeoNode prototype.

NF4. Ease of Use

GeoNode is designed so that non-technical users can use GeoNode to the fullest of its capabilities which makes the designing of this prototype easier. It has a user-friendly interface with step-by-step instructions along the way.

NF5. Interoperability

GeoNode is a cross-platform application that renders correctly on all operating systems and browsers. GeoNode also uses OGC web services to correctly render and share spatial data. As the prototype was installed on a server, it could be accessed on multiple internet-enabled devices enabling it to render correctly.

NF6. Longevity

GeoNode version 2.10 was used for the prototype meaning the only concern for project longevity is that the prototype is using Python 2.7, which will eventually be deprecated and GeoNode will no longer maintain it. To solve this problem, the prototype ought to be moved to the latest version of GeoNode which uses Python 3.0. GeoNode is an OSGeo project, which has a strong developer and user base.

NF7. Online Capabilities

GeoNode is an online platform that supports multi-person use as it allows users to collaborate with other professionals. The prototype works on a server that allows it to have online capabilities.

NF8. Privacy

As personal data is collected by the users, it is important that the data that is stored in the prototype is secure and private. Certain data can be restricted from being viewed and downloaded by other users. The server the prototype was installed on uses LetsEncrypt (<https://letsencrypt.org>) and CertBot (<https://certbot.eff.org>), making the communication secure. Users can hide or display any personal data that is stored in their profile.

NF9. Reliability

The prototype used GeoNode version 2.10, which is a stable version and performance is not an issue. GeoNode is an OSGeo backed project, which means that a team of volunteers and developers maintain GeoNode and correct most bugs that may arise which reduces the amount of downtime.

NF10. Reusability

As the prototype was setup and moved into a production environment, it is operational and users can reuse the prototype over multiple years and in different use cases.

NF11. Robustness

As the prototype was installed on a stable server, it is highly robust, and downtime between reboots is minimal. A Docker install was not done, which means that there is a separate Docker container with the data in a PostgreSQL database, which is easily backed up as necessary. Using a Docker install can increase the robustness as components are kept separate.

NF12. Scalability

As the prototype can be accessed via an internet-enabled device and it was hosted on a powerful server, it has the ability to adjust to the number of users using it at any given time as increasing the CPU, RAM and hard disk space allows the application to be scaled to more users..

NF13. Security

On the server the prototype was installed, LetsEncrypt (<https://letsencrypt.org>) and CertBot (<https://certbot.eff.org>) were installed to protect communication between the server and user.

NF14. Speed

A powerful server was used to host the prototype, where it was quicker for the uploading of layers and documents and processing any requests. If users have a strong and fast network connection, the uploading process

NF15. User Documentation

The documentation of use for GeoNode is publicly available under the GeoNode User's Guide (<https://docs.geonode.org/en/2.10/usage/index.html>). The documentation is well kept and up to date with the versions of GeoNode. Also, documentation was developed for this specific prototype, which will be discussed later in 4.4. Documentation – Design and Implementation.

4.4. Documentation – Design and Implementation

The aim of developing new documentation was to assist the architectural students with using the prototyped solution. The architectural students and lecturers do not have an in-depth geospatial background, therefore the documentation needs to be aimed at this group of users.

The documentation also needed to be explained in such a way that anyone who was not able to attend the training on the prototype or arrived late in the design studio was able to join in and not fall behind.

Although already existing documentation is available for the users of the GeoNode prototype, it was decided that new and supporting documentation was also be drawn up that was more focused and aligned to the architects use of the prototype. Special attention was given to the features that the architects would be using repeatedly or may struggle to use. YouTube videos were developed as a visual aid in relation to the reading material, so that they would support each other. In certain cases where the architectural students may be unable to follow the text, the videos may provide more clarity on the issue, as users may be more visual learners, rather than learners that respond better to text-based resources. The new documentation has links to the already existing documentation for other issues that may arise, which are not covered in the new documentation.

The documentation took some time to develop as the features that the architects would be using repeatedly or may struggle to use first had to be identified. Once this was done, the process of writing the text-based resources began and then the recording of the process followed. For example, for the process of uploading data, the steps to describe would be: (1) to load the prototype in a web browser, (2) log in using their approved account, (3) and then depending on what they would like to upload, a layer or document, they then select the option of their choosing and an uploading screen will appear. Users will then upload the document or layer and fill in the metadata details that are required by the prototype. Once all the details are filled in and confirmed, the layer or document will be successfully uploaded. The text-based resource does not include the step-by-step instructions on how to do this, but does provide some instructions, tips, and hints for the uploading process, such selecting appropriate file names, as well as filling out the metadata, and how to correctly set the permissions of each layer or document.

Finally, the text-based resource points the user of the documentation to the YouTube video that explains how to follow the process step-by-step. Once all the text-based resources were drawn up and the YouTube videos published, the documentation was demonstrated to the architectural lecturers for suggestions on improving the new documentation. These suggestions were then taken into consideration and the documentation was thereafter improved. This process was repeated numerous times before and even during Test Case One – Hammarkullen, Sweden, which is common in the design science paradigm. The architectural students then used this new documentation in Test Case One – Hammarkullen, Sweden and

suggestions were also taken from the students and implemented before Test Case Two – Mamelodi, South Africa.

The documentation is all available online as to assist with NF4. Ease of Use as well as to add to the documentation in NF15. User Documentation. Any changes that are requested to be made to the documentation get reflected immediately, and there are no outdated versions of the documentation to confuse the users of the prototype. Having all the documentation online in a GitHub page also ties into NF6. Longevity, for example, instead of having the documentation saved on a hard drive that may become corrupted, the documentation is freely available online. The documentation was first available as a Wikipedia page, but then it was later decided to host the documentation on a GitHub page as version control and issue tracking was available.

4.4.1.1. The Text-based Resources

The documentation is a combination of text-based resources and a series of YouTube tutorial style videos. The text-based resources are available on a GitHub page as a README.md file. The README.md file on GitHub (Figure 28) is a lightweight mark-up language with plain text formatting syntax. It is a very simple language used to create presentable readme files for any and all GitHub pages.

The series of YouTube videos are divided into playlists (Figure 29) depending on what content they are explaining such as how to work with QGIS or EpiCollect5 or GeoNode. With the text-based resources being made available on a GitHub page, this also offers a unique feature of issue tracking. Users of the text-based resources can log an issue with the content using GitHub's built-in issue tracker, and anyone who has write access to the README.md file can edit it to correct the issue that has been logged, where these changes are reflected instantly. All throughout the text, hyperlinks are present that connect users of the documentation to all of the YouTube videos and other related documentation for the tools are used. Even though the GeoNode prototype is focused on data storage and management, the documentation also included some information about data collection and what tools can be used for this.

All the documentation is available at: https://github.com/CamGreen/NRF-STINT_Wiki. The documentation covers the following topics that are relevant to the prototype and is structured in the following way:

1) Data Collection Tools

- a) When to use these types of data collection tools?

- b) KoBoToolbox
 - i) Setting up a project*
 - ii) Using the application*
 - iii) Viewing and downloading the collected data*
- c) EpiCollect5
 - i) Setting up a project and some useful tips*
 - ii) Using the application and some useful tips*
 - iii) Viewing and downloading the collected data with some useful tips*
- d) OpenStreetMap (OSM)
 - i) Creating an account*
 - ii) How to edit and some useful tips*
 - iii) How to download from OSM*
- e) Field Paper and JOSSM
- f) Maptionnaire

2) Overview of GeoNode (Stitching the City)

- a) What is GeoNode?
- b) Why would you use GeoNode? Why do you need to manage and share your data?
- c) What data can you manage with GeoNode?
 - i) Document formats*
 - ii) Layer formats*
 - iii) Please note warning*

3) Using Stitching the City (GeoNode) for your data

- a) Uploading data
- b) Viewing and downloading data
- c) Creating, editing and downloading maps
- d) Editing Metadata

4) Useful tricks with QGIS

- a) Converting between data formats
 - i) Converting CSV to SHP*
 - ii) Convert GeoJSON to SHP*
- b) Editing vector data
- c) Viewing or editing your GeoNode data in QGIS

5) Additional resources

- a) EpiCollect5
 - i) EpiCollect5 Data Collection Users Guide*
- b) OpenStreetMap (OSM)
 - i) OSM Beginner's Guide*

- ii) *Learn OSM*
 - iii) *Learn OSM beginner's Guide*
- c) **GeoNode**
 - i) *GeoNode FAQs*
 - ii) *GeoNode Complete Users Guide*
- d) **QGIS**
 - i) *QGIS User Guide*
 - ii) *A Gentle Introduction to QGIS*
 - iii) *QGIS Training Manual*
 - iv) *Introducing GIS Tutorial with QGIS*
 - v) *GIS Basics with QGIS*
 - vi) *Learn GIS for free: The complete course*
- e) **Other**
 - i) *Colour Brewer – Colour Advice for Maps*

6) Video Links

- a) **EpiCollect5**
 - i) *Setting up a project*
 - ii) *Using the application*
 - iii) *Viewing and downloading the collected data*
- b) **OpenStreetMap (OSM)**
 - i) *Creating an account with OSM*
 - ii) *Mapping buildings in OSM*
- c) **Using GeoNode for your data:**
 - i) *Uploading a layer to GeoNode*
 - ii) *Viewing and downloading a layer in GeoNode*
 - iii) *Creating, editing and downloading a map in GeoNode*
 - iv) *Editing metadata in GeoNode*
- d) **QGIS**
 - i) *Converting .CSV files to Shapefiles (.shp) in QGIS*
 - ii) *Converting GeoJSON files to Shapefiles (.shp) in QGIS*
 - iii) *Opening data from GeoNode in QGIS*

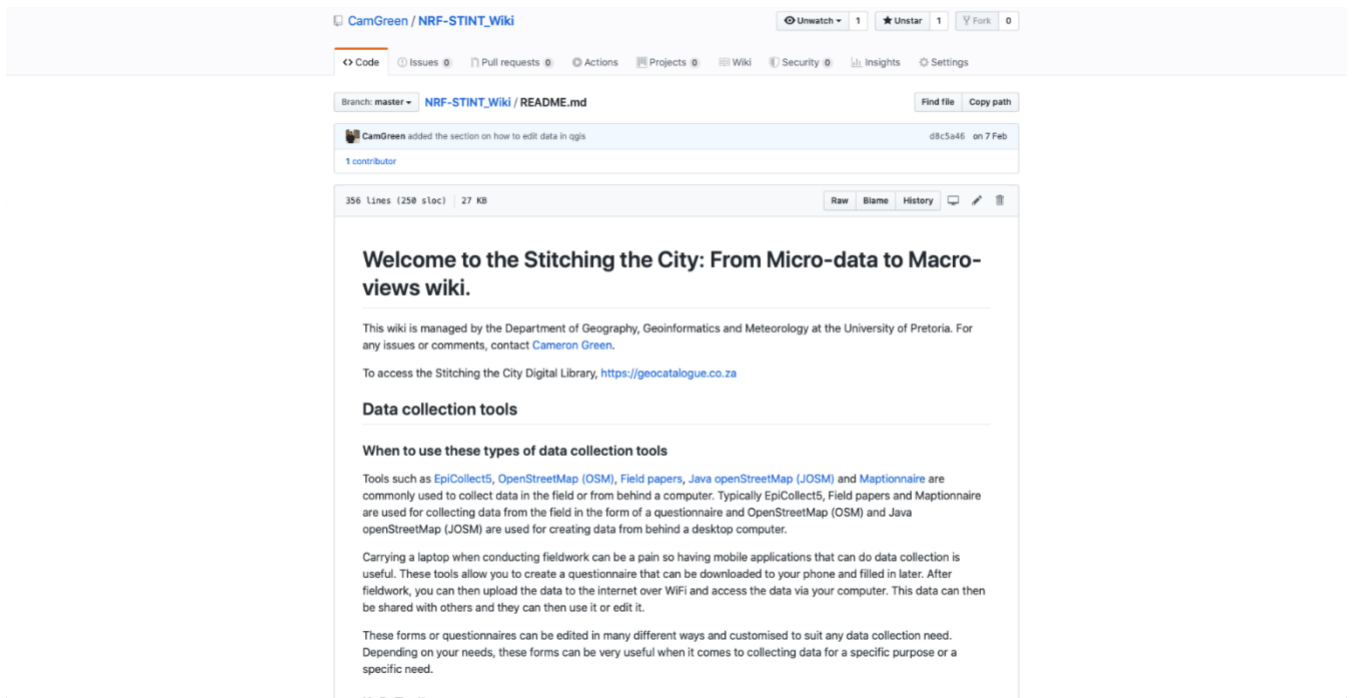


Figure 28: GitHub README.md text file

4.4.1.2. The YouTube Videos

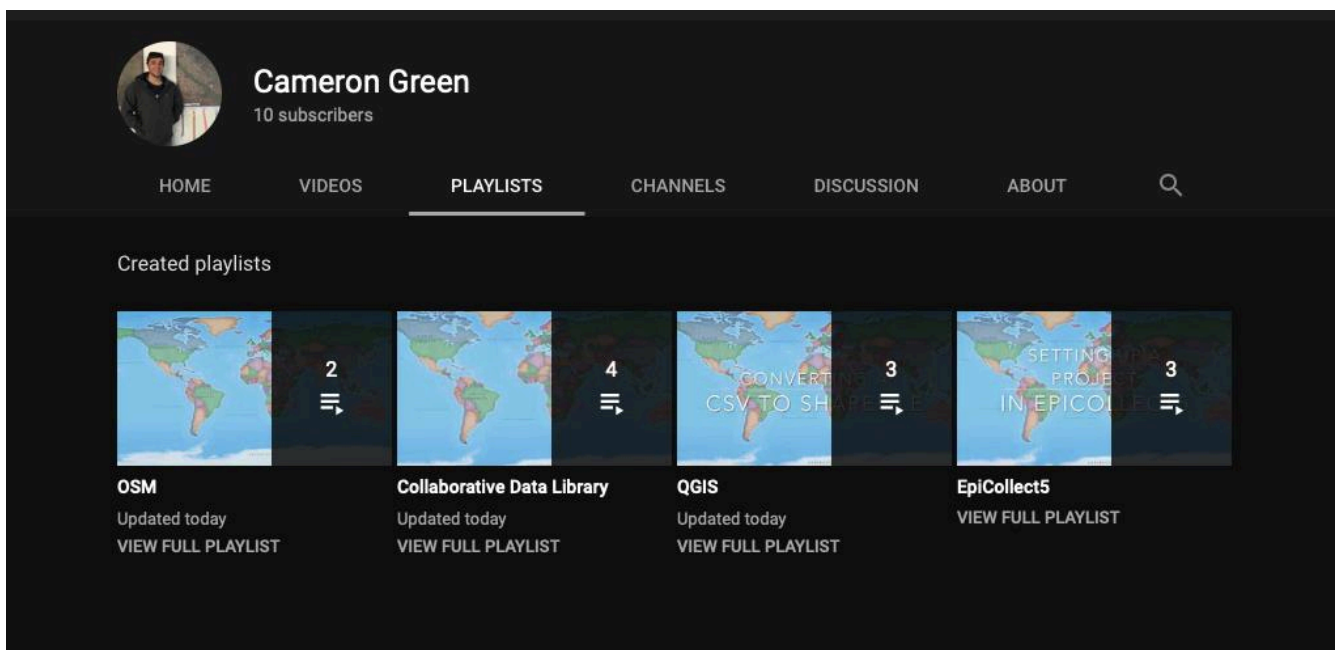


Figure 29: YouTube Playlists for QGIS, Collaborative Data Library, OSM and EpiCollect5

The YouTube videos are publicly available and are grouped into playlists depending on what content they are explaining such as how to work with QGIS, EpiCollect5, OpenStreetMap (OSM), or the GeoNode prototype.

The QGIS playlist, <https://bit.ly/30z72Tf>, contains three videos, converting .CSV files to shapefiles (.shp) using QGIS, converting GeoJSON files to shapefiles (.shp) in QGIS and opening data from GeoNode in QGIS.

The Collaborative Data Library or the GeoNode prototype playlist, <https://bit.ly/2UEyEml>, contains four videos on how to upload a layer to GeoNode, view and download a layer in GeoNode, create, edit and download a map in GeoNode and edit metadata in GeoNode.

The OpenStreetMap (OSM) playlist, <https://bit.ly/2XUDaz1>, contains two videos about creating an account with OSM and mapping buildings in OSM.

The EpiCollect5 playlist, <https://bit.ly/37nfS85>, contains three videos on setting up a project, using the application and viewing and downloading the collected data.

Chapter 5 – Evaluation and Discussion of the Prototyped Solution

5.1. Chapter Overview

The test cases where the GeoNode prototype and its associated documentation were used are discussed in detail in this chapter. The observations that were made during these test cases were used to evaluate whether the GeoNode prototype met all of the requirements of the architectural studios at the University of Pretoria and Chalmers University of Technology. The observations were made during September 2019 in Hammarkullen, Sweden, and during January 2020 in Mamelodi, South Africa. This chapter covers Research Objective 4.

5.2. Test Case One – Hammarkullen, Sweden

Test Case One took place over the week of 23–27 September 2019 at the Chalmers University of Technology campus in Hammarkullen, Sweden. Hammarkullen is a satellite suburb located roughly 10km northeast of the city centre of Gothenburg, the second largest city in Sweden, with a population of around 572 000 people. The area of Hammarkullen was constructed during the Million Programme period in the 1960s and 1970s (Hall and Viden, 2005). This was in response to the rising need for urgent housing and degradation in the city centre. Individuals can get from the Gothenburg city centre to Hammarkullen in 15 minutes with public transport, but this area has long been stigmatised, and there has been little interest from developers in investing in its refurbishment and upgrade. Hammarkullen is made up of 96 subareas, covers an area of roughly 3,6 square kilometres, with a population of 8241 people in 2019 with 84 different nationalities currently present in the area (Gothenburg City Management Office, 2020) making Hammarkullen a multicultural melting pot of different nationalities, which means many different social lives intertwine. The Chalmers University of Technology's Design and Planning for Social Inclusion design studio has a studio in Hammarkullen, where Master's students can interact with the community from and complete their design projects.



Figure 30: Hammarkullen (Source: OpenStreetMap)

During the week of observations of the local area and data collection, multiple observations were made by the architectural students and they collected a large amount of data in various formats, such as notes, sketches, images, voice recordings, videos, and points of interest. It is important to mention that the first part of the week was spent learning about the prototype and the other tools such as QGIS and KoBoToolbox, and getting to learn and know the community, the actual mapping came in the second part of the week. This is detailed in Table 10 below. One key thing to note is that this information is both analogue and digital in nature. All of this information now needs to be converted into meaningful data, and saved somewhere safe and reliable. Before the architectural students were given free reign over what data they needed to collect, they were first shown a presentation of what data needed to be collected, and what some of the key aspects they needed to be aware of, such as how to politely interact and have a conversation with a local resident of Hammarkullen. The demonstration also covered data collection tools such as KoBoToolbox, Maptionnaire and FieldPapers, as well as the GeoNode prototype.

Table 10: Overview of the week's activities for Test Case One, 23 - 25 September 2019

Day	Primary Focus
Day 1 – Monday 23 September	First impressions without technology, introduction to studio and Hammarkullen area.
Day 2 – Tuesday 24 September	Analogue mapping by the architectural students of social aspects in the community and the geospatial research team loaded the analogue data into the GeoNode prototype.
Day 3 – Wednesday 25 September	Digital mapping using KoBoToolbox, creating shapefiles in QGIS and loading the data into the GeoNode prototype.
Day 4 – Thursday 26 September	Digital mapping using KoBoToolbox and loaded the data into the GeoNode prototype.
Day 5 – Friday 27 September	Student feedback session.

Day 1 – Monday 23 September 2019:

On the first day of the week, the architectural students were required to gain an understanding of the environment through something that was called first impressions. This is where the architectural students were first exposed to the local environment without the use of any digital tools, aerial imagery, or maps. When the architectural students were walking around the Hammarkullen area, they developed a mental map of the area, trying to get an understanding of the environment they were working in. Once the architectural students were comfortable and familiar with the area, they regrouped in the meeting venue to discuss what they have seen, what they learnt and what they found very interesting about the area. The geospatial research team also spent some time learning the area and its layout as well as preparing for the next day.

Day 2 – Tuesday 24 September 2019:

During the course of the second day, the students completed the analogue mapping part of their project. This is similar to the first impressions, but this time the students were aware of the local area and they could plan what social aspects, such as play grounds or safe areas, they wish to document. They documented these social aspects by drawing sketches. The architectural students were asked to locate where they drew their sketches. An aerial image measuring 1,78m x 1,68m of Hammarkullen was pinned on the wall and the architectural students then placed stickers with a unique code that relates to them on the aerial image. The

sketches were scanned and loaded to the GeoNode prototype as an image. The geospatial research team, composed of the researcher of this dissertation and a supervisor then digitised all these locations using FieldPapers and QGIS to create shapefiles and georectified tiffs, both of which were loaded into the GeoNode prototype by the research team. A simplified version of events can be seen in Figure 31.

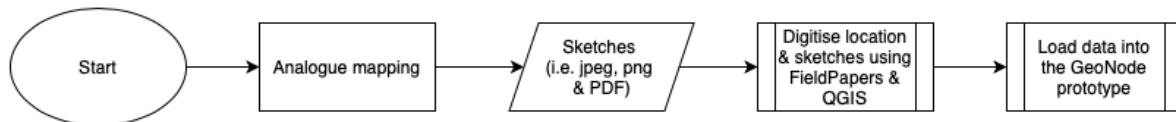


Figure 31: Processes followed on Day Two

Day 3 – Wednesday 25 September 2019:

It was during the third day that the students were introduced to digital mapping using data collections tools such as KoBoToolbox and Maptionnaire. The architectural students were presented with a presentation on how to use these tools, the importance of being accurate when collecting their data, as well as why it was important to be as descriptive as possible with the metadata. The GeoNode prototype was also demonstrated to the architectural students, as they were required to load their collected data into the GeoNode prototype at the end of the day.

Once the architectural students had collected their data, they were then presented with a step-by-step interactive demonstration on how to export their data from KoBoToolbox, create shapefiles using QGIS, and then import those shapefiles into the GeoNode prototype (Figure 32).

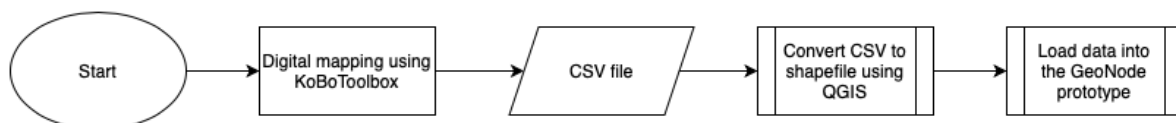


Figure 32: Processes followed on Day Three

Day 4 – Thursday 26 September 2019:

Following on from the work of the previous day, the architectural students then went and mapped similar features to the day before, but this time, these students used Maptionnaire. Maptionnaire creates shapefiles automatically for users therefore the architectural students did not have to create shapefiles using QGIS. The architectural students loaded the shapefiles and Microsoft Excel sheet to the GeoNode prototype using the knowledge they gained from the step-by-step interactive demonstration they received the day before. As the architectural

students were more confident in their use of the GeoNode prototype, the uploading went quicker on Day Four. This process can be seen in Figure 33.

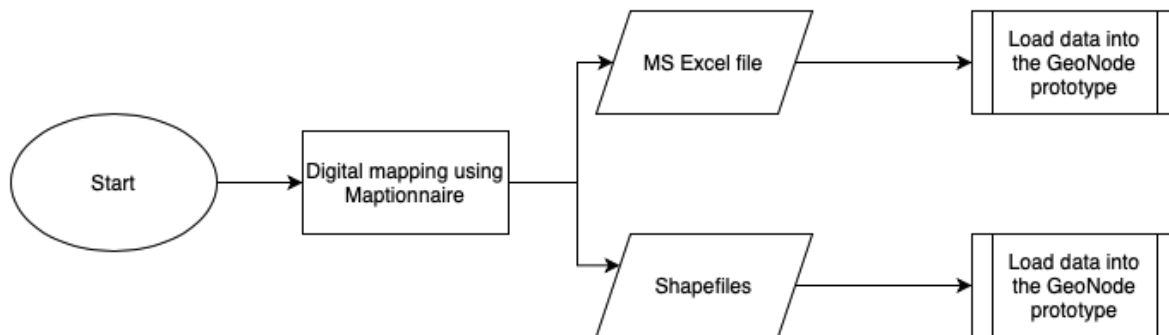


Figure 33: Processes followed on Day Four

Day 5 – Friday 27 September 2019:

This day was reserved for a group discussion on the week’s activities. The architectural students discussed what they enjoyed, what they found easy, what they struggled with and then also provided some specific feedback related to the GeoNode prototype. This feedback session was a similar length of time as the training sessions which lasted almost 3 hours. The feedback that was provided was discussed later in this chapter.

5.3. Test Case Two – Mamelodi, South Africa

This workshop took place over two weeks, 27 January 2020-07 February 2020, and was hosted at the University of Pretoria’s Hatfield campus and the mapping was done in Mamelodi East, Pretoria. Mamelodi East is the study area for the Urban Citizen Studio, where architectural students from the University interact with the community and collect data about their day-to-day life. As a result of South Africa’s apartheid past, Mamelodi East is a historically disadvantaged township that was segregated from Pretoria’s central business district (CBD). Mamelodi East is roughly 40 minutes away from the Pretoria city centre in comparison to the 15-minute commute of Hammarkullen. The last official census of 2011 (<https://census2011.adrianfrith.com/place/799046>) documented 334 577 people living in Mamelodi, an area that covers approximately 45,19km², although this number is disputed, and it is estimated to be closer to one million people (Van Eck, Renkin and Ntakirutimana, 2016). As with many African informal settlements, there is a high percentage of unemployment in the area, which forces the local inhabitants into entrepreneurial practices (Mothibi and Malebana, 2019).

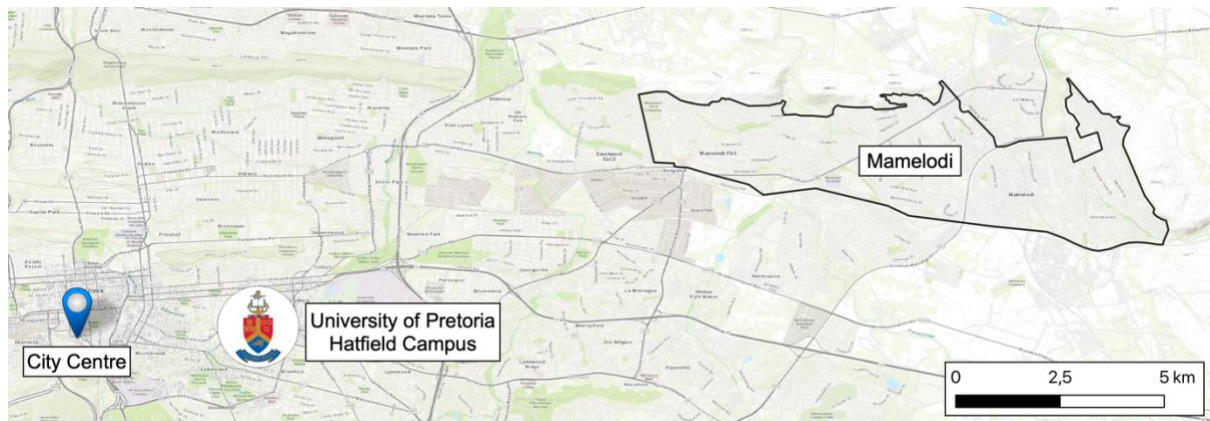


Figure 34: Mamelodi East (Source: OpenStreetMap)

This workshop was run in a similar fashion as to the first workshop in Hammarkullen, but this time, the geospatial research team, composed of the researcher of this dissertation and his two supervisors, took a stepped-back approach, where they were not directly involved with any of the training of the new architectural students, but instead, were nearby to answer any queries or solve any technical issues when it came to KoBoToolbox, QGIS, and GeoNode. This stepped back approach allowed for the research team to evaluate the usability of the GeoNode prototype and the documentation without any input from the research team that designed and implemented the GeoNode prototype. During this workshop, not as much data was collected as in the first workshop and the architectural students that were present in Hammarkullen, presented the training, based off the training they received in Hammarkullen.

As the workshop was over the course of two weeks, the geospatial research team was not present each day of the two-week workshop, but only present for a selection of the days. This was in order to provide room to the architectural students and architectural research team which was composed of six architectural lecturers, to fully use the GeoNode prototype and its documentation without any interference or biased views from the geospatial research team.

Table 11: Overview of the week's activities for Test Case Two, 27 January – 07 February 2020

Day	Primary Focus
Monday 27 January	Reflection and recap session of Test Case One. Lessons learnt that needed to be brought forward in Test Case Two in Mamelodi, South Africa.
Thursday 30 January	Personal demonstration to a smaller group of architectural students and answering any questions they had.
Friday 07 February	Full day of training for a new group of architectural

	students that have not participated in the Urban Citizen Studio.
--	--

Day 1 – Monday 27 January 2020:

Day one of the two-week workshop was a reflection and recap session of all the activities that were completed in Test Case One – Hammarkullen, Sweden as well as the lessons that were learnt and needed to be brought forward in Test Case Two – Mamelodi, South Africa. At this stage of the project it had been a while since the architectural students had worked with the GeoNode prototype, therefore they needed to review the documentation again.

Day 2 – Thursday 30 January 2020:

The architectural students started to develop their own version of the documentation to customise the documentation to make it more approachable for their peers that later became a methodological framework for collaboratively working with data in the spatial design principles. For this, they were given another demonstration, but on a more personal basis, as it was in a smaller group where they could ask questions at any step along the way. All the relevant steps that the architectural students needed to be aware of and know how to use were covered. This session ended with the architectural students understanding the bigger picture and feeling more confident to go and work with the GeoNode prototype in their own time.

Day 3 – Friday 07 February 2020:

This was a full day of training for a new group of architectural students that have not participated in the Urban Citizen Studio previously. The new group of architectural students were in the honours programme as well as the Master's students, who had already completed the module and also completed the training in Hammarkullen. The purpose of this training session was to familiarise the new students with the GeoNode prototype, and this was done by the architectural students that were previously trained in Hammarkullen. To assist with this, a member of the geospatial research team was present for the training on this day in case there were any technical issues that needed to be sorted out and that member was the researcher of this dissertation.

5.4. Discussion of Results

5.4.1. Before starting

Before the prototype could be designed and built, the problem at hand needed to be studied and understood. The problem that needed to be solved at the beginning of the project was to

provide the architectural students with a way to correctly and effectively managing their spatial and non-spatial data as well as to control and manage the metadata that is associated with their spatial and non-spatial data. In early August of 2019, the architectural lecturers were expecting one tool for data collection, conversion, and management, but this was not the case. To bridge this gap, documentation was created to assist with the data collection and data conversion. As with any application or project, first impressions are important, since this can determine the response and willingness to use the application or project by the intended user, and in this case the intended users were the architectural students. GeoNode was designed with a non-technical user interface meaning that people who do not have a spatial background can use GeoNode with relative ease. It was understandable that the architectural students were uncomfortable at first with the GeoNode prototype as the concept was strange and unfamiliar to them, but that as time went on and the architectural students used the GeoNode prototype more and more, they became increasingly confident in using it. This can be understood as the architectural students only really had a couple of hours a day to actually interact and work with the GeoNode prototype during the test cases. Modern day students require an adequate amount of time to be able to manage their academic time, as well as the social activities of their daily lives. All of these activities are determined by the student's ability to achieve these objectives without anxiety, stress, or the feeling of inability to take on all these tasks (Barbera, Gros and Kirschner, 2015; Alvarez Sainz, Ferrero and Ugidos, 2019).

5.4.2. The Test Cases

The objective of the test cases was not to see if the students can learn to work with the GeoNode prototype in such a short amount of time, but to rather see if the GeoNode prototype could be used for all their data and to see if the GeoNode prototype can actually be used. In the first workshop, Test Case One – Hammarkullen, Sweden, the architectural students were presented with an in-depth presentation about data collection, management, metadata as well as GeoNode. This was overwhelming for them, as they were not used to this kind of work. The architectural students are visual and practical learners so once they actually got to using these tools for data collection and over time, they started to feel more comfortable and confident in using them. This fell under a classic scenario in which they needed to play around more with the tools to learn what they did not know or understand so they could ask questions about it. Once the architectural students had completed their analogue data collection, their points were digitised into shapefiles using FieldPapers and QGIS. This process was actually much easier than previously thought. The architectural students placed a sticker on the FieldPaper atlas of where they took their sketches. These stickers were then digitised as points in QGIS, and these points were also later loaded into the GeoNode prototype. This process was repeated for a second time when the architectural students completed another round of

sketches. The architectural students did enjoy seeing their sketches as digital images with a location on a map, this provided a certain real-life connection to their images as their one concern was that making things digital, would result in a loss of depth of information. Unfortunately for these architectural students, the same point had to be repeated over and over again, which was difficult, because different approaches were used to try to explain these same concepts. It took much longer to explain these concepts than what the researcher is used to but reflecting back on this, it can be understood why that is the case. It is because of the target audience. The architectural students were hesitant to learn something new in their final year of study, as they failed to see the bigger picture, and they also wanted detailed instructions covering each step of the process. A recommendation to assist with this would be for the architectural students to attend a module on data collection, management, and analysis in their respective degree programmes. This module does not necessarily need to focus on spatial data, but rather it can be focused on more general data.

5.4.3. The First Test Case

In the first test case of the GeoNode prototype, it was important that the workshop was a success in terms of student participation, and that the GeoNode prototype could be used, did what it was designed to do and did not crash when the architectural students were using it. The GeoNode prototype worked effectively even though it was not moved into production before the architectural students used the prototype. The process of digitising the sketches and creating a points shapefile of locations using FieldPapers and QGIS was completed successfully, without any problems. Those points and the georectified TIFFs were then saved to the GeoNode prototype and loaded in the correct location on a map. All the documents that were saved to the prototype were saved correctly, and without any issues. These documents were then viewable again at a later date. As at one time, all the architectural students, a group of around 13 students, were using the GeoNode prototype at the same time, which unfortunately slowed down the GeoNode prototype, making some of the uploading fail or take a longer time than usual. As most of the architectural students do not have a background in geoinformatics and other forms of desktop GIS, they were slower to follow instructions and struggled to following the correct steps. The architectural students did not follow any standard naming or metadata conventions. They often left the file names as the original name such as "Survey 1", and failed to fill in the metadata, which often rendered this data unusable, as little was known about it. There were also instances where the architectural students upload duplicated data. One problem that was unforeseen before the workshop was EpiCollect5. EpiCollect5 has not been maintained or updated in a long time, therefore, the older and outdated mobile application did not work with newer mobile phones. The original plan was that the architectural students would be required to work with EpiCollect5, but due to this problem,

it was decided that another mobile data collection application would be used, viz. KoBoToolbox. The architectural students were not shown or told about EpiCollect5, in an attempt to avoid confusion and problems with collecting data in the field.

5.4.4. The Second Test Case

Following Test Case One, it was identified that the documentation, text-based and videos, were valuable in helping the architectural students understand the bigger picture. It was therefore decided that a more permanent and cost-effective solution needed to be found in order to maintain and keep the existing documentation available to the architectural students and lecturers. The documentation that was developed for this workshop was hosted on a Wikipedia page, but was later moved to a GitHub page for the purpose of maintenance and longevity. Other benefits of moving to a GitHub page included issue tracking meaning that if the text-based resources became outdated or an error was identified, architectural students could log an issue with the issue tracker in the GitHub page. Another benefit was the change log, which meant that the architectural students could see who has updated the text-based resources, and when it was last updated, so they could get a sense of the relativity of the material. As KoBoToolbox was now being used for mobile data collection, a section about KoBoToolbox was added to the text-based resources so the architectural students could have any questions answered. This statement is relevant as the GeoNode prototype and the framework associated around it for this research are largely dependent on the documentation that was developed. The documentation of the framework (i.e. the data collection, data conversion, and use of the GeoNode prototype) was documented on a GitHub repository that is freely available for the architectural students and lecturers to use as a reference or guide. The code documentation of GeoNode is also well documented in its own GitHub repository, and constantly maintained by a team of dedicated developers and volunteers.

5.4.5. Lessons Learnt

Taking the lessons learnt from the first workshop, the second workshop, Test Case Two – Mamelodi, South Africa, was more successful from an architectural standpoint. One lesson that was learnt during the first workshop was that more than one system administrator was needed during the times of high activity on the GeoNode prototype. As architectural students who were registering to the system needed to be approved before they were granted access, either one of the system administrators was able to do that and ensure there was no extended delay in the learning process for the new architectural students. This system of approving users before they were granted access helped in preserving the integrity of the GeoNode prototype and the data. Approving users was done in the admin interface, which was a Django

product and was very easy and user friendly. All the functions that a system administrator could and would need are neatly organised into categories. For a system administrator, this was a huge benefit, as it shortened the time spent looking for settings and other tools that were needed. The admin interface also allowed for constant monitoring of the GeoNode prototype as a whole, which meant that the system administrators could be proactive about certain situations, rather than reactive, which shortened time that would be spent on problem-solving.

At the end, it can be said that the GeoNode prototype handled both environments with relative ease, as it had managed data from both environments as effectively as both the geospatial and architectural research teams would have wanted. The architectural students started off uncomfortable with the change in the way they had done things, but as time went on, they soon adapted. As modern students in the 21st century, they did adapt at varying paces, some quicker than others. Detailed information is presented in Chapter 4 with regards to how the GeoNode prototype ought to have performed against the requirements in a working environment, but based on observations, it was identified that the GeoNode prototype did actually perform smoothly. There were times that the GeoNode prototype suffered some down time but once it was detected, the GeoNode prototype was quickly rebooted, and none of the data was lost or corrupted. When the GeoNode prototype was in use, it met some requirements better than others, for example, the non-functional requirement of Ease of Use was not extensively visible in these test cases as the architectural students did struggle at first, which goes against the non-functional requirement of Ease of Use. However, it did meet others such as the non-functional requirements of Robustness and Reliability.

Palomino, Muellerklein, and Kelly (2017) notably state that no one single tool fulfils all the requirements of a specific project and that a multitude of tools are often required to accomplish a geospatial task. This supported the motivation for this research to include a data collection tool such as KoBoToolbox and Maptionnaire, and a data conversion tool, QGIS, for the architectural students to assist them with enriching their research. As the GeoNode prototype does not include any data collection or data conversion tools, it was important to include these aspects as architectural students do have a need for these tools. Even though data collection was not listed as a project requirement, it was still deemed relevant as it featured a prominent aspect of the project life cycle for architectural students and their design projects. Data conversion was also deemed relevant as the architectural students often had data in many different formats and the GeoNode prototype was configured to only accept a few of the well-known data formats such as portable document format (.pdf), shapefiles (.shp), Microsoft Word documents (.docx), and Microsoft Excel documents (.xlsx), however, GeoNode can be

configured to accept a wide range of data formats. These tools were not included in the GeoNode prototype, but they were included in the documentation.

5.4.6. Going Forward

Going forward, a sandbox GeoNode is worth further investigation. This GeoNode instance would be purely for educational purposes. In this instance, the architectural students will be allowed to practice and learn as much as they would like. They can upload as much dummy data as desired, name it all incorrectly, upload incomplete metadata, and much more, where the sole purpose of this GeoNode instance would be to teach the architectural students about basic data management. To assist with this learning process, the architectural students can complete a short course on basic data skills to assist them and improve their understanding of data. The role of data skills in geography go beyond the processes of measurements and analysis (Harris, 2018). This is worth further research, as the researchers did observe that architectural students who have been taught how to use a desktop GIS did not struggle when it came to using the GeoNode prototype, and the architectural students who have never worked with a desktop GIS before did struggle, as they were only able to search for data and download it (see also Steiniger *et al.*, 2017). To collect more information about the GeoNode prototype and how the architectural students use it, a user study can be conducted, where the researchers will have the ability to control certain variables such as the data that is being loaded and measure results. This study will allow the researchers to be able to gather results that will be comparable, replicable, and quantifiable. Based on the lessons learnt when designing, implementing and working with the GeoNode prototype, it would be advisable for a bigger research team with more resources to spend more time customising the GeoNode instance. This GeoNode prototype is a vanilla GeoNode instance that can be customized so much more, for example, a simple customization could be allowing GeoJSON files to be uploaded, and where the landing page could be tailored to the project. The most significant suggestion would be spending more time learning how the whole GeoNode instance works from a developer's point of view as there is still lots to be worked with. This would allow for the best version possible to be designed. Having a developer's understanding will allow the developers to enable and disable any extra functions that they may wish to do so such as enabling the monitoring functionality as that was not setup in this GeoNode prototype. Monitoring would allow the users to monitor network traffic and which user is performing which functionalities of the GeoNode prototype. The possibilities are almost endless.

Chapter 6 – Conclusion

6.1. Chapter Overview

This chapter is focused on discussing the results of the GeoNode prototype, how it handled the differing environments and linking it back to the literature that is discussed in Chapter 2. This chapter makes the link between the relevant literature and related work to the work that was completed for this research clear and how the results that were obtained from this study are supported by previous works. The research objectives that were identified in Chapter 1 are discussed in this chapter, as the results that were obtained in regard to each objective are explained.

The work in this dissertation proposed a possible method for architectural students to effectively collect, manage and share their fine-grained data. This final chapter provides both an overview of the main results, and recommendations for future work.

6.2. Summary of Research Objectives achieved

As the aim of this research was to specify the high-level requirements for a solution for managing, organising and sharing of fine-grained data that is gathered in local contexts for educational purposes, and to prototype and evaluate the use of such a solution in architectural design studios relevant literature had to be consulted in order to have an in-depth understanding of the problem at hand, and as well as the many different potential solutions available to solve the problem faced by this research. To achieve the aim of this research, a number of research objectives had to be met, that are discussed below.

6.2.1. Objective 1 – Literature Study, Understanding Current Practices and Evaluation of Possible Tools

Objective 1 was to read and review relevant literature, spend time with the two design studios to become familiar with the processes followed by the architectural design studios at the University of Pretoria and Chalmers University of Technology in their current method of collection, management, and sharing of fine-grained data, as well as to review and evaluate possible tools that might be able to provide an effective solution to this problem.

The results of the literature study are presented in Chapter 2, and cover a wide range of relevant topics, such as software engineering practices, requirements engineering, web geographic information systems (GIS), geoportals and spatial data infrastructures (SDIs),

research data archives (RDAs), virtual research environments (VREs), content management systems (CMS) as well the F.A.I.R data policy. The relevant work section in particular was of importance, because it guided the researcher on what is currently being done to solve similar problems and it was this literature that highlighted the possibility of combining one or more of these topics to solve this problem. The results of the literature study provided a set of desirable characteristics the prototyped solution needed to possess.

The second part of Objective 1 was to spend time with the two design studios so as to become familiar with the processes followed by the architectural design studios at the University of Pretoria and Chalmers University of Technology in their current method of collection, management and sharing of fine-grained data. This was done at the beginning of the first year of research in 2019 and is discussed in the first part of Chapter 3. It was important to understand their current practices, as this provided the researcher with an understanding of what the possible solution needed to be capable of doing. Providing the architectural students with a prototype of something they already used would have not helped to further this research nor their ability to effectively collect, manage and share their fine-grained data.

The third and final aspect of Objective 1 was to review and evaluate possible tools that might be able to provide an effective solution to this problem. This was done in the second part of Chapter 3, and builds a bridge to Objective 2. Two open source tools were evaluated as possible solutions, GeoNetwork, and GeoNode. An evaluation was performed on each tool in terms of its capabilities, its system architecture. As well as how each tool was either capable or not of meeting the functional and non-functional requirements that were identified. After this extensive review, it was determined that GeoNode was the best open source tool for the prototype. Once this was identified, the implementation of the prototype could begin. The literature review also pointed to the selection of GeoNode as the tool as the characteristics were the closest to a content management system, because GeoNode allowed for the upload and management of data in an online collaborative environment.

6.2.2. Objective 2 – Develop the context of use and specify the requirements for managing, organising, and sharing of fine-grained data

Specifying the requirements was an important milestone for this research. By having the requirements specified, this ensured that there was a measurable standard by which to evaluate the GeoNode prototype in order to provide an effective and objective evaluation. The specified requirements are presented in the first half of Chapter 3. In this chapter, ten functional requirements were identified along with 15 non-functional requirements. These requirements were derived from the context in which both design studios from the University

of Pretoria and Chalmers University of Technology collect, manage, and share their fine-grained data. To ensure that these requirements were sufficient and suited to this project, they underwent multiple rounds of iterations and editing. Once these requirements were satisfactory, the development of the prototyped solution could begin.

6.2.3. Objective 3 – Prototype a solution for managing, organizing and sharing of the fine-grained data:

Once Objectives 1 and 2 were completed, the requirements were detailed, and it was understood which open source tool was going to be used to develop the prototype. At this stage, the development of the prototype could begin. As GeoNode was selected to be the prototype, the GeoNode documentation was read so that an understanding of how to best install GeoNode on an Ubuntu server could be determined. Once the GeoNode prototype was successfully installed on the Ubuntu server, and was running without any issues, documentation was developed to assist with the data collection, management, and sharing procedures that the architectural students would follow. Originally, this documentation was accessible on a Wiki page, but later this information was moved to a GitHub page so as to ensure better maintenance and longevity. This documentation covered various topics, such as where to download data from, how to work with KoBoToolbox, QGIS and GeoNode and other useful hints and tips the architectural students may need. Once all of this was completed, the prototype solution and its associated documentation were demonstrated to the architectural lecturers so as to get their thoughts and opinions on how to improve the prototype. This feedback was taken into consideration, and the prototyped solution was improved before use in the two test cases. This is extensively discussed in Chapter 4.

6.2.4. Objective 4 – Evaluation of the prototype:

The evaluation of the prototype was done throughout both the test cases and the results of the evaluation are presented in Chapter 4 and Chapter 5. Based on the evaluation that was conducted, it can be said that the GeoNode prototype handled both test cases, and it met the requirements that needed to be met as they were set out in Chapter 4.

6.2.5. Objective 5 – Draw conclusions and make recommendations:

The conclusions that were drawn from this research are presented in this chapter. The conclusion states that this research was successful in providing the architectural students and lecturers with a working prototype solution for effectively managing, organising and sharing of the fine-grained data that was gathered in local contexts for educational purposes by architectural students for their design studios and projects. The recommendations that are

made from this research include the broadening of the architectural students understanding of the basic concepts of geospatial data in terms of collection, management, analysis, and metadata, as well as a basic understanding of data formats. This can greatly improve their ability to use the GeoNode, and this in turn will improve the effectiveness of the GeoNode prototype and the data within it. The other major recommendation is that the GeoNode prototype's ability to store different data types be extended and the monitoring functionalities be enabled. This can be done during the initial development phase of the GeoNode instance.

6.3. Main results obtained from this research

The aim of this research was to specify the high-level requirements for a solution for managing, organising, and sharing of fine-grained data that was gathered in local contexts for educational purposes, and to prototype and evaluate the use of such a solution in architectural design studios and the results of this are spread out through this dissertation.

The main output from this research was the development of a working GeoNode prototype for the managing, organising and sharing of fine-grained data as well as the documentation gathered on the best practices concerned with data collection, conversion, and the use of the GeoNode prototype. It must be noted that this solution was only a prototype based of a set of requirements that were identified meaning that a better solution might exist that was not explored by the researcher.

The architectural students were able to use the GeoNode prototype. It cannot be stated that they used the GeoNode prototype to the best of its capabilities, as they did not fill in many of the metadata fields, and left the file name as the default file name, which had little to no meaning, but they did use it to the best of their own personal and professional capability. After being trained in how best to use the GeoNode prototype, the architectural students were more capable in using it and it can be seen that there is an improvement in use from Test Case One to Test Case Two. It must also be stated that the GeoNode prototype has been used since the end of the Test Case Two, and the patterns of use that were identified in these Test Cases are still evident while finishing this dissertation.

All five of the research objectives were met throughout the duration of this research and a working GeoNode prototype was developed, along with the framework around it. This framework was so effective that the architectural students went on to write their version of the documentation. The architectural students are visual learners, and they turned all the text-

based documentation into pictures and graphics as they learn and understand better from visual sources, as indicated by the student feedback.

6.4. Recommendations for future work

As this was only a prototype of what is needed to suit the requirements of the architectural students and lecturers, there is the potential to expand on this by turning this prototype into a concrete tool that is going to be used for years, by either leaving the GeoNode in its current state, or by configuring and customising it further.

A case of further work for this research would be to see how effectively the GeoNode prototype is performing in a few years' time, and whether it is still meeting the functional and non-functional requirements that were set out by this research. One can even go as far as to evaluate whether the functional and non-functional requirements are still relevant to the two design studios at the University of Pretoria and Chalmers University of Technology. This can be done by means of a user study.

To ensure that the GeoNode prototype is used to the best of its capabilities, it would be important to broaden the architectural students' understanding of the basic concepts of geospatial data. The basics being the collection, management, analysis, and metadata as well as an understanding of data formats. This can greatly improve their ability to use the GeoNode, which will in turn improve the effectiveness of the GeoNode prototype and the data within it. It could also be the case that this basic understanding does not have to be spatial in nature, but of a more general sense of research data. To achieve this, the architectural students can be asked to take a course on basic data management, or a course is incorporated into one of their final year modules. They are required to take and pass this module before they are admitted into their honours or Master's programmes, where they are required to use the GeoNode prototype.

Another major recommendation is that the GeoNode prototype's ability to store different data types be extended and the monitoring functionalities be enabled. By extending the GeoNode prototype's ability to handle many different data formats, spatial or non-spatial, the ability to manage any data that is collected by the architectural students is greatly augmented. Enabling the monitoring functionalities of GeoNode affords the architectural lecturers the ability to monitor all the activity within the prototype. They will be able to see the system's health status in terms of error checking, any maintenance that needs to be performed and the memory status. They are also afforded an overview of GeoNode, and are able to monitor the respective

activities of each architectural student, whether it be work-related, or indeed nefarious in nature. All of this can be done during the initial development phase of the GeoNode instance. Other recommendations can be found in the Going Forward in the Discussion of Results detailed in Chapter 5.

References

1. Abbott, J. and Douglas, D., 2003. The use of longitudinal spatial analyses of informal settlements in urban development planning. *Development Southern Africa*, 20(1), pp.3-19.
2. Adelfio, M., Kain, J.H., Stenberg, J. and Thuvander, L., 2019. GISualization: visualized integration of multiple types of data for knowledge co-production. *Geografisk Tidsskrift-Danish Journal of Geography*, 119(2), pp.163-184.
3. Agrawal, S. and Gupta, R.D., 2017. Web GIS and its architecture: a review. *Arabian Journal of Geosciences*, 10(23), p.518.
4. Alesheikh, A.A., Helali, H. and Behroz, H.A., 2002, July. Web GIS: technologies and its applications. In *Symposium on geospatial theory, processing and applications* (Vol. 15).
5. Almarashdeh, I., 2016. Sharing instructors experience of learning management system: A technology perspective of user satisfaction in distance learning course. *Computers in Human Behavior*, 63, pp.249-255.
6. Alvarez Sainz, M., Ferrero, A. M. and Ugidos, A., 2019. Time management: skills to learn and put into practice, *Education and Training*, 61(5), pp. 635–648.
7. Aly, A.G. and Labib, N.M., 2013. Proposed Model of GIS-Based Cloud Computing Architecture for Emergency System. *International Journal Of Computer Science*, 1(4), pp.17-28.
8. Assante, M., Candela, L., Castelli, D., Coro, G., Lelii, L. and Pagano, P., 2016. Virtual research environments as-a-service by gCube. *PeerJ Preprints*, 4, p.e2511v1.
9. Barker, D., 2016. *Web content management: Systems, features, and best practices*. " O'Reilly Media, Inc."
10. Barbera, E., Gros, B. and Kirschner, P., 2015. Paradox of time in research on educational technology, *Time & Society*, 24(1), pp. 96–108.

11. Barker, M., Olabarriaga, S.D., Wilkins-Diehr, N., Gesing, S., Katz, D.S., Shahand, S., Henwood, S., Glatard, T., Jeffery, K., Corrie, B. and Treloar, A., 2019. The global impact of science gateways, virtual research environments and virtual laboratories. *Future Generation Computer Systems*, 95, pp.240-248.
12. Beatty, B. and Ulasewicz, C., 2006. Faculty perspectives on moving from Blackboard to the Moodle learning management system. *TechTrends*, 50(4), pp.36-45. use one in MSc
13. Bhat, M.A., Shah, R.M. and Ahmad, B., 2011. Cloud Computing: A solution to Geographical Information Systems (GIS). *International Journal on Computer Science and Engineering*, 3(2), pp.594-600.
14. Blower, J.D., 2010, June. GIS in the cloud: implementing a Web Map Service on Google App Engine. In *Proceedings of the 1st International Conference and Exhibition on Computing for Geospatial Research & Application* (pp. 1-4).
15. Bolstad, P., 2005. *GIS fundamentals: A first text on geographic information systems*. Eider (PressMinnesota).
16. Borgman, C.L., 2012. The conundrum of sharing research data. *Journal of the American Society for Information Science and Technology*, 63(6), pp.1059-1078.
17. Boulos, M.N.K., Warren, J., Gong, J. and Yue, P., 2010. Web GIS in practice VIII: HTML5 and the canvas element for interactive online mapping. *International Journal of Health Geographics*, 9, pp. 1–13.
18. Brown, M., Dehoney, J. and Millichap, N., 2015. The next generation digital learning environment. *A Report on Research. ELI Paper. Louisville, CO: Educause April*, 5(1), pp.1-13.
19. Cabot, J., 2018. WordPress: a content management system to Democratize publishing. *IEEE Software*, 35(3), pp.89-92.
20. Candela, L., Castelli, D. and Pagano, P., 2011. History, evolution, and impact of digital libraries. In *Organizational Learning and Knowledge: Concepts, Methodologies, Tools and Applications* (pp. 837-866). IGI Global.

21. Candela, L., Castelli, D. and Pagano, P., 2013. Virtual research environments: an overview and a research agenda. *Data Science Journal*, pp.GRDI-013.
22. Reimer, T.F. and Carusi, A., 2010. Virtual research environment collaborative landscape study.).
23. Coetzee, S., Steiniger, S., Köbben, B., Iwaniak, A., Kaczmarek, I., Rapant, P., Cooper, A., Behr, F.J., Schoof, G., Katumba, S. and Vatseva, R., 2017, July. The Academic SDI—Towards understanding spatial data infrastructures for research and education. In *International Cartographic Conference* (pp. 99-113). Springer, Cham.
24. Corti, P., Bartoli, F., Fabiani, A., Giovando, C., Kralidis, A.T. and Tzotsos, A., 2019. *GeoNode: an open source framework to build spatial data infrastructures* (No. e27534v1). PeerJ Preprints.
25. Dareshiri, S., Farnaghi, M. and Sahelgozin, M., 2019. A recommender geoportal for geospatial resource discovery and recommendation. *Journal of Spatial Science*, 64(1), pp.49-71.
26. van Deventer, M., UP, H. and Morris, J., 2009, May. Virtual research environments: learning gained from a situation and needs analysis for malaria researchers. In *African Digital Scholarship and Curation Conference* (pp. 12-14).
27. Van Deventer, M. and Pienaar, H., 2015. Research data management in a developing country: a personal journey. *International Journal of Digital Curation*, 10(2), pp. 33–47.
28. Devillers, R., Bédard, Y. and Jeansoulin, R., 2005. Multidimensional management of geospatial data quality information for its dynamic use within GIS. *Photogrammetric Engineering & Remote Sensing*, 71(2), pp.205-215.
29. Doorn, P. and Tjalsma, H., 2007. Introduction: archiving research data. *Archival science*, 7(1), pp.1-20.
30. Dunn, S., 2009. Dealing with the complexity deluge: VREs in the arts and humanities. *Library Hi Tech*, 27(2), pp.205-216.
31. Van Eck, E., Renkin, W. and Ntakirutimana, E., 2016. The parable of the Feast (Lk 14:

- 16b–23): Breaking down boundaries and discerning a theological–spatial justice agenda. *HTS: Theological Studies*, 72(1), pp.1-8.
32. Van Eck, E., Renkin, W. and Ntakirutimana, E., 2016. The parable of the Feast: Breaking down boundaries and discerning a theological–spatial justice agenda.
33. Elovaara, P. and Mörtberg, C., 2007. Design of digital democracies: performances of citizenship, gender and IT. *Information, Community and Society*, 10(3), pp.404-423.
34. Fraser, M., 2005. Virtual research environments: overview and activity. *Ariadne*, (44).
35. Fu, P. and Sun, J., 2011. GIS in the Web Era. *Web GIS: Principles and applications*, pp.2-24.
36. Gautreau, C., 2011. Motivational factors affecting the integration of a learning management system by faculty. *Journal of Educators Online*, 8(1), pp. 1–25.
37. Goodchild, M.F., Fu, P. and Rich, P., 2007. Sharing geographic information: an assessment of the Geospatial One-Stop. *Annals of the Association of American Geographers*, 97(2), pp.250-266.
38. Gordov, E.P., Okladnikov, I.G., Titov, A.G. and Fazliev, A.Z., 2016. Some aspects of development of virtual research environment for analysis of climate change consequences. *CEUR Workshop Proceedings*, 1752, pp.291-297.
39. Gothenburg City Management Office (2020) *Göteborgsbladet 2020, Statistics and Analysis*. Available at: <https://goteborg.se/wps/portal/enhetssida/statistik-och-analys/goteborgsbladet/hamta-statistik/faktablad/goteborgsbladet> (Accessed: 23 September 2020).
40. Grunzke, R., Hartmann, V., Jejkal, T., Prabhune, A., Herold, H., Deicke, A., Hoffmann, A., Schrade, T., Meinel, G., Herres-Pawlis, S. and Stotzka, R., 2016, June. Towards a Metadata-driven Multi-Community Research Data Management Service. *CEUR Workshop Proceedings*, 1871(June), pp. 8–10.
41. Guadamuz, A., 2009. Free and open-source software. *Law and the Internet*, 3rd Ed., L. Edwards, C. Waelde, eds., Oxford: Hart.

42. Gunay, A., Akcay, O. and Altan, M.O., 2014. Building a semantic based public transportation geoportal compliant with the INSPIRE transport network data theme. *Earth Science Informatics*, 7(1), pp.25-37.
43. Guo, H., Liu, Z., Jiang, H., Wang, C., Liu, J. and Liang, D., 2017. Big Earth Data: a new challenge and opportunity for Digital Earth's development. *International Journal of Digital Earth*, 10(1), pp.1-12.
44. Hall, T. and Vidén, S., 2005. The Million Homes Programme: a Review of the great Swedish planning project. *Planning Perspectives*, 20(3), pp.301-328.
45. Harris, R., 2018. From data to knowledge: teaching data skills in geography. *Geography*, 103(1), pp.12-18.
46. Haynes, M., Coetzee, S.M. and Rautenbach, V., 2019. Supporting urban design with open source geospatial technologies. *International Society for Photogrammetry and Remote Sensing*.
47. Helmi, A.M., Farhan, M.S. and Nasr, M.M., 2018. A framework for integrating geospatial information systems and hybrid cloud computing. *Computers & Electrical Engineering*, 67, pp.145-158.
48. Hevner, A.R., March, S.T., Park, J., Ram, S. and Ram, S., 2004. Research essay design science in information. *MIS Q*, 28(1), pp.75-105.
49. Iwanaik, A., Kaczmarek, I., Kubik, T., Lukowicz, J., Paluszynski, W., Kourie, D., Cooper, A.K. and Coetzee, S., 2011. An intelligent geoportal for spatial planning. 25th International Cartographic Conference, Paris, 4-8 July 2011.
50. Jiang, H., van Genderen, J., Mazzetti, P., Koo, H. and Chen, M., 2019. Current status and future directions of geoportals. *International Journal of Digital Earth*, pp.1-22.
51. Karabegovic, A. and Ponjavic, M., 2012, September. Geoportal as decision support system with spatial data warehouse. In *2012 Federated Conference on Computer Science and Information Systems (FedCSIS)* (pp. 915-918). IEEE.
52. Karnatak, H.C., Saran, S., Bhatia, K. and Roy, P.S., 2007. Multicriteria spatial decision

- analysis in web GIS environment. *Geoinformatica*, 11(4), pp.407-429.
53. Kinder-Kurlanda, K., Weller, K., Zenk-Möltgen, W., Pfeffer, J. and Morstatter, F., 2017. Archiving information from geotagged tweets to promote reproducibility and comparability in social media research. *Big Data & Society*, 4(2), pp.1-14.
54. Kraak, M.J., 2004. The role of the map in a Web-GIS environment. *Journal of Geographical Systems*, 6(2), pp.83-93.
55. Laster, S., 2010. Model-driven design: Systematically building integrated blended learning experiences. *Journal of Asynchronous Learning Networks*, 14(1), pp.39-53.
56. Maguire, D.J. and Longley, P.A., 2005. The emergence of geoportals and their role in spatial data infrastructures. *Computers, Environment and Urban Systems*, 29(1), pp.3-14.
57. Martinez-Caro, J.M., Aledo-Hernandez, A.J., Guillen-Perez, A., Sanchez-Iborra, R. and Cano, M.D., 2018. A comparative study of web content management systems. *Information (Switzerland)*, 9(2), p.27.
58. McCarthy, J.D. and Graniero, P.A., 2006. A GIS-based borehole data management and 3D visualization system. *Computers & Geosciences*, 32(10), pp.1699-1708.
59. Mehdi, S.A., Ali, M., Nima, G., Zahra, R., Reyhaneh, S. and Peyman, B., 2014. How to implement a governmental open source geoportal. *Journal of Geographic Information System*, 2014
60. Mell, P.M. and Grance, T. (2018). The NIST Definition of Cloud Computing. [online] Special Publication (NIST SP) - 800-145. Available at: <https://www.nist.gov/publications/nist-definition-cloud-computing>.
61. Mothibi, N.H. and Malebana, M.J., 2019. Determinants of entrepreneurial intentions of secondary school learners in Mamelodi, South Africa. *Academy of Entrepreneurship Journal*, 25(2), pp.1-14.
62. Mullinix, B. and McCurry, D., 2003. Balancing the learning equation: Exploring effective mixtures of technology, teaching and learning. *The Technology Source*.

63. Ng'eno, E. and Mutula, S., 2018. Research Data Management (RDM) in agricultural research institutes: a literature review. *Inkanyiso: Journal of Humanities and Social Sciences*, 10(1), pp.28-50.
64. Ožana, R. and Horáková, B., 2008. Actual State in developing GeoNetwork opensource and metadata network standardization. *GIS Ostrava 2008*.
65. Leme, L.A.P.P., Brauner, D.F., Casanova, M.A. and Breitman, K., 2007. A Software Architecture for Automated Geographic Metadata Annotation Generation. *Proc. XXII Simpósio Brasileiro De Banco De Dados, SBBD, João Pessoa, Brazil*.
66. Palomino, J., Muellerklein, O.C. and Kelly, M., 2017. A review of the emergent ecosystem of collaborative geospatial tools for addressing environmental challenges. *Computers, Environment and Urban Systems*, 65, pp.79-92.
67. Patel, S.K., Rathod, V.R. and Parikh, S., 2011, December. Joomla, Drupal and WordPress-a statistical comparison of open source CMS. In *3rd International Conference on Trendz in Information Sciences & Computing (TISC2011)*(pp. 182-187). IEEE.
68. Pittinsky, M., 2004. The networked learning environment. *Stepping Beyond Courses to a More Expansive Online Learning Experience* (October), pp. 2–4.
69. Powell, T.A., Jones, D.L. and Cutts, D.C., 1998. *Web site engineering: beyond Web page design*. Prentice-Hall, Inc..
70. Pressman, R.S., 2005. *Software engineering: a practitioner's approach*. Palgrave macmillan.
71. Rachel, V. and Parthasarathy, M., 2016. Learning management system using open source moodle for computer science students in higher educational institute. *International Journal of Computer Science & Engineering Technology (IJCSET)*, 7(1), pp.13-18.
72. Rautenbach, V., Coetzee, S. and Iwaniak, A., 2013. Orchestrating OGC web services to produce thematic maps in a spatial information infrastructure. *Computers*,

Environment and Urban Systems, 37, pp.107-120.

73. Rhode, J., Richter, S., Gowen, P., Miller, T. and Wills, C., 2017. Understanding faculty use of the learning management system. *Online Learning*, 21(3), pp.68-86.
74. Ribeiro, C., da Silva, J.R., Castro, J.A., Amorim, R.C., Lopes, J.C. and David, G., 2018. Research Data Management Tools and Workflows: Experimental Work at the University of Porto. *IASSIST Quarterly*, 42(2), pp.1-16.
75. De Roure, D. and Goble, C., 2007. myExperiment – A Web 2.0 Virtual Research Environment. *International Workshop on Virtual Research Environments and Collaborative Work Environments, Edinburgh, UK*, pp. 1–4.
76. De Roure, D., Goble, C. and Stevens, R., 2007, December. Designing the myExperiment virtual research environment for the social sharing of workflows. In *Third IEEE International Conference on e-Science and Grid Computing (e-Science 2007)* (pp. 603-610). IEEE.
77. Simpson, M., Payne, F., Munro, R. and Hughes, S., 1999. Using Information and Communications Technology as a Pedagogical Tool: who educates the educators?. *Journal of Education for Teaching*, 25(3), pp.247-262.
78. Sommerville, I., 2011. Software engineering 9th Edition. *ISBN-10, 137035152*, p.18.
79. Steiniger, S., De La Fuente, H., Fuentes, C., Barton, J. and Muñoz, J.C., 2017. Building a geographic data repository for urban research with free software—learning from Observatorio. CEDEUS. cl. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences – ISPRS Archives*, 42(4W2), 42, p.147.
80. Tait, M.G., 2005. Implementing geoportals: applications of distributed GIS. *Computers, Environment and Urban Systems*, 29(1), pp.33-47.
81. Thanos, C., 2013. A vision for global research data infrastructures. *Data Science Journal*, 12, pp.71-90.
82. Unsworth, J., 2006. *Our Cultural Commonwealth: the report of the American Council of learned societies commission on cyberinfrastructure for the humanities and social*

sciences. ACLS: New York.

83. Wang, C.Y., 2002. Walk a Mile in Students' Shoes—An Approach to Faculty Development on Integrating Web-Based Collaborative Learning into Instruction. *Computer support for Collaborative learning: Foundations for a CSCL community*, pp. 639–640.
84. Wilkinson, M.D., Dumontier, M., Aalbersberg, I.J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.W., da Silva Santos, L.B., Bourne, P.E. and Bouwman, J., 2016. The FAIR Guiding Principles for scientific data management and stewardship. *Scientific data*, 3(1), pp.1-9.
85. Wright, B., Brunner, O. and Nebel, B., 2018, January. On the Importance of a Research Data Archive. *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*, pp. 7941–7946.
86. W3techs.com. (2009). W3Techs - extensive and reliable web technology surveys. [online] Available at: <https://w3techs.com/>.
87. Van Wyk, J., Bothma, T. and Holmner, M., 2020. A conceptual virtual research environment model for the management of research data, a South African perspective. *Library Management*. pp. 417–446.
88. Van Wyk, J. and Van der Walt, I., 2014. Going full circle: research data management@ University of Pretoria. eResearch Africa 2014.
89. Yamashkin, S.A., Radovanović, M.M., Yamashkin, A.A., Barmin, A.N., Zanozin, V.V. and Petrović, M.D., 2019. Problems of designing geoportal interfaces. *GeoJournal of Tourism and Geosites*, 24(1), pp.88-101.
90. Yang, Z.L., Cao, J., Hu, K., Gui, Z.P., Wu, H.Y. and You, L., 2016. Developing a cloud-based online geospatial information sharing and geoprocessing platform to facilitate collaborative education and research. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences – ISPRS Archives*, 41, p.3-7.

Appendix A



UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Faculty of Natural and Agricultural Sciences
Ethics Committee

E-mail: ethics.nas@up.ac.za

13 July 2020

ETHICS SUBMISSION: EXTENSION LETTER

Dr V Rautenbach
Department of Geography Geoinformatics and Meteorology
Faculty of Natural and Agricultural Science
University of Pretoria

Reference number: NAS128/2019

Project title: Producing spatial knowledge with the community to empower them

Dear Dr V Rautenbach,

We are pleased to inform your application for ethics extension conforms to the requirements of the Faculty of Natural and Agricultural Sciences Research Ethics committee.

Please note the following about your ethics approval:

- Please use your reference number (NAS128/2019) on any documents or correspondence with the Research Ethics Committee regarding your research.
- Please note that the Research Ethics Committee may ask further questions, seek additional information, require further modification, monitor the conduct of your research, or suspend or withdraw ethics approval.
- Please note that ethical approval is granted for the duration of the research (e.g. Honours studies: 1 year, Masters studies: two years, and PhD studies: three years) and should be extended when the approval period lapses.
- The digital archiving of data is a requirement of the University of Pretoria. The data should be accessible in the event of an enquiry or further analysis of the data.

Ethics approval is subject to the following:

- The ethics approval is conditional on the research being conducted as stipulated by the details of all documents submitted to the Committee. In the event that a further need arises to change who the investigators are, the methods or any other aspect, such changes must be submitted as an Amendment for approval by the Committee.

Post approval submissions including application for ethics extension and amendments to the approved application should be submitted online via the ethics work centre.

We wish you the best with your research.

Yours sincerely,

A handwritten signature in blue ink, appearing to read 'Z. J. J. J.', on a light blue background.

Chairperson: NAS Ethics Committee

Appendix B

In the appendix below, the entire set of requirements from the first iteration are presented. These requirements were drawn up during the workshop in early March 2019 in Hammarkullen, Sweden. These requirements were drawn up by the architectural lectures with the help of a geospatial researcher.

Methodological framework/protocol

repository / platform / framework / tool / portal / interactive library / digital archive

- library of neighborhood level data
- participatory library
- collaborative library for fine-grained geospatial data
- co-created library

Commonalities

- Process of data collection
 - o field work: transect walks, implies data is captured in the field (ease of use!/mobile apps, sketches, be able to use tool that suits them, maptionnaire, interviews/conversations, Comapper, FieldPapers, paper maps, flexibility!)
 - o joint mapping: on (paper) maps, e.g. smartie/sticky note maps (physical and perceptions), implies it sometimes has to happen on paper; many stakeholders involved (one 'scribe'?); iterative process of improving one's understanding...
- Data
 - o points (e.g. at this point the interview took place / photo was taken)
 - o line (e.g. the hawkers walk along this route from the station)
 - o polygon (e.g. this area is safe / is used by hawkers) (annotations)
 - o raster for backdrops (e.g. aerial/satellite image, temperature grid)
 - o Complex features/objects (e.g. many points together make a single feature, donuts)
- Data management
 - o Metadata: Where and when did I capture this data? Who was the source? How and why did I capture this data? What did I capture? ...
- Data collaboration / sharing
 - o Students work in groups (group members can view/edit to each other's data)
 - o Lecturer / tutor has overview of all the data
 - o System administrator can do everything

- External stakeholders should get access to specific/subsets of data

Functional requirements

- Data management and viewing, NOT spatial analysis, NOT map design, NOT data collection
- Data
 - Backdrop added to map through web service
 - e.g. OpenStreetMap (OSM) basemap data → teachosm.org
 - e.g. City of Tshwane...
 - Observations as points, lines, and polygons, complex features; attributes could be anything: text, PDF, voice notes, ...
 - stored in PostGIS on a server
- Data collection and editing
 - Basemap data collection and editing with OSM idEditor and other OSM tools
 - Collect data with other tools (FieldPapers, EpiCollect, SmartPhone, etc.), then upload it
- Data management
 - Data is stored in the 'cloud' (on a server)
 - Upload data collected in the field
 - Organize data into folders
 - Group data into identifiable units (datasets)
 - Filter and search for data (based on spatial extent, attributes, metadata)
 - View some parts of data offline for field work (decide on which data/functionality)
- Data viewing
 - Switch layers on and off (OSM is one of the layers)
 - Change styling of layers (non-OSM only?)
 - Zoom, pan, etc.
 - Info button (click on feature to display attributes and metadata)
- Exchanging data
 - Export selected / filtered data as SHP file
 - Export current view as PDF
 - Export legend for current view as PDF
 - Access data via WFS / WMS

Non-functional requirements

- 4 types of users (see above): student, lecturer/tutor, system administrator, external stakeholder
- Security based on AAA
 - o Authentication: is it really you?
 - o Authorization: what are you allowed to do after login?
 - o Accounting: monitoring and recording what you do (for longitudinal studies)
- Workable with TUKS wifi and 3G network (NOT fibre, NOT LAN)
- Help manuals and tutorials
- Scalability: max 100 groups and 200 users simultaneously
- Reliability: 100%, except downtime for regular maintenance once a month

*Review mapable and GISlike and ArcGIS Online do not duplicate existing functionality
Differentiate our work from geoportal and research data (RDA, Australia, data preservation, ...)
and Open Science platform (spatial viewing in our research!)*

Knowledge and skills required by students

- Geospatial data: datasets, features, layers
- Metadata
- Projections, measurements
- Data collection tools (EpiCollect, OSM idEditor, FieldPapers...)
- Data privacy and ethics issues
- Map preparation tools (QGIS, ArcGIS, mapable ...)
 - o Information communication principles
 - o Adobe Photoshop skills to refine and produce the final map products

Scenario description

The lecturer decides on a study area for the module.

Open module. The lecturer demarcates the study area for a particular year. Students are divided into groups. The system administrator creates logins for each student, assigning them to their groups.

Lecturer gives students access to relevant datasets; this could include data from previous years.

Students' login for the first time and view the available data in the study area. The first transect walk could be before or after this first login. The students review the study area (zooming and panning) to better understand the study area spatially.

Students download data, prepare maps, 3D models or whatever else is needed (questionnaires) in other tools to prepare for the fieldwork.

Students go into the field to collect data with other tools.

Subsequently, the data is uploaded (if digital) or digitized (e.g. from sticky notes).

Metadata is added - Dublin Core and a bit more?

Download data for design and planning for use in other tools.

Close this module.

Three test cases:

1. Masters at Chalmers in Hammarkullen (includes 2019 UP honours students) - Sept 2019
2. Honours at UP in Mamelodi - Feb 2020
3. Masters at UP in Mamelodi (includes Chalmers Masters students) - Feb 2020

Appendix C

In Appendix C below, the full set of requirements from the second iteration are presented. These requirements were drawn up based off the requirements in iteration one and the knowledge gained during the first workshop in early March 2019.

Functional Requirements:

Concept	Description	Long/Short Term	Lecturer/Student
Data Storage of non-geographical data	The system should be able to store various kinds of data formats such as voice notes, images, .pdf documents, text files that are non-geographic in nature (but can be linked to a location via the metadata) and uploaded by the students.	Short	Students
Data Storage of geographical data	The system should be able to store geographic data formats such as vector and raster files that are uploaded by the students.	Short	Student
Search Functionalities	The system will allow users such as the students and lecturers to search the metadata of the data for specific attributes such as date collected, who collected it, what extent it was collected at and any other attributes that were collected.	Short	Student/Lecturer
Online Capabilities	Online capabilities allow for working partners from to collaborate. Basically the cloud.	Long	Student/Lecturer
Filtering the data	Users must be able to filter the data according to an attribute of their choosing i.e. the user wants to view all the photos that are stored in the system and then they can use the search feature to filter the data even further.	Short/Long	Student/Lecturer

Map Production	The users of the system must be able to export the data of their choosing into a simple map that can be saved as a PDF document for later photoshopping.	Short	Student
Groups	The users of the system will be working in project groups and each member of the group will need access to the data of their group and not be allowed to view the data of the other groups.	Short	Student
Login Types	The users of the system should have different rights, roles and privileges. Students can only view, add data and have access to their group's data, lecturers can only view, add data and have access to all the groups data while the system administrator has rights to add, view and delete all the data that is stored in the system, they maintain and monitor the system.	Short	Student/Lecturer
Viewing	The users of the system should be able to view the data in a simple manner with an OSM base map.	Short	Student
Metadata	The users of the system must be able to add metadata to the system. Metadata that can be added include, who collected the data, what purpose the data was collected for, at what scale the data was collected and so forth.	Short	Student

Non-functional Requirements:

Concept	Description	Long/Short Term	Lecturer/Student
Scalability	Scalability is the ability of the system to process or handle the growing amount of work or users, its potential to accommodate the growth.	Long	Lecturer/Student
Data Integrity	The maintenance of the accuracy and consistency of the data over its entire life-cycle is very important for a working system.	Short	Lecturer/Student
Documentation of use	Clear and precise documents on the correct management, use and maintenance of the system.	Short	Lecturer/Student
Documentation of code	Clear and precise documents detailing the development process and the code that was used to develop the system so that it can be open-source and understood by system admins.	Long	Lecturer/Student
Speed	The system should be fast. The processed transactions per second should be high, the user/event response time should be as little as possible.	Long	Lecturer/Student
Ease of Use	The system should be easy to use for users, especially the users that have a non-GIS background. The functions and capabilities will not be GIS intensive in a way that only working GIS professionals can use the system.	Long	Lecturer/Student
Reliability	The system should experience as little down time as possible while	Long	Lecturer/Student

	keeping the rate of failure as low as possible.		
Robustness	The time for the system to restart after a failure/reboot, probability of data corruption on failure.	Long	Lecturer/Student
Privacy	Data privacy is important as personal questions and personal data will be collected. The personal data that is collected should not be visible to those who do not or should not have access to it.	Long	Lecturer/Student
Security	The system should be secure and safe as personal data and information will be collected and stored. No personal data leaks as this can cause ethical issues for the researchers.		