Bo Zhou. The Design and Development of a New Interactive Publication Management System for Renaissance Computing Institute: Project Report. A Master's Paper for the M.S. in I.S.degree. May, 2020. 37 pages. Advisor: Ryan Shaw.

This paper design and develops a interactive Publication Management System for Renaissance Computing Institute at the University of North Carolina at Chapel Hill. This system supports allows users to insert publication by only providing the DOI information, search publication stored in the database with up to 5 metrics.

Renaissance Computing Institute develops and deploys advanced technologies to enable research discoveries and practical innovations. Publication Management System would play an important role in making full use of the existing publications, as well as keeping and managing publications.

Headings:

Web Development

Web Design

Data Visualization

THE DESIGN AND DEVELOPMENT OF A NEW INTERACTIVE PUBLICAITION MANAGEMENT SYSTEM FOR RENAISSANCE COMPUTING INSTITUTE: PROJECT REPORT

by
Bo Zhou

A Master's paper submitted to the faculty
of the School of Information and Library Science
of the University of North Carolina at Chapel Hill
in partial fulfillment of the requirements
for the degree of Master of Science in
Information Science.

Chapel Hill, North Carolina

May 2020

Approved by

Ryan Shaw

Table of Contents

Section 1. Introduction

1.1 Background

Publication management is a relatively time-consuming and sophisticated procedure. Even though we can make it easier by implementing software or applications, potential risks exist when unexpected publication breaks the system, leading the server down and the entire system encounter privacy issues. Interactive web design is a good way to reduce potential risks and improve usability. Changes in the database or server should be shown at user's interface interactively, instead of reloading the entire web page. On the other hand, any updates from the client side should be processed by backend server as well.

Renaissance Computing Institute (RENCI) is an institute of the University of North Carolina at Chapel Hill, aiming to develop and deploy advanced technologies to enable research discoveries and practical innovations[1]. Researchers at RENCI publish numerous publications every year and are involved in different research team. How to collect various types of paper and make full use of these data as research goes on is a big concern to RENCI. Currently, the papers are all stored through RENCI's homepage, which can be inserted through a form, allowing users to fill in different attributes, such as title, author, category, published date, DOI, etc… These operations seem easy to use and are user-friendly, however, when the scale becomes larger and larger, these steps would be extremely time-consuming and even impossible to complete. Therefore, an interactive automation process should be in place to manage publications more efficiently and further enable advanced function to make full use of these publications.

As a web developer intern at RENCI, I mainly responsible for the design and development of RENCI Publication Management System to help them improve efficiency and make full use of existing publications. At higher level, the RENCI Publication Management System is built using the MERN (MongoDB, Express, React, Node.js) stack. The user interface is created using React JavaScript Framework, which is responsive thus users can have more interactions without even refreshing the page. MongoDB is chosen as a database system since the publication is more document-oriented instead of having relationships.

1.2 Objectives

The main goal of this paper is to provide an overview of the entire process of RENCI Publication Management System design and development in a real-world environment, including gathering requirements, system design and analysis, implementation, deployment and evaluation. As for requirement gathering, I will meet with both project managers and researchers at RENCI, understanding their demands and concerns. Since the Publication Management System would be integrated into RENCI's current website in the future, it is important for me to generate a design plan that would fit into RENCI's overall design. Moreover, the development of Publication Management System should not only realize required CRUD (create, read, update, delete) functions, but also be able to handle large data sets interactively and prevent unexpected error. Usability testing would be collected in the final stage and an analysis will be generated to reflect overall result. Therefore, this paper can provide suggestions of Publication Management System development and design for future project managers and web browser developers.

1.3 Research Questions

This research is conducted based on the following research questions:

RQ1: What are the advantages and disadvantages of NoSQL database, compared with relational database?

Relational and NoSQL database are often compared with each other in different development settings. Therefore, the research will dig into current area of interest, publication management system, and compare both strengths and weaknesses of each other.

RQ2: How are users interacting with the system, what information do users need?

Potential users at RENCI have various roles, from researchers to project managers. To understand how well our system is being used by users and make sure users get the information they want, the research will collect user's feedback individually and generate an analysis report to answer this research question.

RQ3: Does the usability testing results meet the user's needs?

A proposed plan for usability testing of the RENCI Publication Management System, including goals and tasks, will be beneficial for this research to understand the overall user's experience when using this tool and polish this system to better meet the user's demand.

1.4 Definition of Terms

Relational Database (RDBMS) refers to a database that stores data in a structured format, using rows and columns. The relational structure makes it possible to run queries across multiple tables at once.

NoSQL Database includes simplicity of design, simpler horizontal scaling to clusters of machines and finer control over availability. The data structures used by NoSQL databases are different from those used by default in relational databases which makes some operations faster in NoSQL database software.

Section 2. Literature Review

Introduction

In this section, literature that is related to Publication Management System and web development process is reviewed. Much emphasis has been focused on system design and evaluation, which are the backbones and guidelines for this project.

2.1 The Practices of Publication Management System

Publication Management System is not the only term that is used to describe the software that manages research paper. If we separate them into different purposes, there are mainly three types: Bibliography Management Systems for recording and utilizing bibliography citations, various online platforms for researchers to locate scientific literature and enterprise content management systems created to manage critical information for organizations (Naak, 2008).

2.1.1 Bibliography Management Systems

Bibliography management systems are often referred as citation management tool, used by many researchers and authors. Many existing platforms such as BibTeX and EndNote, allowing users to store citations which later could generate bibliographies, like a list of citation references. The flow of completing an article generation with XBMS can be separated into 4 phases, which are editing phase, validating phase, associating phase and transformation phase.

However, the increase in the size of collections has made bibliography management one of the most frustrating tasks that researchers have encountered (Naak, 2008, p2). Therefore, automation processes are essential if we are going to build modern comprehensive Publication Management System.

2.1.2 Online resources for locating Scientific Literature

Among various online resources, *TechLens* combines Collaborative and Content-Based recommendation functions to help user better locate scientific literature. In addition, Nishikant has made a way to improve the TechLens workflow by using user’s personal citation libraries to enhance his profile and better provide recommendations (Nishikant, 2007). Tang also propose a recommender system which takes pedagogical aspects into consideration (Tang, 2009).

Therefore, personal profile seems like a good way for online publication management system to either provide accurate results or make the workflow simpler.

2.1.3 Enterprise Content Management Systems

One of the deficiencies in business software system development is that developed system can’t change adaptively when the business upper management pattern change (Yu & Li, 2006). Enterprise Content Management (ECM) systems are designed to facilitate the storage, sharing, management and retrieval of critical information and processes for organizations, teams and individuals. It is necessary to have always available information on who did what, how and in cooperation with whom, when processing a task (Simonova & Hudec, 2015). Systems like LiveLink is designed to manage context and relative knowledge in an institutional context, as well as some other advanced functions like archiving, collaboration and digital asset management. These supportive functions are very helpful to big organizations where different people have different goals to achieve. Morever, when it comes to enterprise content management systems, we should focus more on stability and consistency.

2.2 System Design

Web application is an interactive system which allows user to have various of interaction with the system. Yan and Guo summarized that in information architecture, the focus is on the organization, navigation, labeling and search systems that offer accessibility to the end user (Yan & Guo, 2010). However, currently, there is lack of a conceptual design and design management framework leveraging design thinking and system thinking that take human-centeredness, understanding and acting of various stakeholders systemically into consideration (Vinnakota, 2016).

Therefore, we would rather identify the aesthetics of a user interface as a separate metric apart from its functionality. We should focus more on the satisfactory of a human user rather than the completion of required and other supportive functions. It is essential for us to separate the entire design into functional design and UI design.

The ever-growing complexity and operational scale of such systems means that the software must be validated at a level of scrutiny far greater than what manual quality assurance can accomplish (Hamill et. al., 2009). The Validation and verification techniques for effective privacy controls for secure data transmission. As for publications within an organization, which are properties of an organization, we need to validate each publication that is indeed accurate and has no missing values. Therefore, users should be able to interact with the system in a secure condition, and both frontend and backend server need to have validation function in place before any operations happen to the database.

A “user” in an UCD approach is considered to be knowledgeable and motivated about their work tasks and as such the goal for a designer is to design a tool to help users complete their work tasks easily and efficiently (Quintana et. al., 2000). Along with the idea of user-centered design, UI design will emphasize more on interaction and responsiveness of the application. This project aims to apply the latest web technology and framework to Publication Management area, in order to make it more interactive and responsive.

2.3 Evaluations

Since our web applications are versatile and diverse, the importance of evaluation and usability testing are not a child’s play. Jacob Nielsen, a Usability guru defines the usability according to the five components: Learnability, Efficiency, Memerobility, Few Errors, and Users' Satisfaction (Nielson, 1994). Another approach taken by H. Shahizan and Li Feng for the evaluation of the web usability is benchmarking (Shahizan & Li, 2005). They include seven comprehensive phases, which are screen appearance, consistency, accessibility, and navigation, media use, interactivity and content.

According to Wang and Yan, linking the analysis results of usability testing data and product design patterns would find out the source of usability problems and give practical guide (Wang & Yan, 2011). In addition, Voigt-Antons found that their results reveal that participants seem to be unable to rate the usability and Quality of Experience independently from each other (Voigt-Antons et. al., 2018).

Therefore, when conducting evaluation and usability testing, it is always important to link to the actual situation and also take user’s quality of experience into consideration. This provides a guideline for how to gather user’s feedback for evaluating the entire RENCI Publication Management System.

Section 3. System Design and Methods

3.1 User Requirements Collection

In this paper, requirements collection would be done individually through interviews within Renaissance Computing Institute about the needs for the Publication Management System from different user groups. I will set up appointments with various user groups, including but not limited to project managers, researchers, research developers, digital content specialist, administration people. Thoughts and relevant information will be collected, and the final requirement analysis should point out the main functionalities for Publication Management System which satisfies various user needs.

3.2 User Requirements Analysis

“Requirement analysis is the process of analyzing a new or modified product and determining the user expectations for the product” (Rouse, 2007, paragraph 1).

Specifically, the requirements analysis includes gathering user’s feedback, analyze the data, and make a guideline for further development and testing.

Here are some required functionalities:

- The system should be able to create, read, update and delete publications from the backend database. As the core function of Publication Management, it should categorize different RENCI publications, store and update them responsively.
- It should support basic and advanced search function, where users can choose multiple filters to get the search results they want. This would require the system assuring not only the database is ready to process any search but also the frontend is reflecting any new updates such as new categories or time period.
- The system would support data visualization function, showing the category patterns in a pie chart.
- The system is capable of inserting multiple publications at the same time, or have some automation process that allows bulk ingest function. Due to the large amount of RENCI publication, it is not a wise decision if the system only allows user to insert one publication at one time.
- The system should validate and verify publication before any operation can be done. Due to the large amount of publications, there are chances that users mistype the DOI of a publication, causing problems or potential breakdown when inserting publication

into database. What’s even worse is that the system stores the wrong publication into the database and user does not have a chance to figure that out.

Some other requirements might exist along with these required functions. For example, Nishikant *et al.*’s approach to develop a personal profile page, store search history or customized homepage as user’s demand. However, due to the limited time period, these advanced functionalities might only be considered as future improvement of the proposed system.

3.3 Architecture Design

Data in this project is mainly about publications at RENCI. This section discusses two main part where data transfer happens, one is at the database server, and the other is consume RESTful API to retrieve data from external servers. How are these data stored and fetched are concerned and proposed.

3.3.1 Database Design

The database used to store publication for the entire project has two models required in the project.

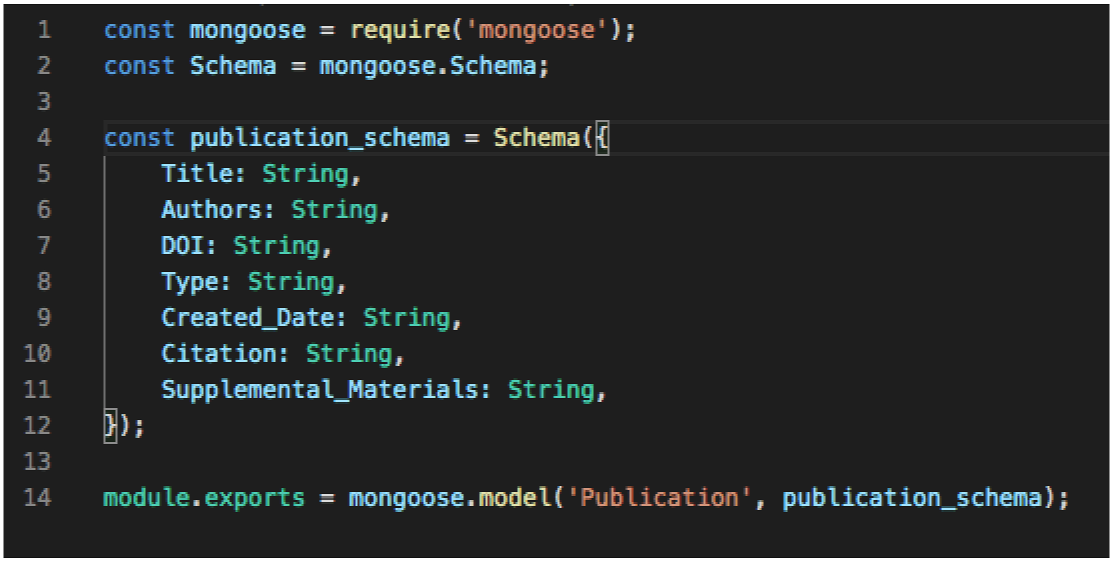
1. User Model:

   This user model is used to store user’s account information and credentials so only authorized users can get access to the RENCI

Publication Management System. Basic user schema would include id, email, first name, last name and password.

2. Publication Model:

   Publication model is relatively more sophisticated compared to user model, containing DOI information, title, author, category, citation, created date information and supplemental materials fields, which users can use to store information rather than the previous fields provided (Figure 1).

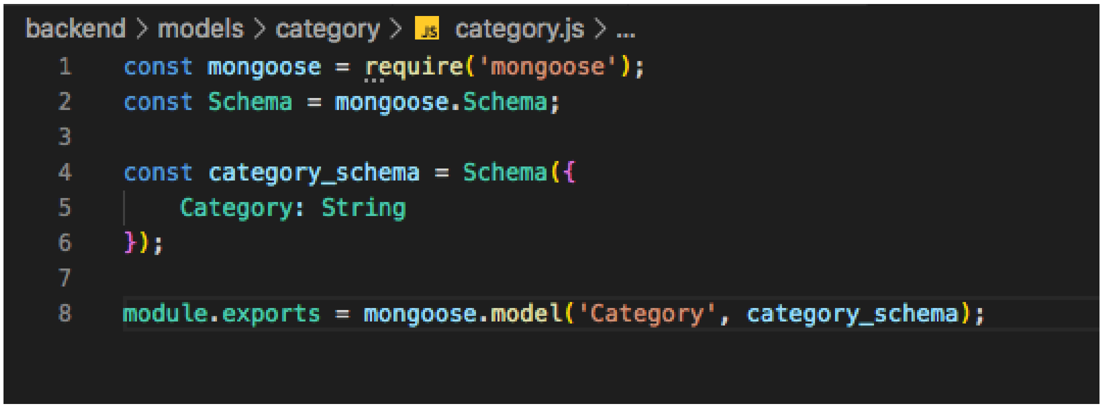


```
const mongoose = require('mongoose');
const Schema = mongoose.Schema;

const publication_schema = Schema({
    Title: String,
    Authors: String,
    DOI: String,
    Type: String,
    Created_Date: String,
    Citation: String,
    Supplemental_Materials: String,
});

module.exports = mongoose.model('Publication', publication_schema);
```

Figure 1. Publication Data Model

3. Category Model:

   When linking database to our system, it should be vital that any new categories added to the database should be reflected to the frontend page interactively (Figure 2).

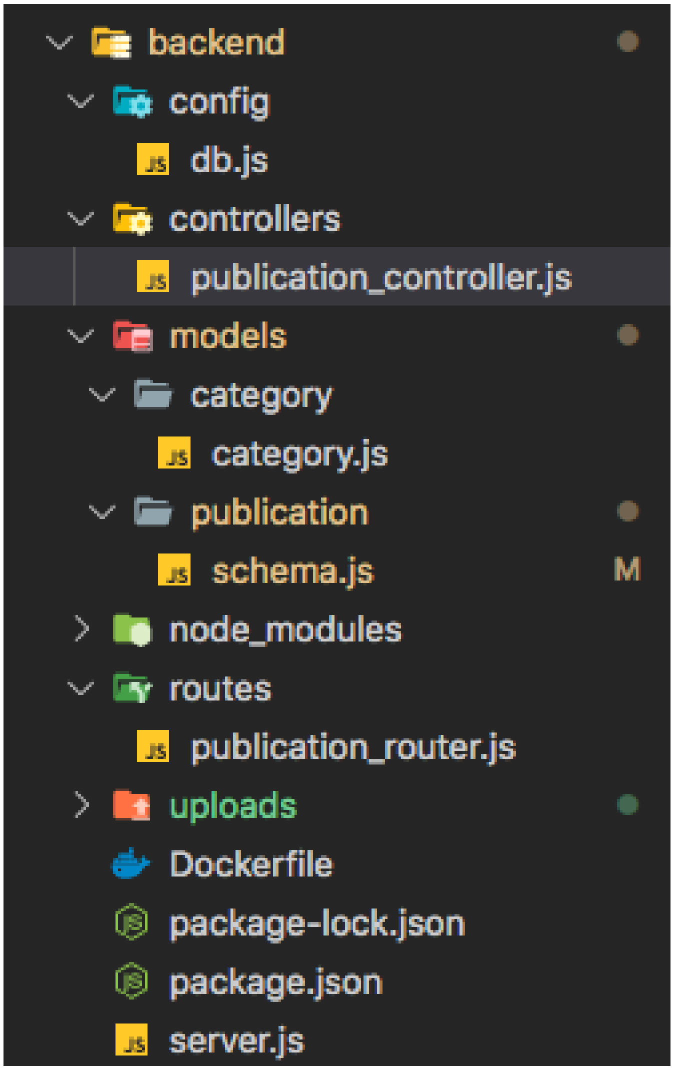


```javascript
const mongoose = require('mongoose');
const Schema = mongoose.Schema;

const category_schema = Schema({
    Category: String
});

module.exports = mongoose.model('Category', category_schema);
```

Figure 2. Category Data Model

3.3.2 Backend Design

The entire backend system is built on top of Express.js framework. To improve code readability and architecture, it will be made up of controllers, data models, database configuration and router components. In this way, developers can have a better understanding of how the backend system works and it is easier to modify or add new features in the future. A tree view of the backend system can be found here (Figure 3).



```
backend
  config
    db.js
  controllers
    publication_controller.js
  models
    category
      category.js
    publication
      schema.js                M
  node_modules
  routes
    publication_router.js
  uploads
  Dockerfile
  package-lock.json
  package.json
  server.js
```

Figure 3

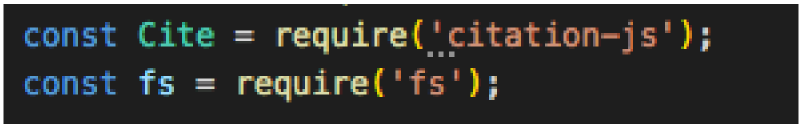
3.3.3 Database Insertion & RESTful API Design

This section mainly describes how to use external API to improve overall efficiency when inserting a large amount of publications into the database. Rather than typing in each field manually, setting up RESTful API would save both user and systems' processing time and improve the workflow at the same time.

Citation.js API provides RESTful API where our system can retrieve publication information by sending requests, including title, author, doi, creation date. This would be implemented in the express backend server.

a) Setup

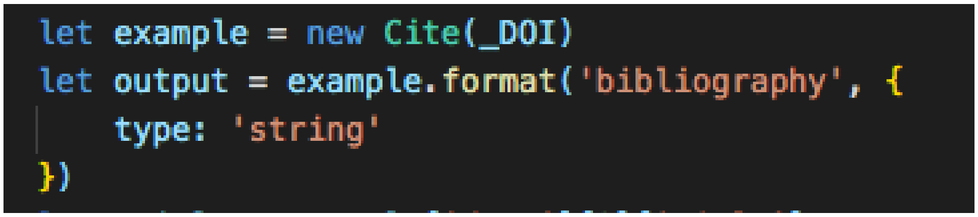
First of all, we need to load Citation.js into our controller (Figure 4).



```
const Cite = require('citation-js');
const fs = require('fs');
```

Figure 4

Once we have a publication’s DOI, we can fetch every piece of metadata it includes, by calling ‘Cite’ method (Figure 5). By specifying bibliography, we are able to generate an APA style citation. Along with citation, it also includes publication title, authors, created date, category and other details which can be further used in our publication management system.

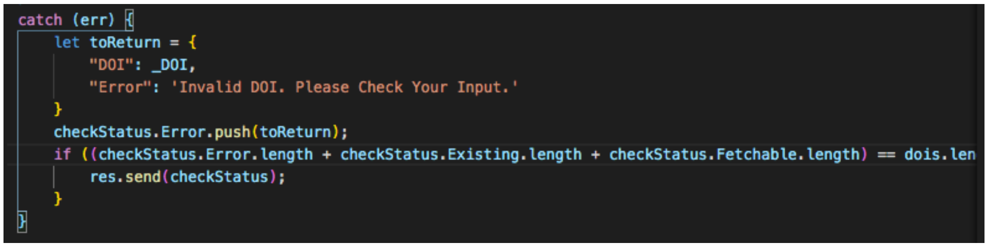


```
let example = new Cite(_DOI)
let output = example.format('bibliography', {
    type: 'string'
})
```

Figure 5

b) Backend Validation

Once we fetch all the data using Citation.js, it is essential to make sure that the data retrieved is valid. There would be times that the DOI is invalid, or the Citation.js itself does not include this publication information. A ‘Try-Catch’ logic is implemented (Figure 6) to detect potential errors and prevent this application from breaking down.



```
catch (err) {
    let toReturn = {
        "DOI": _DOI,
        "Error": 'Invalid DOI. Please Check Your Input.'
    }
    checkStatus.Error.push(toReturn);
    if ((checkStatus.Error.length + checkStatus.Existing.length + checkStatus.Fetchable.length) == dois.len
        res.send(checkStatus);
    }
}
```

Figure 6

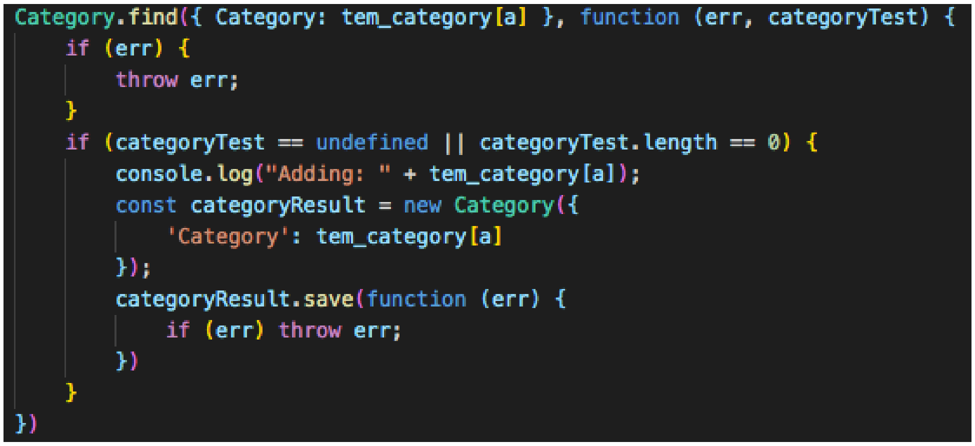
c) Frontend Validation

If we use backend validation to filter those invalid results, this frontend validation is to provide a way for users to decide whether the information is the one that they want to add. If data retrieved from the backend system is irrelevant or incorrect. They can choose not to insert them at this process. By checking boxes on the left side of each publication, the frontend React application would send the filtered requests to the backend system, where it would be insert into the database.

d) Database Insertion

Like we discussed in the previous section, our MongoDB consists of three collections, which are publications, publication categories and user information. We keep it in this way because publication categories would play an important role in our ‘Search Function’. Keeping it in a separate, relatively light-weight database allows us to update the React frontend application very quickly.

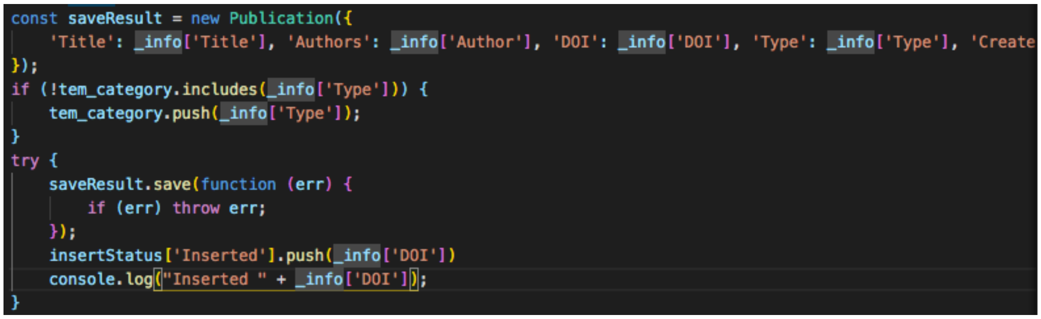
As the importance of publication categories being discussed, it is necessary for us to make sure that each publication will have its category inserted into the category collection. MongoDB will not detect duplicates when doing insertion operations. Therefore, find method is used to check whether the publication has its category in place, and if not, we will add these categories accordingly (Figure 7).



```
Category.find({ Category: tem_category[a] }, function (err, categoryTest) {
    if (err) {
        throw err;
    }
    if (categoryTest == undefined || categoryTest.length == 0) {
        console.log("Adding: " + tem_category[a]);
        const categoryResult = new Category({
            'Category': tem_category[a]
        });
        categoryResult.save(function (err) {
            if (err) throw err;
        })
    }
})
```

Figure 7

Once we make sure that the publication category is in place, we can insert the publication into the ‘publication’ collection (Figure 8).



```
const saveResult = new Publication({
    'Title': _info['Title'], 'Authors': _info['Author'], 'DOI': _info['DOI'], 'Type': _info['Type'], 'Create
});
if (!tem_category.includes(_info['Type'])) {
    tem_category.push(_info['Type']);
}
try {
    saveResult.save(function (err) {
        if (err) throw err;
    });
    insertStatus['Inserted'].push(_info['DOI'])
    console.log("Inserted " + _info['DOI']);
}
```

Figure 8

3.3.3 Frontend Design

Interactive JavaScript Framework has played a vital role when designing and developing modern responsive websites. React is used in this application as we can take advantage of its high responsiveness, updating web contents without refreshing the entire website, which would take more time and might further decrease user experience.

Material UI is implemented throughout entire application, which makes web development faster and easier.

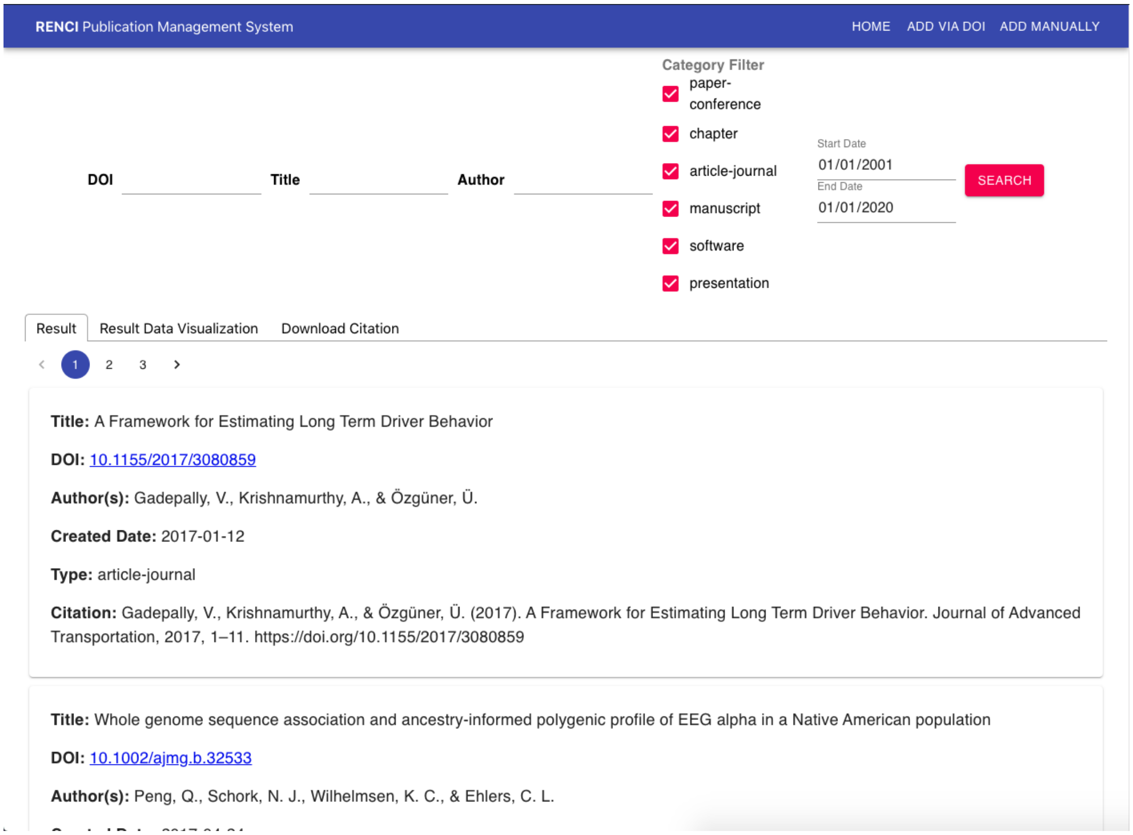
a) Homepage Design

The main function of homepage is the search function, where user can search for publication stored in the database using up to 5 metrics, including DOI, title, authors, category, start date and end date (Figure 9).

The category filter fields will load before any search operation, which would require very quick database response. Again, keeping publication category in a separate collection would greatly improve loading performance.

Along with search function, this application also provides a data visualization board for users to better understanding the category patterns of results. Pie chart is used to show the percentage of each category in resulting publications (Figure 10).

As we discussed in the previous section, each publication would come along with a citation. For users who want to search for some publications and then cite them accordingly. Our application also provides a download citation tab, where users can download a list of citations, which will be stored in a text file (Figure 11).



DOI ______ Title ______ Author ______

Category Filter

☑ paper-conference

☑ chapter

☑ article-journal

☑ manuscript

☑ software

☑ presentation

Start Date 01/01/2001

End Date 01/01/2020

Result | Result Data Visualization | Download Citation

Title: A Framework for Estimating Long Term Driver Behavior

DOI: 10.1155/2017/3080859

Author(s): Gadepally, V., Krishnamurthy, A., & Özgüner, Ü.

Created Date: 2017-01-12

Type: article-journal

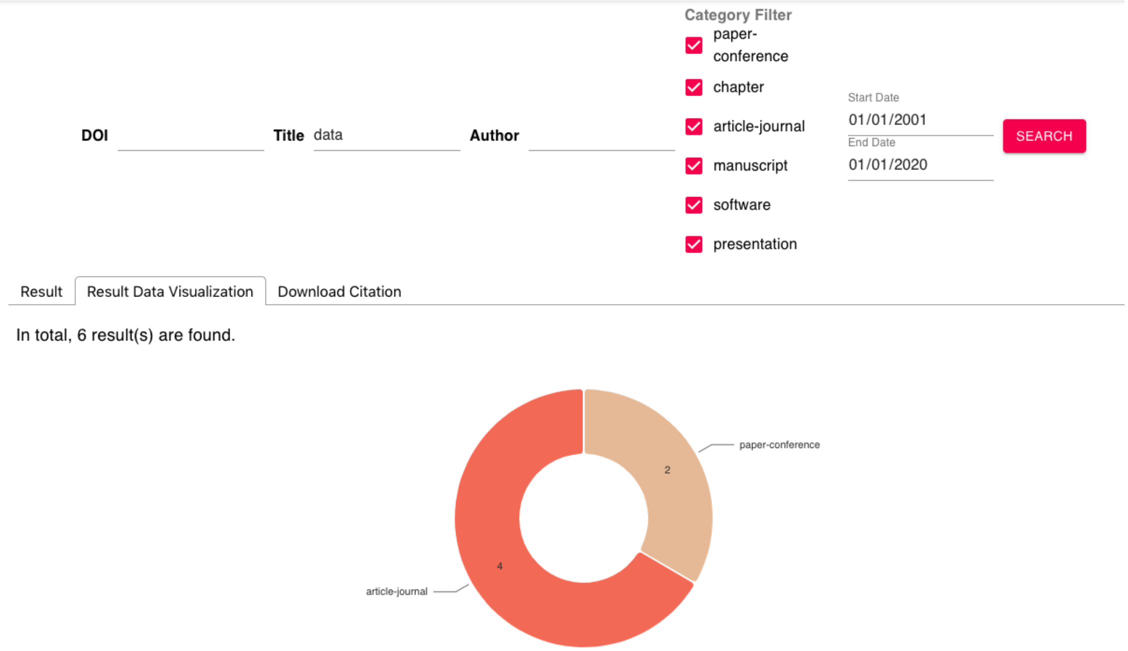
Citation: Gadepally, V., Krishnamurthy, A., & Özgüner, Ü. (2017). A Framework for Estimating Long Term Driver Behavior. Journal of Advanced Transportation, 2017, 1–11. https://doi.org/10.1155/2017/3080859

Title: Whole genome sequence association and ancestry-informed polygenic profile of EEG alpha in a Native American population

DOI: 10.1002/ajmg.b.32533

Author(s): Peng, Q., Schork, N. J., Wilhelmsen, K. C., & Ehlers, C. L.

Figure 9



DOI ______ Title data Author ______

Category Filter

☑ paper-conference

☑ chapter

☑ article-journal

☑ manuscript

☑ software

☑ presentation

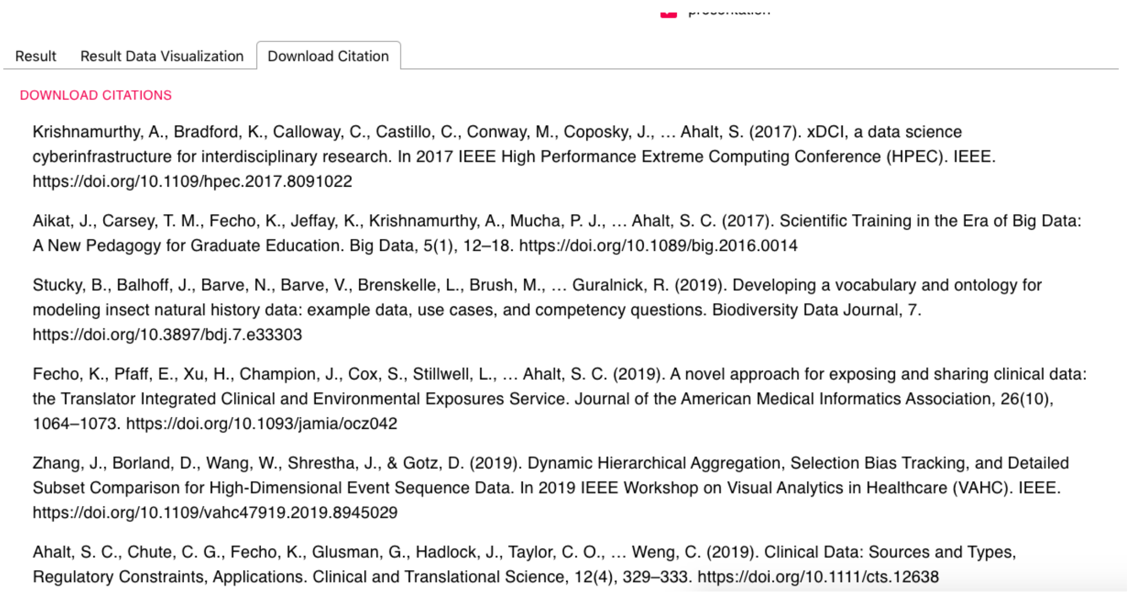
Start Date 01/01/2001

End Date 01/01/2020

Result | Result Data Visualization | Download Citation

In total, 6 result(s) are found.

Figure 10



DOWNLOAD CITATIONS

Krishnamurthy, A., Bradford, K., Calloway, C., Castillo, C., Conway, M., Coposky, J., … Ahalt, S. (2017). xDCI, a data science cyberinfrastructure for interdisciplinary research. In 2017 IEEE High Performance Extreme Computing Conference (HPEC). IEEE. https://doi.org/10.1109/hpec.2017.8091022

Aikat, J., Carsey, T. M., Fecho, K., Jeffay, K., Krishnamurthy, A., Mucha, P. J., … Ahalt, S. C. (2017). Scientific Training in the Era of Big Data: A New Pedagogy for Graduate Education. Big Data, 5(1), 12–18. https://doi.org/10.1089/big.2016.0014

Stucky, B., Balhoff, J., Barve, N., Barve, V., Brenskelle, L., Brush, M., … Guralnick, R. (2019). Developing a vocabulary and ontology for modeling insect natural history data: example data, use cases, and competency questions. Biodiversity Data Journal, 7. https://doi.org/10.3897/bdj.7.e33303

Fecho, K., Pfaff, E., Xu, H., Champion, J., Cox, S., Stillwell, L., … Ahalt, S. C. (2019). A novel approach for exposing and sharing clinical data: the Translator Integrated Clinical and Environmental Exposures Service. Journal of the American Medical Informatics Association, 26(10), 1064–1073. https://doi.org/10.1093/jamia/ocz042

Zhang, J., Borland, D., Wang, W., Shrestha, J., & Gotz, D. (2019). Dynamic Hierarchical Aggregation, Selection Bias Tracking, and Detailed Subset Comparison for High-Dimensional Event Sequence Data. In 2019 IEEE Workshop on Visual Analytics in Healthcare (VAHC). IEEE. https://doi.org/10.1109/vahc47919.2019.8945029

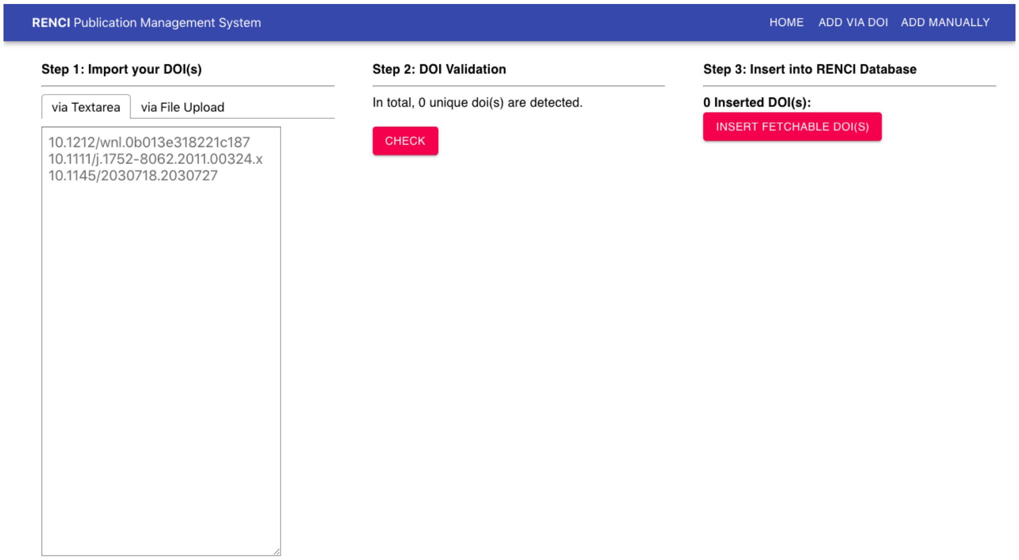
Ahalt, S. C., Chute, C. G., Fecho, K., Glusman, G., Hadlock, J., Taylor, C. O., … Weng, C. (2019). Clinical Data: Sources and Types, Regulatory Constraints, Applications. Clinical and Translational Science, 12(4), 329–333. https://doi.org/10.1111/cts.12638

Figure 11

b) Add Page Design

As being discussed, this ‘Add’ page would support publication insertion. Built on top of RESTful API, which means users only need to provide DOI information of a publication and the system would do the rest of work. Considering that there might be a huge amount of user input request, we provide two different methods for users to dump in their DOIs. One solution is that user can copy a list of DOI into the textarea provided and the other is through file upload (Figure 12).

To complete publication insertion, a user needs to go through three steps as listed on the page. After loading all the DOI information, the next step is to click the ‘CHECK’ button to validate all the input DOI information is valid and the metadata shown is exactly the same as user's intention. If there is any DOI that has already added to the database, it would be detected at this step and notify user that it is stored in the database. If every piece of information is correct and users are ready to insert them into the database, they can click the 'INSERT' button at the right. All the confirmed metadata would be added to MongoDB database at this step.



Step 1: Import your DOI(s)

10.1212/wnl.0b013e318221c187
10.1111/j.1752-8062.2011.00324.x
10.1145/2030718.2030727

Step 2: DOI Validation

In total, 0 unique doi(s) are detected.

Step 3: Insert into RENCI Database

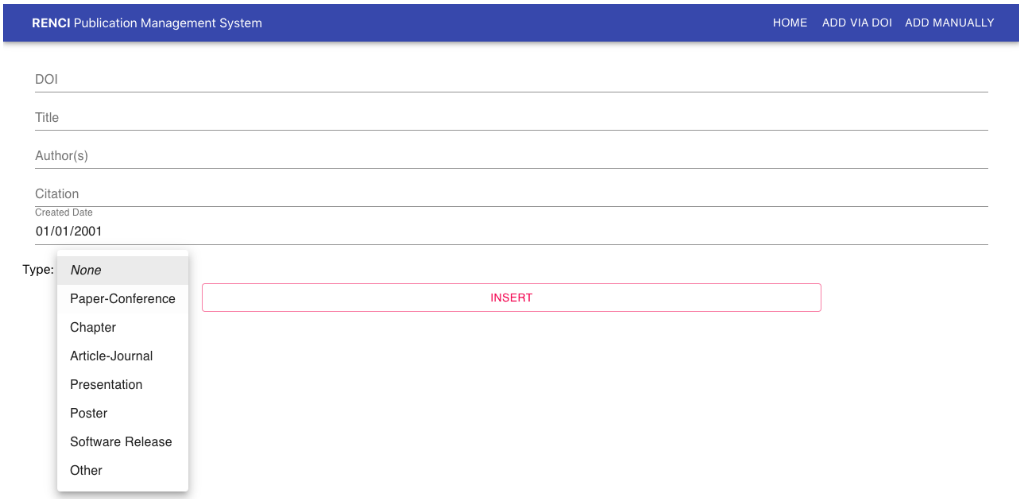
0 Inserted DOI(s):

Figure 12

c) Second Add Page Design

There might be circumstances that the main 'Add' function does not provide correct publication metadata, or a user does not have its DOI information. This second Add page would serve as an alternative solution. Five fields are provided, and users can choose one category field out of seven given categories (Figure 13).

Once the user clicks ‘INSERT’ button, all metadata would be added to the database, including the specific category collection. Supporting this function would allow us to expand the user target group.



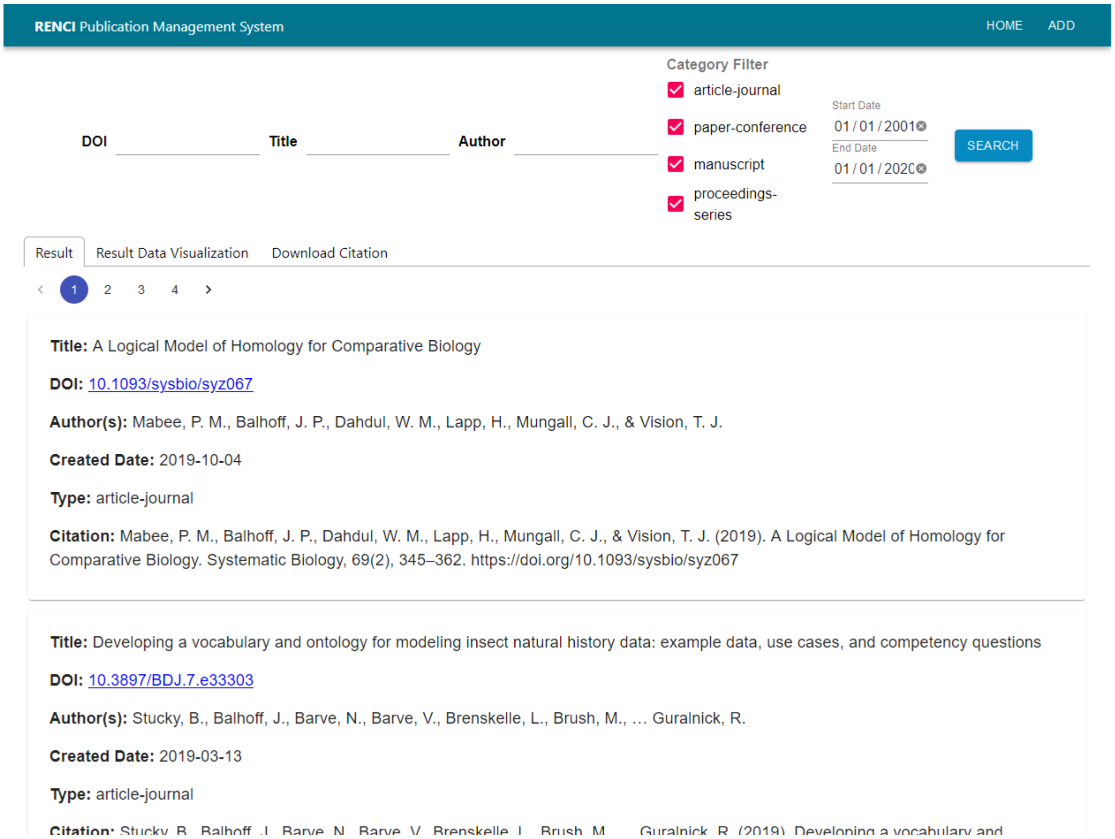
Created Date
01/01/2001

Figure 13

3.4 Anticipated Impact

If the system is developed successfully and meet all required functions, it should be able to have a homepage (Figure 14), which would allow users to browse all publications stored in RENCI’s database, along with ‘search’ function, where users are able to make query to get results they want, an ‘add’ page, where users can paste a list of publication’s dois (Figure 15a) or upload a text file, containing a list of DOIs to insert into the system

(Figure 15b), along with user authentication and verification. The development is still in progress, so the figure shown here is of full functionalities.


RENCI Publication Management System
HOME
ADD
DOI
Title
Author
SEARCH
Result
Result Data Visualization
Download Citation
1
2
3
4

Category Filter

☑ article-journal

☑ paper-conference

☑ manuscript

☑ proceedings-series

Start Date 01 / 01 / 2001

End Date 01 / 01 / 2020

Title: A Logical Model of Homology for Comparative Biology

DOI: 10.1093/sysbio/syz067

Author(s): Mabee, P. M., Balhoff, J. P., Dahdul, W. M., Lapp, H., Mungall, C. J., & Vision, T. J.

Created Date: 2019-10-04

Type: article-journal

Citation: Mabee, P. M., Balhoff, J. P., Dahdul, W. M., Lapp, H., Mungall, C. J., & Vision, T. J. (2019). A Logical Model of Homology for Comparative Biology. Systematic Biology, 69(2), 345–362. https://doi.org/10.1093/sysbio/syz067

Title: Developing a vocabulary and ontology for modeling insect natural history data: example data, use cases, and competency questions

DOI: 10.3897/BDJ.7.e33303

Author(s): Stucky, B., Balhoff, J., Barve, N., Barve, V., Brenskelle, L., Brush, M., … Guralnick, R.

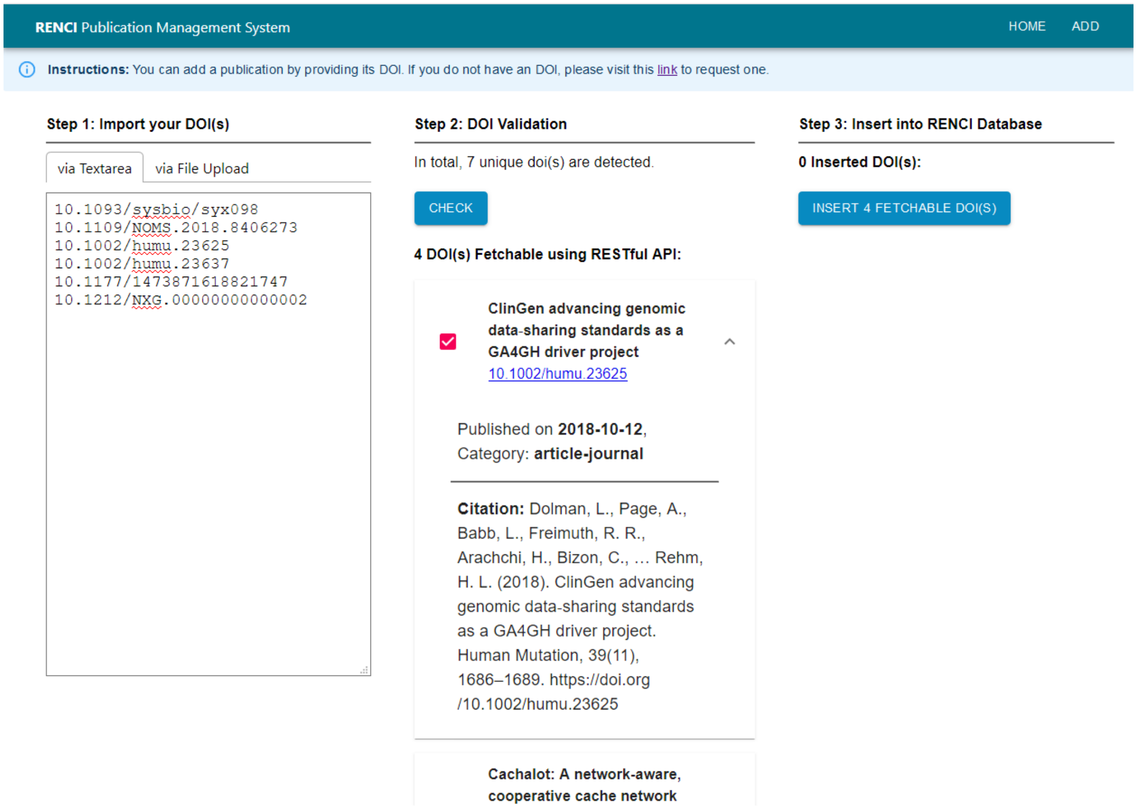
Created Date: 2019-03-13

Type: article-journal

Citation: Stucky, B., Balhoff, J., Barve, N., Barve, V., Brenskelle, L., Brush, M., … Guralnick, R. (2019). Developing a vocabulary and

Figure 14. Homepage of RENCI Publication Management System

As for UI design, the primary goal to implement user-centered design, where most of the goals in the user's requirements are met. The second goal is that the Publication Management System should fit into RENCI's overall design ideas and style, because in the future it might be integrated into the RENCI website.


RENCI Publication Management System
HOME
ADD
via Textarea
via File Upload
CHECK
INSERT 4 FETCHABLE DOI(S)

Instructions: You can add a publication by providing its DOI. If you do not have an DOI, please visit this link to request one.

Step 1: Import your DOI(s)

```
10.1093/sysbio/syx098
10.1109/NOMS.2018.8406273
10.1002/humu.23625
10.1002/humu.23637
10.1177/1473871618821747
10.1212/NXG.00000000000002
```

Step 2: DOI Validation

In total, 7 unique doi(s) are detected.

4 DOI(s) Fetchable using RESTful API:

☑ **ClinGen advancing genomic data-sharing standards as a GA4GH driver project**
10.1002/humu.23625

Published on **2018-10-12**,
Category: **article-journal**

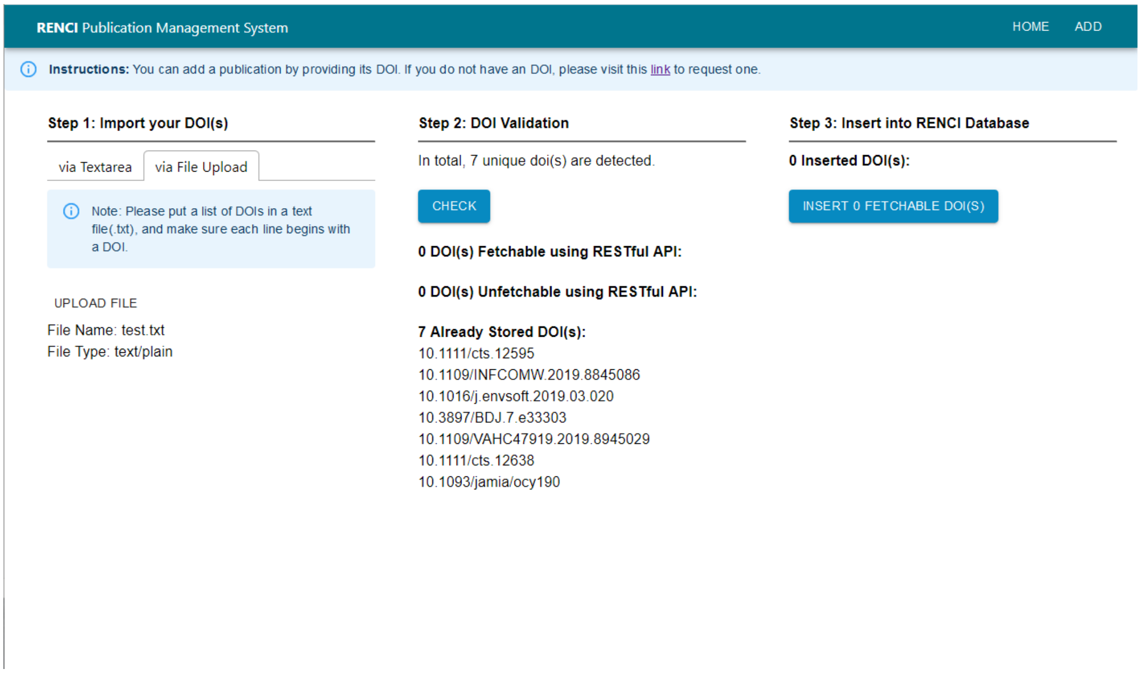
Citation: Dolman, L., Page, A., Babb, L., Freimuth, R. R., Arachchi, H., Bizon, C., … Rehm, H. L. (2018). ClinGen advancing genomic data-sharing standards as a GA4GH driver project. Human Mutation, 39(11), 1686–1689. https://doi.org/10.1002/humu.23625

Cachalot: A network-aware, cooperative cache network

Step 3: Insert into RENCI Database

0 Inserted DOI(s):

Figure 15a



Instructions: You can add a publication by providing its DOI. If you do not have an DOI, please visit this link to request one.

Step 1: Import your DOI(s)

Note: Please put a list of DOIs in a text file(.txt), and make sure each line begins with a DOI.

File Name: test.txt
File Type: text/plain

Step 2: DOI Validation

In total, 7 unique doi(s) are detected.

0 DOI(s) Fetchable using RESTful API:

0 DOI(s) Unfetchable using RESTful API:

7 Already Stored DOI(s):
10.1111/cts.12595
10.1109/INFCOMW.2019.8845086
10.1016/j.envsoft.2019.03.020
10.3897/BDJ.7.e33303
10.1109/VAHC47919.2019.8945029
10.1111/cts.12638
10.1093/jamia/ocy190

Step 3: Insert into RENCI Database

0 Inserted DOI(s):

Figure 15b

Section 4. Conclusion and User Feedback

This paper gives an overview of the design and development of a Publication Management System in a real-world environment. Single Page Application has become a hot topic in the recent year, its high responsiveness enables user to have a better user experience when comparing with traditional website, especially when it comes to dashboard and interactive system design and development. From the usage of NoSQL in this application, it greatly improves the overall backend performance as it is relatively a lightweight database choice. Express.js is always a hot pick in the node.js backend development.

4.1 User Feedback

Throughout the entire development process, several talks and meetings have been scheduled to listen to user' feedback and make the application better meet user's demand. It is very essential that we should keep user's words in mind while developing, rather than just implementing what we have in our mind.

Brittany Todd, one of the Project Manager Team at Renaissance Computing Institute, gave us the suggestion of integrating this application within the big RENCI homepage, which is very inspiring. Developing a project probably would take a period of time, which means there might be changes in the outside world and we need to make sure our application can fit into the bigger picture very well.

4.2 Limitations

There are certain limitations existing in this paper. Automatic publication insertion is really useful when users have their DOI in place. However, when it comes to those publications that does not have a DOI, it might take extra time and effort to insert a certain amount of publications using ‘Add-Manually’ function.

There could also be chances that some users might not be in flavor in of auto-insert function when compared with traditional input-insert function. The result might be not generalizable.

References

A. Naak, H. Hage and E. Aïmeur, "Papyres: A Research Paper Management System," *2008 10th IEEE Conference on E-Commerce Technology and the Fifth IEEE Conference on Enterprise Computing, E-Commerce and E-Services,* Washington, DC, 2008, pp. 201-208. doi: 10.1109/CECandEEE.2008.132

E. Folmer and J. Bosch. "Architecting for Usability: A Survey," *Journal of Systems and Software*, Issue 70-1, January 2004, pp. 61-78.

Emmanuel Hooper, 2009: E. Hooper, "Intelligent strategies for secure complex systems integration and design, effective risk management and privacy," *2009 3rd Annual IEEE Systems Conference*, Vancouver, BC, 2009, pp. 257- 261. doi: 10.1109/SYSTEMS.2009.4815808

Guan *et al.* S. Guan, W. Hu and H. Zhou, "Front-end and Back-end Separation - React Based Framework for Networked Remote Control Laboratory," *2018 37th Chinese Control Conference (CCC)*, Wuhan, 2018, pp. 6314-6319. doi: 10.23919/ChiCC.2018.8483988

P. Hamill, D. Alexander and S. Shasharina, "Web Service Validation Enabling Test-Driven Development of Service-Oriented Applications," *2009 Congress on Services* - I, Los Angeles, CA, 2009, pp. 467-470. doi: 10.1109/SERVICES-I.2009.113

H. Shahizan, L. Feng, "Evaluating the Usability and Content Usefulness of Web Sites: A Benchmarking approach," *Journal of Electronic Commerce in Organizations, ABI/INFORM Global*, Apr-Jun 2005, pp. 46

Introduction to NoSQL. (n.d.) Retrieved from https://www.geeksforgeeks.org/introduction-to-nosql/.

N. Kapoor, J. Chen, J. T. Butler, G. C. Fouty, J. A. Stemper, J. Riedl, and J.A. Konstan, "Techlens: a researcher's desktop," in *Proceedings of the 2007*

ACM conference on Recommender systems Minneapolis, MN, USA: ACM, 2007, pp. 183-184.

Quintana, C., Krajcik, J., & Soloway, E., Exploring a Structured Definition for Learner-Centered Design. In *B. Fishman & S. O'Connor-Divelbiss (Eds.), Fourth International Conference of the Learning Sciences*, pp. 256-263, 2000.

RDBMS. (n.d.). Retrieved from https://techterms.com/definition/rdbms/.

Rouse, M. (2007). requirements analysis (requirements engineering). Retrieved from http://searchsoftwarequality.techtarget.com/definition/requirements-analysis

S. Simonova and A. Hudec, "Enterprise content management based on identified requirements," 2015 International Conference on Information and Digital Technologies, Zilina, 2015, pp. 324-329. doi: 10.1109/DT.2015.7222991

Tang, T. Y., & Mccalla, G. (2009). A Multidimensional Paper Recommender: Experiments and Evaluations. *IEEE Internet Computing*, 13(4), 34–41. doi: 10.1109/mic.2009.73

Tseng-Chang Yen and Shang-Juh Kao, "XBMS - an open XML bibliography management system," *Third International Conference on Information Technology and Applications* (ICITA'05), Sydney, NSW, 2005, pp. 88-94 vol.1. doi: 10.1109/ICITA.2005.314

T. Vinnakota, "A conceptual framework for complex system design and design management," 2016 Annual IEEE Systems Conference (SysCon), Orlando, FL, 2016, pp. 1-6. doi: 10.1109/SYSCON.2016.7490575

J. Voigt-Antons, T. Hoßfeld, S. Egger-Lampl, R. Schatz and S. Möller, "User Experience of Web Browsing - The Relationship of Usability and Quality of Experience," *2018 Tenth International Conference on Quality of Multimedia Experience (QoMEX)*, Cagliari, 2018, pp. 13.doi:10.1109/QoMEX.2018.8463383

W. Haiyan and Y. Baozhu, "A data-processing mechanism for scenario-based usability testing," 2011 IEEE 2nd International Conference on Computing, Control and Industrial Engineering, Wuhan, 2011, pp. 395-398. doi: 10.1109/CCIENG.2011.6008042

X. Huang, A. An, N. Cercone, and G. Promhouse, "Discovery of interesting association rules from Liv elink web log data," in IEEE International Conference on Data Mining (ICDM 2002), Maebashi, Japan, 2002, pp. 763- 766.

Pei Yan and Jiao Guo, "The research of Web usability design," 2010 The 2nd International Conference on Computer and Automation Engineering (ICCAE), Singapore, 2010, pp. 480-483. doi: 10.1109/ICCAE.2010.5451619

Y. Hai-jie and L. Xiang-yang, "Enterprise Software System Development Based o n Management Pattern Components," 2006 International Conference on Management Science and Engineering, Lille, 2006, pp. 213-217. doi: 10.1109/ICMSE.2006.313853