

**A framework for teaching secure coding
practices through a blended learning
approach**

by

Vuyolwethu Sizoli Mdunyelwa

A framework for teaching secure coding practices through a blended learning approach

by

Vuyolwethu Sizoli Mdunyelwa

Dissertation

submitted in fulfilment
of the requirements
for the degree

Master of Information Technology

in the

**Faculty of Engineering, the Built Environment and
Information Technology**

of the

Nelson Mandela University

Supervisor: Prof. Johan van Niekerk

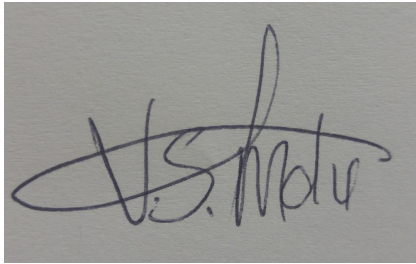
Co-supervisor: Prof. Lynn Fletcher

April 2019

Declaration

I, Vuyolwethu Sizoli Mdunyelwa, hereby declare that:

- The work in this dissertation is my own work.
- All sources used or referred to have been documented and recognised.
- This dissertation has not previously been submitted in full or partial fulfilment of the requirements for an equivalent or higher qualification at any other recognised educational institute.

A handwritten signature in dark ink, appearing to read 'V.S. Mdunyelwa', is shown on a light-colored background.

Vuyolwethu Sizoli Mdunyelwa

Abstract

With the recent increase in cyber-related attacks, cybersecurity is becoming a key area of concern for many organisations. Cybersecurity vulnerabilities are typically addressed through the implementation of various cybersecurity controls. These controls can be operational, technical or physical in nature. The focus of this research, however, is on technical controls with a specific focus on securing web applications.

This research firstly investigated whether third year software development students at the Nelson Mandela University adhered to secure coding practices in their capstone projects. In order to determine adherence, secure coding practices were identified from OWASP for the data access layer in web applications developed in the .NET environment. This was addressed by Secondary Objective 1, which was *To determine what secure coding practices a web application developer should adhere to in the .NET environment*. These secure coding practices were used to conduct a code review on 2015 third year capstone projects, and addressed Secondary Objective 2, *To determine the adherence of third year software development capstone projects to the identified secure coding practices*. The results for the code review were analysed and indicated low levels of adherence which led to the Problem Statement of this research, namely: *Undergraduate software development students do not consistently adhere to secure coding practices when developing their third year capstone projects, thereby leading to vulnerabilities in their web applications*. In order to address this Problem Statement, the Primary Objective was identified, *To develop a framework for teaching secure coding practices through a blended learning approach*.

Secondary Objective 3, *To determine whether third year software development students have the requisite knowledge relating to secure coding*, took the form of a questionnaire to assess students' knowledge relating to secure

coding practices. This required the achievement of further sub-objectives which addressed both the knowledge and behaviour of software development students. The results of this questionnaire indicated that many of the third year software development students lacked the requisite knowledge.

This lack of knowledge and adherence was addressed through an educational intervention, meeting Secondary Objective 4, *To design and implement an educational intervention to support software development students in the development of secure web applications*. In terms of knowledge, online lessons were developed addressing each of the secure coding practices identified. In order to address adherence, students were given a checklist to monitor their adherence to the identified secure coding practices.

Secondary Objective 5, *To determine the effect of the educational intervention on both student adherence and their requisite knowledge regarding secure coding practices*, involved the verification of the educational intervention, and comprised of two components, knowledge and behaviour. Knowledge verification took the form of an online questionnaire given to 2017 third year project students. To address behavioural adherence, the researcher conducted a code review on the 2017 capstone projects. The results from the verification showed a general improvement in students' knowledge and high levels of adherence to secure coding practices.

Finally, a framework was developed that encompassed the key elements of this research, thereby providing guidance to support the development of secure web applications in higher education institutions and meeting the primary objective of this study.

Acknowledgements

Foremost, I would like to express my sincere gratitude to my supervisors Prof. Johan van Niekerk and Prof. Lynn Fitcher for their continuous support during my Master's study. Furthermore, I would like to express my gratitude for their patience, motivation, enthusiasm, and immense knowledge. Their guidance assisted me throughout the research and writing of this dissertation. I could not have imagined having better supervisors for my Master's study. Furthermore, I would like to thank the following benefactors for their financial assistance:

- The financial assistance of the National Research Foundation (NRF) towards this research is hereby acknowledged. Opinions expressed and conclusions arrived at, are those of the authors, and are not necessarily to be attributed to the National Research Foundation.
- The financial assistance of the Nelson Mandela University's Post Graduate Research Scholarship (PGRS) is also hereby acknowledged.

Finally, I would like to thank my parents, (Mzikayise and Lulama), my siblings (Amanda, Undilile and Sivenathi), whose support helped me to progress my studies.

Contents

Declaration	i
Abstract	ii
Acknowledgements	iv
1 Introduction	1
1.1 Theoretical Background	2
1.2 Secure Coding Practices	3
1.3 Problem Statement	4
1.4 Primary Research Objective	5
1.5 Secondary Research Objectives	5
1.6 Delineation	6
1.7 Overview of Research Design	6
1.8 Ethical Considerations	9
1.9 Chapter Outline	9
1.10 Conclusion	11
2 Web Application Security	12
2.1 Introduction	13
2.2 Evolution of Computing	13
2.3 What is Software Security?	16
2.4 How to Secure Software	18
2.4.1 TCP/IP and OSI Reference Models	18
2.4.2 Application Layer	21
2.5 How to Secure Web Applications	22
2.6 Standards and Best Practices for Secure Web Application Development	23

2.7	Open Web Application Security Project	25
2.8	Conclusion	28
3	Theoretical Grounding	29
3.1	Introduction	30
3.2	Educational Theories	31
3.2.1	Blended learning	31
3.2.2	What is Online Learning?	33
3.2.3	Brain-compatible Educational Principles	34
3.3	Behavioural Compliance Monitoring	38
3.3.1	Evaluation and Feedback	38
3.3.2	Checklist	39
3.4	Conclusion	41
4	Research Process and Design	42
4.1	Introduction	43
4.2	Research Approach	43
4.3	Research Setting	44
4.3.1	Curriculum Guidelines	44
4.3.2	Capstone Projects	45
4.3.3	Participating Sampling	48
4.4	Research Process	49
4.4.1	Phase 1: Behavioural Analysis	50
4.4.2	Phase 2: Knowledge Assessment	50
4.4.3	Phase 3: Educational Intervention	51
4.4.4	Phase 4: Verification	52
4.5	Conclusion	52
5	Phase 1 - Behavioural Analysis	53
5.1	Introduction	54
5.2	Behavioural Analysis Process Flow	54
5.3	Code Review Instrument Design	55
5.4	Conducting the Behavioural Analysis	57
5.4.1	Parameterised SQL Commands	58
5.4.2	Concatenated SQL Strings	58
5.4.3	Input Validation	59

5.4.4	Principle of Least Privilege	59
5.4.5	Authentication	60
5.4.6	Stored Procedures	60
5.4.7	Encrypting Connection Strings	61
5.4.8	Connection Strings	61
5.4.9	Encryption using Acceptable Methods	62
5.5	Results of Code Review	62
5.6	Behavioural Analysis Discussion	64
5.7	Conclusion	66
6	Phase 2 - Knowledge Assessment	67
6.1	Introduction	68
6.2	Knowledge Assessment Process Flow	68
6.3	Knowledge Assessment Instrument Design	69
6.4	Conducting the Knowledge Assessment	76
6.5	Results of Knowledge Assessment	77
6.6	Knowledge Assessment Discussion	79
6.7	Conclusion	80
7	Phase 3 - Educational Intervention	81
7.1	Introduction	82
7.2	Educational Intervention Process Flow	82
7.3	Focus of Educational Intervention	83
7.4	Knowledge Component	84
7.4.1	Knowledge Component Design	85
7.4.2	Overview of the Curriculum	89
7.4.3	Administering the Knowledge Component	97
7.5	Behavioural Compliance Monitoring Instrument	98
7.5.1	Design of Behavioural Compliance Instrument	98
7.5.2	Conducting the Behavioural Compliance Instrument	99
7.6	Conclusion	100
8	Phase 4 - Verification	101
8.1	Introduction	102
8.2	Verification Process Flow	103
8.3	Knowledge Verification	104

8.3.1	Knowledge Verification Instrument Design	104
8.3.2	Conducting the Knowledge Verification Instrument . . .	104
8.3.3	Knowledge Verification Results	105
8.4	Behavioural Verification	109
8.4.1	Behavioural Verification Design	109
8.4.2	Conducting the Behavioural Verification	110
8.4.3	Behavioural Verification Results	111
8.5	Verification Discussion	113
8.6	Conclusion	113
9	Conclusion	115
9.1	Introduction	116
9.2	Summary of Chapters	116
9.3	Meeting the Research Objectives	118
9.4	Research Contribution - The Framework	119
9.4.1	Research Key Elements	120
9.4.2	A Generic Framework for Teaching Secure Coding Prac- tices	122
9.5	Research Limitations	123
9.6	Suggestions for Future Research	124
9.7	Publication	124
9.8	Epilogue	124
	References	126
I	Appendices	134
A	Academic Publication	135
B	Questionnaire (Pre-Test)	146
C	Questionnaire (Post-Test)	147

List of Tables

1.1	Research Objectives and Related Research Methods	7
1.2	Chapter Outline	10
2.1	Standards and Best Practices Compared	23
2.2	OWASP Top 10 Vulnerability List 2017	26
2.3	OWASP Secure Coding Practices	27
3.1	Comprehensive Brain-Compatible Educational Principles . . .	35
3.2	Brain-Compatible Principles Relevant to this Study	36
5.1	Code Review Checklist	56
5.2	Adherence to Secure Coding Practices	64
6.1	OWASP Secure Coding Practices Related Questions	70
6.2	Results from Phase 1 and 2 Compared	79
7.1	OWASP Secure Coding Practices	84
7.2	Code Review Checklist	99
8.1	OWASP Secure Coding Practices Related Questions (Post-Test)	106
8.2	Knowledge Assessment and Verification Results (Pre-test vs Post-Test)	108
8.3	Code Review Checklist	110
8.4	Adherence to Secure Coding Practices	112

List of Figures

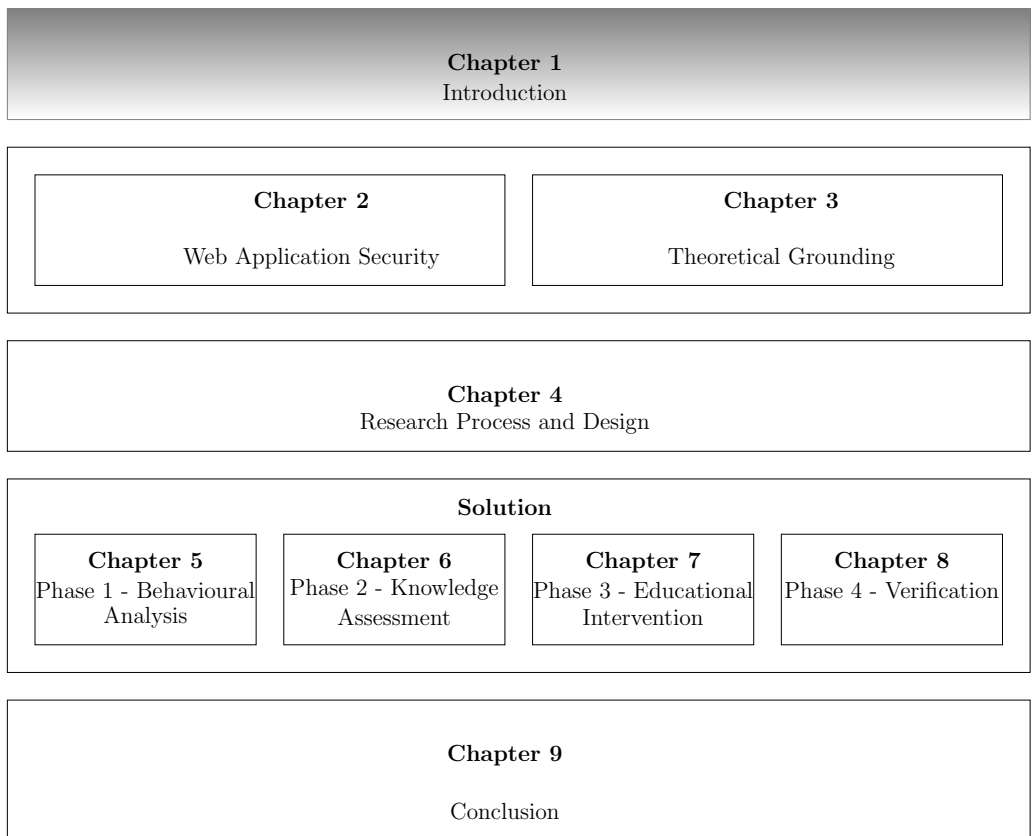
1.1	Research Design.	8
2.1	Security Needs and Roles of Programmers Over Time	15
2.2	OSI and TCP/IP Reference Models	19
3.1	Theoretical Grounding	31
3.2	Blended Learning	32
4.1	Subjects for the Diploma: Software Development	46
4.2	Phased Research Approach	49
5.1	Phase 1 - Behavioural Analysis Process Flow	54
5.2	Database for Code Review	57
5.3	Parameterised SQL Command Example	58
5.4	Concatenated SQL String Example	59
5.5	Principle of Least Privilege Example	60
5.6	Stored Procedure Example	61
5.7	Connection String Example	62
5.8	Captured Data Example (P14)	63
6.1	Phase 2 - Knowledge Assessment Process Flow	68
6.2	Sign In Scenario (Figure 1)	70
6.3	Knowledge Assessment - Question 1	71
6.4	Knowledge Assessment - Question 2	71
6.5	Knowledge Assessment - Question 3	72
6.6	Knowledge Assessment - Question 4	72
6.7	Knowledge Assessment - Question 5	72
6.8	Knowledge Assessment - Question 6	73
6.9	Knowledge Assessment - Question 7	73

6.10	Knowledge Assessment - Question 8	74
6.11	Knowledge Assessment - Question 9	74
6.12	Knowledge Assessment - Question 10	74
6.13	Knowledge Assessment - Question 11	75
6.14	Knowledge Assessment - Question 12	75
6.15	Knowledge Assessment - Question 13	75
6.16	Knowledge Assessment - Question 14	76
6.17	Knowledge Assessment - Question 15	76
6.18	Student Marks	77
6.19	Average percentage per Secure Coding Practice	78
6.20	Average Percentage per Question	78
7.1	Phase 3 - Educational Intervention Process Flow	83
7.2	Lesson Content Process Flow	85
7.3	Focused Attention and Peripheral Perception Example	87
7.4	Natural, Spatial Memory Example	88
7.5	Immediate Feedback Example	88
7.6	Lesson Introductory Slide (SP2)	90
7.7	Definition of Concepts and Examples (SP2)	91
7.8	Lesson Practical Example (SP2)	92
7.9	Lesson Practical Example - Continued (SP2)	93
7.10	Lesson Practical Example - Vulnerability Illustration (SP2)	94
7.11	Lesson Question Example	94
7.12	Example of Student's Incorrect Answer	95
7.13	Web Application Security Course and Quiz	97
8.1	Phase 4 - Verification Process Flow	103
8.2	Knowledge Verification Results (Percentages per Student)	105
8.3	Knowledge Verification Results per Secure Coding Practice and Related Questions	107
8.4	Knowledge Verification Results per Secure Coding Practice	108
8.5	Captured Data Example (P1)	111
9.1	Key Elements	120
9.2	A Generic Framework for Teaching Secure Coding Practices	122

Chapter 1

Introduction

The purpose of this chapter is to introduce the research problem, research questions and research objectives for this study. This chapter also outlines the structure for this research.



1.1 Theoretical Background

Nowadays many organisations use information to perform their daily activities. Both the private and public sectors rely on information systems to carry out their key operational activities. Previously, many software applications were desktop applications which could be used on standalone computers. The arrival of the internet has seen web application development gain significant attention. Presentation software and word processors are typical examples of desktop applications, whilst internet banking and online shopping carts on e-commerce websites can be considered to be web applications (Jeff, 2017). Desktop applications are usually installed on a single computer for a specific task, whereas web applications are made available on servers (Jeff, 2017). Thus, web applications are exposed to a number of threats on the internet that desktop applications may not be exposed to. In addition, web applications often handle very sensitive data, used for carrying out critical tasks such as banking, online shopping and online tax filing (Deepa & Thilagam, 2016). These applications are trusted by billions of users for performing such daily activities.

Web applications have received attention from academia and industry to initiate some defence mechanisms to protect them from security threats (Deepa & Thilagam, 2016). Handling risks related to the security of web applications is a major challenge for many organizations. Information security has thus become one of the top managerial priorities in many organisations (Benbasat, 2010). The need for securing any organisation's network resources has increased over the past decade. Many organisations address this need by including costs for firewalls and other network security controls in their budgets. However, 75% of all attacks on the internet are executed through the application layer of the OSI model (Customs Solutions Group, 2012). This indicates that more focus needs to be put on securing web applications.

More than 75% of web applications have vulnerabilities (Chandrasekar, Cleary, Cox, & O Gorman, 2017). Many of these web applications have common vulnerabilities which can be easily corrected (Zhu, Xie, Lipford, & Chu, 2014). Addressing many of these web application vulnerabilities is relatively straightforward through introducing secure coding practices. Academic institutions may teach programming, but because of limited resources they

often only focus on the correctness of code and end up overlooking security in the code (Bishop & Orvis, 2006). This oversight in addressing security issues is felt most severely during the maintenance phase of the Software Development Life Cycle (SDLC), which is costly (Customs Solutions Group, 2012). Furthermore, if academic institutions would integrate secure coding practices into undergraduate software development courses, students would be more likely to implement them. This in turn could improve security in web applications in industry.

Human aspects in information and cybersecurity usually consists of two major elements, namely knowledge-based and behavioural elements. Knowledge deals with the underlying requisite knowledge that would allow a person to behave in accordance with secure guidelines, whilst behavioural elements deal with everything else which if one can assume requisite knowledge, would make them apply it (Van Niekerk & Von Solms, 2010). In addition, while it is possible for a software development student to have the requisite knowledge and not behave appropriately, it is unlikely for someone to behave appropriately if they do not have the requisite knowledge.

1.2 Secure Coding Practices

The secure processing of information consists of the implementation of various controls in order to reduce risks caused by a specific vulnerability (Von Solms & Van Niekerk, 2013). In the case of web applications, there are various types of controls that could play a role in protecting information, including:

- Physical controls, for example, providing a lock on the computing facility.
- Technical controls, for example, an authentication mechanism.
- Operational controls which deal with human behaviour, for example, having to use the lock to prevent access to the computing facility or requiring authentication before allowing access to computing resources.

If the above-mentioned control types are not collectively addressed, it could lead to security breaches. While some of the controls may deal with end-user behaviour, others deal with the behaviour of Information Technology

(IT) staff in relation to configuration techniques (Chung et al., 2014). This includes programming staff who develop web applications and who should adhere to secure coding practices to mitigate risks associated with web application security. While a technical control would be a secure coding practice, an operational control would be the policy and procedures for a programmer to adhere to such secure coding practices.

In the case of web applications, one of the best sources for securing web applications is the Open Web Application Security Project (OWASP) (OWASP, 2017). OWASP is a non-profit organisation with the goal to improve the security of web applications. This organisation focusses on making security visible, so that individuals and organisations such as governments, companies and educational institutions have more knowledge when developing web applications (OWASP, 2017).

OWASP focusses on many development security platforms. This research study however uses OWASP security practices which apply to the Microsoft .NET framework. There are several ways in which programmers could introduce security flaws when using the .NET framework especially when working at the data access layer.

The controls that OWASP proposes are technical measures, while the programmers themselves having to adhere to them would be an operational measure. Operational controls fail when the human does not adhere to them. Humans will often not adhere to such a control owing to a lack of knowledge or because there is no procedural requirement forcing them to adhere to the control. Therefore, programmers should be educated about secure coding practices to ensure that they are aware of the vulnerabilities that may otherwise be missed or only realised at a stage when it might be too late. This behaviour should also be audited to ensure that programmers adhere to such secure coding practices. This research proposes a framework for teaching secure coding practices through a blended learning approach.

1.3 Problem Statement

More than 75% of web applications currently contain vulnerabilities (Chandrasekar et al., 2017). This raises the question as to whether this lack of adherence stems from a lack of formal education regarding secure coding

practices. This research addresses this problem by educating students about these secure coding practices. The first phase of this research was an investigation into whether students developing their capstone software development projects have adhered to secure coding practices.

The initial investigation confirmed that the students developing their capstone software development projects do not consistently adhere to secure coding practices. *Therefore, the problem this research addresses is that undergraduate software development students do not consistently adhere to secure coding practices when developing their third year capstone projects, thereby leading to vulnerabilities in their web applications.*

1.4 Primary Research Objective

In order to address the identified problem, the following primary objective has been identified:

- To develop a framework for teaching secure coding practices through a blended learning approach.

1.5 Secondary Research Objectives

In order to address the primary objective, the following secondary objectives were identified:

- *Secondary Objective 1:* To determine what secure coding practices a web application developer should adhere to in the .NET environment.
- *Secondary Objective 2:* To determine the adherence of third year software development capstone projects to the identified secure coding practices.
- *Secondary Objective 3:* To determine whether third year software development students have the requisite knowledge relating to secure coding practices.
- *Secondary Objective 4:* To design and implement an educational intervention to support software development students in the development of secure web applications.

- *Secondary Objective 5:* To determine the effect of the educational intervention on both student adherence and their requisite knowledge regarding secure coding practices.

1.6 Delineation

This project will specifically focus on third year software development students at the Nelson Mandela University, School of Information and Communication Technology whose capstone projects are web-based. Furthermore, it focused on the data access layer in the .NET framework.

1.7 Overview of Research Design

This research started with a literature review to determine which secure coding practices software developers should adhere to when developing web applications in the .NET environment and data access layer. The identified secure coding practices were guided the development of an instrument in the form of a checklist that was used in a code review to determine the adherence of past capstone software development projects to these secure coding practices. The results from the code review showed that, to a large extent, students do not consistently adhere to these secure coding practices. In order to determine the reason for the lack of adherence, a questionnaire was created to determine whether students who are currently doing the capstone project have the requisite knowledge regarding secure coding practices. The following phase of this research involved a literature review to determine the educational principles that a secure coding educational intervention should adhere to. This educational intervention involved blended learning lessons and a compliance monitoring instrument (checklist).

The blended learning lessons helped in teaching students the secure coding practices that a software developer should adhere to when developing web applications. The checklist was used to monitor adherence to secure coding practices when developing their web applications. Finally, the researcher determined the effect of the educational intervention by conducting a further code review and a questionnaire (post-test) to determine whether students have the requisite knowledge and to determine their adherence to the secure

coding practices included in the lessons.

Research Objective	Research Method
Primary Objective: To develop a framework for teaching secure coding practices through a blended learning approach.	Argumentation
Secondary Objective 1: To determine what secure coding practices a web application developer should adhere to in the .NET environment.	Literature review
Secondary Objective 2: To determine the adherence of third year software development capstone projects to the identified secure coding practices.	Code review
Secondary Objective 3: To determine whether third year software development students have the requisite knowledge relating to secure coding practices.	Questionnaire (Pre-test)
Secondary Objective 4: To design and implement an educational intervention to support software development students in the development of secure web applications.	Prototype (On-line lessons, Checklist)
Secondary Objective 5: To determine the effect of the educational intervention on both student adherence and their requisite knowledge regarding secure coding practices.	Questionnaire (Post-test, Code Review)

Table 1.1: Research Objectives and Related Research Methods

Table 1.1 provides an overview of the research methods used to achieve the research objectives for this study.

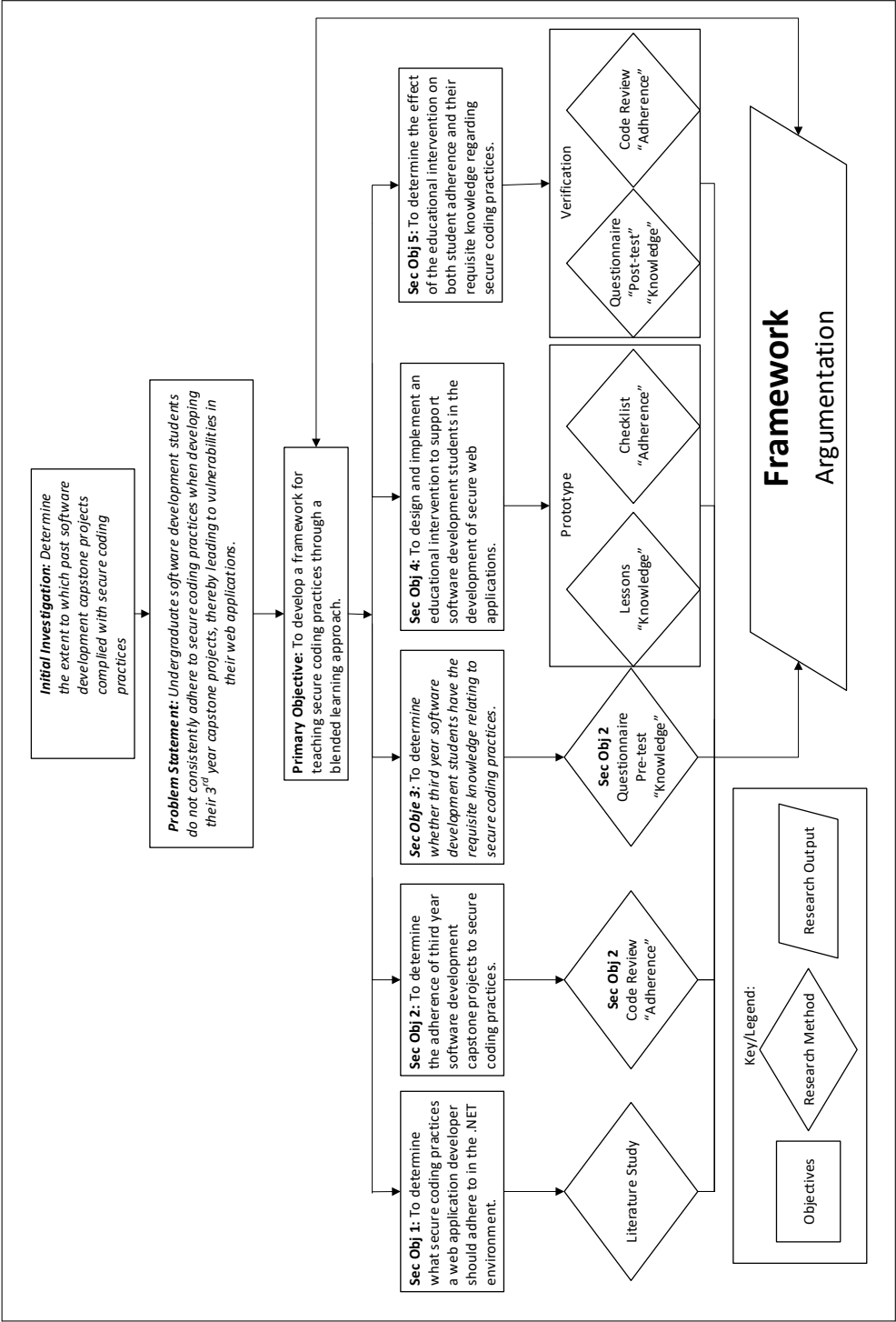


Figure 1.1: Research Design.

Figure 1.1 provides an outline of the design for this research starting with an initial investigation which led to the problem statement. Thereafter the primary research objective was determined together with the secondary

research objectives to address the research problem and the research methods used. For a detailed discussion relating to the research process and design, refer to Chapter ??.

1.8 Ethical Considerations

This research project adhered to all ethical requirements of the Nelson Mandela University and obtained ethics approval from the university research committee (REF H15-ENG-ITE-009).

1.9 Chapter Outline

Table 1.2 provides an overview of the chapter outline for this research.

Chapter	Description
Chapter 1 Introduction	This chapter introduces the research problem and the research objectives for this study.
Chapter 2 Web Application Security	This chapter provides background information and relevance for this study by examining web application security, and highlighting the role of the programmer in web application security.
Chapter 3 Theoretical Grounding	The purpose of this chapter is to provide an overview of the educational and behavioural monitoring theories which informed the design of the educational intervention for this research.
Chapter 4 Research Process and Design	This chapter provides the research process and design, highlighting the research setting and the various phases that comprise the contribution of this research.
Chapter 5 Behavioural Analysis	The solution for this research is split into four phases, which are represented in different chapters, (Chapters 5 to 8). This chapter provides details of Phase 1 of this research, which is the Behavioural Analysis phase of this study.
Chapter 6 Knowledge Assessment	This chapter provides details relating to Phase 2 of this research, which is the Knowledge Assessment phase of this research.
Chapter 7 Educational Intervention	This chapter provides details regarding Phase 3 of this research, which is the educational intervention. This chapter is split into two components, namely the knowledge and behavioural components.
Chapter 8 Verification	This chapter provides details about the final phase of this research, Phase 4, which is the verification phase. This chapter is also split into two parts, namely knowledge and behavioural verification components.
Chapter 9 Conclusion	Finally, Chapter 9 concludes this dissertation and presents the contribution of this research, namely the framework.

Table 1.2: Chapter Outline

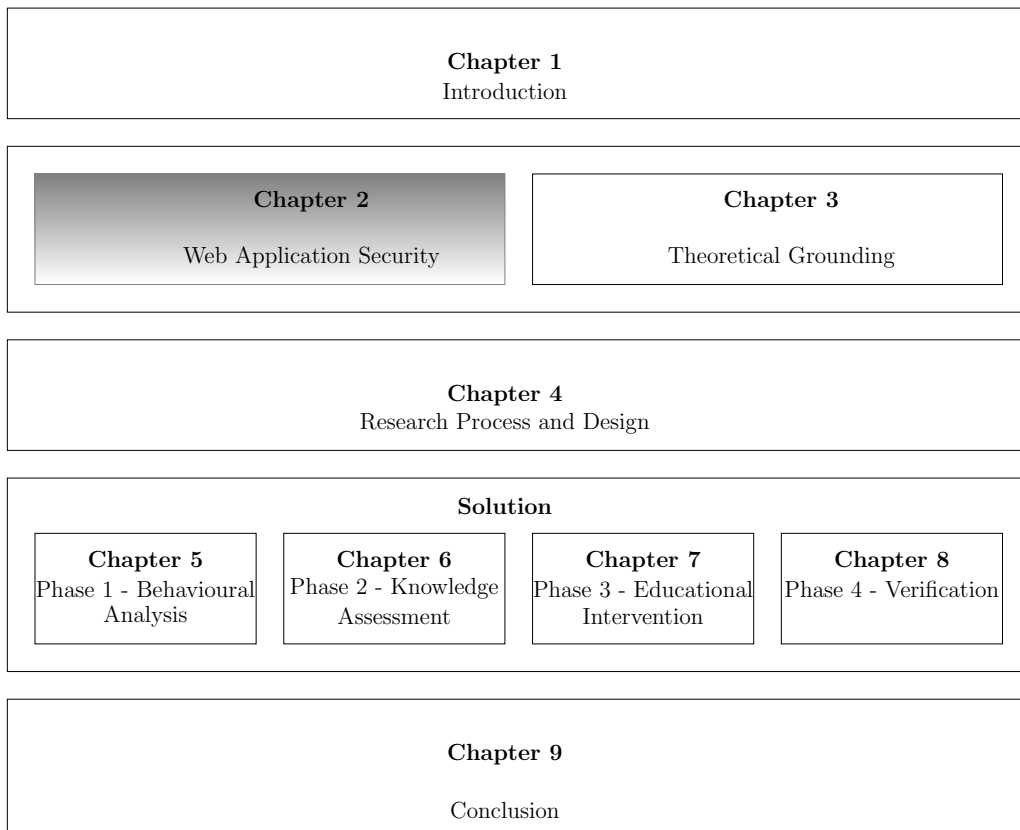
1.10 Conclusion

This chapter introduced the research area, which led to the research problem, and identified a list of objectives that need to be met in order for the research problem to be addressed. In addition, it provides an overview of the research design, including the research methods used to achieve the objectives identified. The next chapter discusses web application security.

Chapter 2

Web Application Security

The purpose of this chapter is to provide background information and relevance for the study. The chapter examines web application security and introduces the reader to the larger field of the study with the aim of highlighting the role of programmers in web application security.



2.1 Introduction

This chapter highlights the need for software developers to be educated with regards to secure coding practices. Firstly, it examines the history of software security in general in order to show how the needs in terms of security have evolved. As mentioned earlier, computing environments exist at various layers, ranging from physical hardware to the application software layer. In order to understand the complexity of securing such environments, this chapter discusses the OSI and TCP/IP reference models which are commonly used to explain these relationships. In addition, it discusses the application layer of OSI and TCP/IP reference models and how it relates to web applications. This chapter follows with an explanation of how web applications should be protected. Furthermore, it provides standards and best practices for web applications with a specific focus on OWASP which is key to this study.

2.2 Evolution of Computing

Since the invention of technology, the stage has not only been set for unprecedented integration of capabilities, but also as a medium for collaboration and interaction between individuals regardless of their physical location. The first electronic computing systems were mainframes. Mainframe computers were installed per single location. Thereafter, mini computers were introduced, which allowed for multiple systems per location (Lee, 2014). Client communication in mainframes involved using a stack of punch cards or paper tapes, which were fed into the computer.

It was then realised that some of the information did not need the computing power of mainframes or mini computers, but needed to only be accessed and presented to the client through a terminal or desktop. On the other hand, single board computers were becoming more powerful and evolved into a new class of computers, which were known as personal computers. Computing was distributed on the workstations because it was cost-effective and it supported situations where data and execution were distributed across one or more network (Hudli & Pidaparti, 1995). These mainframes and personal computers did not require much security other than that the actual hardware

was kept in a secure place.

With the introduction of networking technologies and the internet, software applications have shown extensive growth. In the network environment, several clients and a server existed, which resided on different computers, where the clients made requests to the server for computation (Hudli & Pidaparti, 1995). As technology evolved, there was an increase in software applications which were developed using an n-tier architecture.

The transition from mainframe-based computing infrastructure, through client-server architecture, to global connectivity in today's software applications has resulted in many new security threats and challenges (Bishop, 2003). Security problems and solutions in mainframes and mini computers were addressed in terms of securing files or processing on a single system (Bishop, 2002). Mainframe-based attacks targeted the actual computer. Protecting the computer from attacks was simply by locking the door where the computer is located. Thus a physical control was deemed sufficient for protecting the mainframe computer. Client-server computers required an authentication mechanism before a user could gain access or attack the computer. Securing them also required some form of technical controls which required users to be authenticated before using the computer. Software applications today require more security than client-server and mainframe-based computers. Security problems today focus on networked environments which are used by many users in different locations.

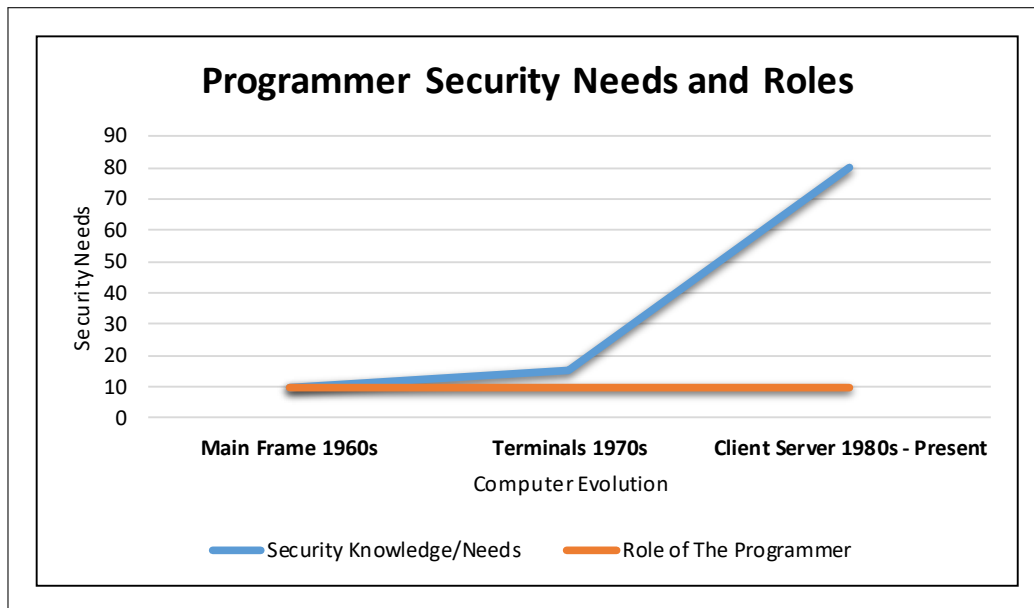


Figure 2.1: Security Needs and Roles of Programmers Over Time

Figure 2.1 shows an increase in the security needs and roles of a programmer over time. Figure 2.1 does not represent any true data, but tries to show the relationship between the role of the programmer and the security knowledge required for programmers throughout the computer evolution. As computing environments became more complex, the necessary controls to protect them also became more complex. Modern information systems exist at multiple layers ranging from physical hardware to connective network hardware all the way to the application software layer. In every level in the system, there has to be some form of security controls in place. Any level that fails when securing these systems negates the actual security. In a modern distributed web environment, which is more complex, everything exists in a different layer. One of the layers at the top of the OSI and TCP/IP reference models is the application software layer. The application layer includes the services that allow operating systems to run on it. The software applications that run on operating systems are used by end users. A software application will fail if end users do not adhere to operational controls when using the software application. They will also fail when people who develop these applications do not adhere to best practices for developing them. The people who develop software applications are typically referred to as programmers or software developers.

Programmers for mainframe and client-server computers focused on getting the machine to achieve its primary goal. Programmers of software applications need to focus on the primary goal of the application, and also focus on error handling and security requirements of the application (Bishop, 2003). Most software applications have different security requirements from those of mainframes and mini-computers. The security knowledge that a programmer of mainframe computers and mini computers required was therefore very different from what is expected of today's software developers. Today's software developers should have much more technical knowledge regarding software security as will be discussed in the following section.

2.3 What is Software Security?

Software applications are programs used by end users to perform tasks on a computer (Beal, 2018). In the early days of computing, these tasks were centrally located (Hugoson, 2008). When computing became decentralised, there was the need to install software applications on separate computers. Nowadays, there are various ways that these applications can be installed, deployed and managed. These range from being centrally located on internal servers to those installed on a device itself, to web-based applications which are published on servers.

Many software applications are used by businesses to store private business information. Software applications that store information are an integral part of information systems. These information systems not only store employee information, but also the knowledge, concepts, ideas and brands of the organisation as stated by ISO 27002 (ISO/IEC, 2013). Organisations need this information to carry out various business processes. This means that, like other business assets, information is a valuable organisational asset and should be protected from threats that are posed by vulnerabilities (Von Solms & Van Niekerk, 2013).

Information security is about ensuring that information is secure (Christopher, Choo, & Dehghantanha, 2016). What constitutes information security is commonly defined in terms of characteristics that information should have. These characteristics include confidentiality, integrity and availability (Rouse, 2018), as follows:

- Confidentiality refers to measures designed to prevent information from reaching unauthorised people. If a person who should not have access to confidential information gets access to it, then its confidentiality has been compromised.
- Integrity involves maintaining the consistency, accuracy and trustworthiness of the data through its life cycle. The data should only be manipulated by authorised people of a particular organisation.
- Availability ensures that information is available to authorised people at all times.

Threats posed by vulnerabilities can be addressed through introducing security controls (ISO/IEC, 2013). These controls can either be physical, technical or operational in nature. Physical controls deal with the physical aspects of the protection of information. An example of a physical control would be a lock on a computer or server room's door (Reid, Van Niekerk, & Von Solms, 2011). Technical controls include technologies that are used to protect information. Examples of technical controls include anti-virus software, firewalls and authenticating users when accessing systems. Operational controls include the administrative, managerial, and procedural controls and other types of controls that relate to the roles that humans play in protecting information. An example of an operational control would be a policy that requires users to logout when they leave their computers and locking the door when they leave their offices. This means that if humans fail to comply with the operational controls, both the physical and technical controls will fail and the information may be at risk.

In order for programmers to know what to do in terms of security, there has to be an operational control. For software applications, there have to be operational controls on how end users should use the software applications. Many researchers (Reid et al., 2011) believe that humans are the biggest problem in information and cybersecurity. Security controls may be applied differently to different employees even within the same department. This also applies to the IT department where there are end users, network designers and programmers. Security controls for end users would be fairly simple like logging out when leaving their computers. Security controls for network designers would include configuring the network correctly and securely. For

programmers, security controls could include checking that software is safe and is not vulnerable to known threats. For the purpose of this research, the focus will be on the security knowledge and behaviour required by programmers when developing software applications as will be discussed in the next section.

2.4 How to Secure Software

As discussed in Section 2.3, the security of software applications depends on IT staff. For software applications to function properly on a network, the network designers would need to configure the network correctly. There is a set of guidelines that is used by network designers to create and implement applications that run on the network (Ravali, 2013). These guidelines are provided by the Open Systems Interconnection (OSI) and Transmission Control Protocol and Internet Protocol (TCP/IP) reference models.

2.4.1 TCP/IP and OSI Reference Models

The OSI and TCP/IP reference models make use of a layered architecture to allow programmers and network designers to create and to implement their applications to run on a network. These models help to understand the functions of communication software relating to network design. The OSI reference model is defined in ISO 1105-2 (ISO/IEC, 2011). The TCP/IP was created by the Advanced Research Project Agency Network (ARPANET) (Aggarwal, Gupta, & Saxena, 2014).

These models also ensure the connection of multiple networks making it easy to meet requirements across different platforms (Aggarwal et al., 2014). These two models do not have an equal number of layers. Although the OSI model has seven layers and the TCP/IP model consists of four layers, the layers work very similarly for both models with each layer supporting the layer above it. In the next section, the layers of the two models are discussed.

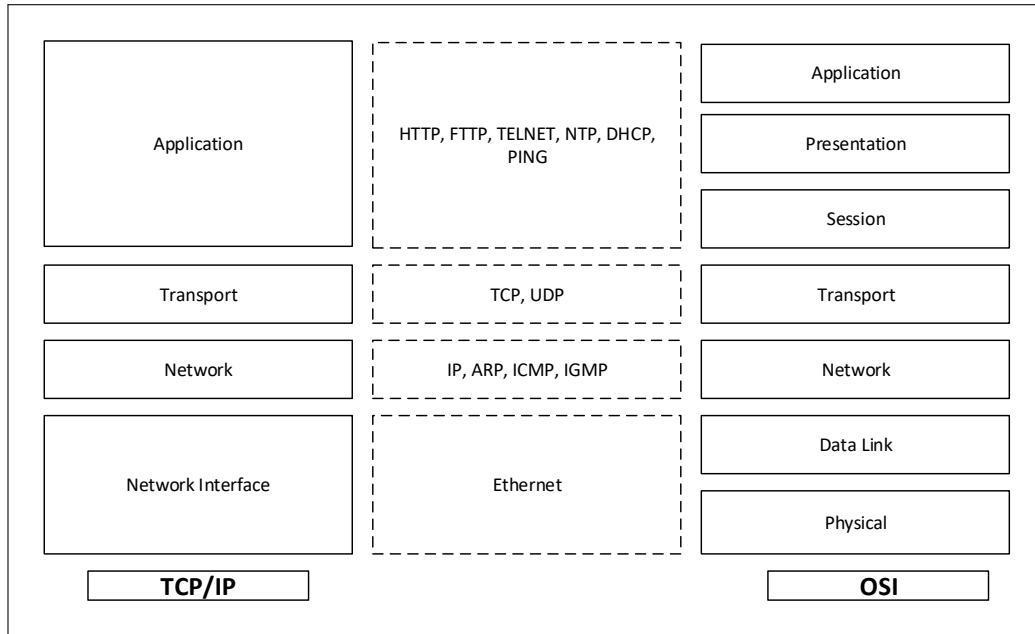


Figure 2.2: OSI and TCP/IP Reference Models

Figure 2.2 shows how the OSI and TCP/IP reference models relate and the services they provide for a software application to function. The OSI reference model consists of seven layers as follows:

- **Physical layer:** this is the lowest layer of the OSI model, which includes physical networking media like cable docking connectors. This layer is responsible for detecting voltage generated in order to send and receive signals carrying data (Li, Li, Cui, & Zhang, 2011).
- **Data link layer:** this layer provides the reliability of data between nodes. It produces frames of packages from the network layer and passes them to the physical layer. It also detects and corrects errors (Li et al., 2011).
- **Network layer:** the function of this layer is to deliver packets from the source to the destination across multiple links.
- **Transport layer:** this layer ensures that data is sent and ensures that it arrives intact and in order. This layer breaks the data into small units that make it easy for the network to handle it (Li et al., 2011).

- **Session layer:** the function of this layer is to establish a session between two parties, and maintain communication, controlling logging on and off, and user authentication (Li et al., 2011).
- **Presentation layer:** this layer is responsible for encrypting and decrypting data, like system password protection and data that is in binary (Aggarwal et al., 2014).
- **Application layer:** this is the last and top layer of the OSI reference model. The function of this layer is to provide an interface to allow programs to use network services (Li et al., 2011).

The OSI and TCP/IP reference models have layers with similar functions. TCP/IP has fewer layers and like the OSI reference model, starts from the bottom and moves to the top, as follows.

- **Network interface layer:** this is the lowest layer of the TCP/IP reference model and consists of two sublayers: namely the media access control sublayer and the physical sublayer. This layer corresponds with the OSI physical and data link layers (Aggarwal et al., 2014).
- **Internet layer:** the second layer of the TCP/IP reference model is the network or internet layer. Its function is to put the data into packets and is based on connectionless services. This layer is similar to the OSI network layer (Fujitsu, 2006).
- **Transport layer:** its function is to establish a session between source and destination machines so that data packets can exchange between them. It corresponds to the transport layer in the OSI model (Aggarwal et al., 2014).
- **Application layer:** TCP/IP's application layer corresponds to the OSI reference model's session, presentation and application layers. This layer includes protocols such as Hypertext Transfer Protocol (HTTP), which is the core protocol of the World Wide Web (WWW), File Transfer Protocol (FTP) which enables files to be sent and received to client from server, Telnet, Simple Mail Transfer Protocol (SMTP) and the Domain Name System (Ravali, 2013).

After considering the above reference models, it can be seen that although they do not have the same number of layers, they function very similarly. However, the OSI reference model is a complete, robust network architecture and the TCP/IP reference model is an architecture of the internet which is used to promote success in development (Li et al., 2011).

Both these models include network and application layers where a programmer and network designer would need to have knowledge of each layer respectively. This research focusses on the application layer which software applications run on as is discussed in the following section.

2.4.2 Application Layer

The application layer is the OSI and TCP/IP layer that interacts with software applications that implement a communicating component (Ravali, 2013). This layer is responsible for providing an interface to allow software applications to use network services. The application layer consists of many protocols. All protocols at this layer are application specific (Beal, 1999). Protocols help users to accomplish their tasks. Users use different types of software applications depending on the tasks they require to complete their goals. Application software consists of programs that the user interacts with to complete a certain task. The types of software applications include desktop, mobile and web applications.

- Desktop applications are software applications that are installed on stand-alone computers and that require operating systems that are compatible with that particular software. Examples of desktop software would be a word processor and a media player. A media player designed for a Microsoft Windows computer would only be installed on operating systems that support Microsoft Windows.
- Mobile applications are applications that are installed on mobile devices such as smartphones and tablets. Mobile applications also require applications to be compatible with the operating system of the mobile device. Examples of mobile applications are mobile banking applications and mobile messengers such as WhatsApp messenger.
- Web applications are published on a server. The same web application

can be viewed on computers with different operating systems. Examples of web applications are online booking systems and e-commerce web sites.

All these software applications need to be secure because they carry critical information of users and organisations. Therefore, developers of these applications need to consider security when developing these applications. For the purposes of this research, the focus is on securing web applications as is discussed in the following section.

2.5 How to Secure Web Applications

With the rise in the need for securing the information of the organisations, the focus has been on the network layer, with more than 76% of organisational spending on the network layer. However, more than 75% of attacks reported are founded on the application layer of the TCP/IP reference model (Customs Solutions Group, 2012).

More than 90,000 vulnerabilities have been recorded in the Symantec comprehensive vulnerability database over the past two decades, from 24,560 vendors representing over 78,900 products. There were more than 76% scanned web applications in 2014, of which 20% were critical and might have caused major problems for businesses if they were not detected. On average, over 340,000 web attacks were blocked from web applications per day in 2014 (Chandrasekar et al., 2017). This improved to 229,000 in 2016 (Chandrasekar et al., 2017).

This means that most attacks are no longer on the networks, but more on the software applications that run on the application layer. More than 76% of web applications contain vulnerabilities (Customs Solutions Group, 2012) and correcting these vulnerabilities is possible. If 76% of web applications contain known vulnerabilities, it means that 24% of the scanned web applications do not contain known vulnerabilities. Therefore, it is possible for web application to avoid known vulnerabilities. Those web applications without known vulnerabilities probably adhere to some form of best practice for secure software development. The next section discusses the standards and best practices that should be considered when developing secure web applications.

2.6 Standards and Best Practices for Secure Web Application Development

Various standards and best practices help in developing secure software applications. These standards help organisations in providing policies and help programmers by providing guidelines on securing code. Different standards and best practices provide different levels of detail depending on their target audience. There are organisations and institutions responsible for developing these standards and best practices. These include the American Standard, National Institute of Standards and Technology (NIST), the International Organizations for Standardization (ISO) and the International Electro-Technical Commission (IEC), (ISO/IEC) (Commission & Division, 2013; ISO/IEC, 2010), the Microsoft Developer Network (MSDN) which is a division which focusses on providing relationships with developers, testers, and analysts, and finally, the Open Web Application Project (OWASP) which provides best practices for improving security in web applications. These institutions provide different guidance in terms of web applications security, and the best practices for this research were selected based on the criteria as shown in Table 2.1.

STDs and BPs	Guidelines for web apps	Provide Guidance on security	Secure Coding guide-lines for .NET	Threat Scenarios and Exam-ples	Has many code plat-forms
ISO/IEC 23026	✓		✓		
ISO/IEC 27034	✓	✓		✓	
NIST SP-800-63-3		✓			
MSDN	✓	✓	✓		
OWASP	✓	✓	✓	✓	✓

Table 2.1: Standards and Best Practices Compared

To develop web applications that are usable and secure, there has to be some form of standard that a developer complies with. Developers who develop web applications without checking standards and best practices would pose challenges for web browsers and web pages owing to poorly developed web applications (ISO/IEC, 2010). There are various international standards and best practices available to guide software developers and network engineers in the development of web applications. These standards and best practices are shown in Table 2.1 and discussed as follows:

- The standard that deals with improving the productivity of web applications, together with the efficiency of development and maintenance practices, is the international standard ISO/IEC 23026 (ISO/IEC, 2010). This standard only focusses on how web applications should be developed. The standard provides a high-level overview of how security should be included in web applications.
- ISO/IEC 27034 provides guidelines on how to secure application security in general (Commission & Division, 2013). This standard also provides a high-level overview of security and is not specific for which types of applications should use this standard. This standard is for an entire organisation to provide security.
- The NIST 800-63-3 defines technical requirements for application systems security (Grassi, Garcia, & Fenton, 2017). This standard provides guidelines for application development. It also provides high-level guidelines on application software, but does not provide details on what can be done from a technical point of view.
- MSDN also provides guidelines on how to use libraries in a secure manner when developing applications, and specifically web applications (Wagner & Wenzel, 2015).
- OWASP provides information about security for many different types of development platforms. This project provides information on many web development platforms. Examples of projects that OWASP provides include the Java and .NET Projects to assist software developers specifically in these environments. In addition, OWASP provides re-

cent information on threats and specifically focuses on web application security (OWASP, 2017).

OWASP is suitable for this research since it is most comprehensive a providing security knowledge to students. OWASP is key to this research since it provides best practices for web applications as is discussed in the next section.

2.7 Open Web Application Security Project

OWASP is known by many organisations for its Top 10 Vulnerability List that it publishes and updates periodically (Customs Solutions Group, 2012; Li, 2013; Chung et al., 2014). This list focusses on identifying the most serious web application security vulnerabilities for many organisations (OWASP, 2017). The Top 10 list changes according to which vulnerability is most dominant at any given time.

Vulnerability	Description
SQL Injection	Injection flaws occur when untrusted data is sent to an interpreter as part of a command or query.
Broken Authentication	This relates to authentication and session management that are often implemented incorrectly. It allows attacks to compromise information, and session tokens or exploit other implementation flaws.
Sensitive Data Exposure	Many web applications and Application Programming Interfaces (APIs) do not properly protect data allowing attackers to steal or modify data.
XML External Entities	External entities can be used to disclose files using the file Uniform Resource Identifiers (URIs) handler, internal file shares, internal port scanning, remote code execution, and denial of service attacks.
Broken Access Control	Restrictions on what authenticated users are allowed to do are often not properly enforced, allowing attackers to exploit flaws to access unauthorized functionality, or data, such as other users' accounts or change access rights.
Secure Misconfiguration	This is mostly a result of insecure default configurations, incomplete or ad hoc configurations.
Cross-Site Scripting	Also known as XSS, it allows attackers to execute scripts in the victim's browser which can hijack user sessions, deface web sites, or redirect users to malicious sites.
Insecure Deserialisation	Insecure deserialisation leads to remote code execution, deserialisation flaws can also be used to perform tasks such as injection attacks, and privilege escalation attacks.
Using Components with Known Vulnerabilities	Components including libraries, frameworks and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such attacks can facilitate serious data loss or server take over.
Insufficient Logging and Monitoring	This allows attacks to further attack systems, maintain persistence, pivot to more systems, and temper, extract, or destroy data.

Table 2.2: OWASP Top 10 Vulnerability List 2017

Table 2.2 shows the Top 10 Vulnerability List provided by OWASP for web applications. The risk posed by each of these vulnerabilities can be reduced by more than one type of control. For the data access layer within .NET OWASP recommends controls for data access. These controls relate to some of the Top 10 vulnerabilities in Table 2.2. As an example, parameterised SQL commands or the use of stored procedures can reduce SQL injection. Therefore more than one control can reduce a vulnerability. The nine secure coding practices for data access are presented in Table 2.3.

SP Number	OWASP Secure Coding Practices
SP1	Use Parameterised SQL commands for all data access, without exception.
SP2	Do not use SQL command with a string made up of a concatenated SQL string.
SP3	Properly validate input fields.
SP4	Apply the Principle of Least Privilege when setting up the database of your choice.
SP5	When using SQL Server, prefer integrated authentication over SQL authentication.
SP6	Using stored procedures is the most effective way to counter the SQL injection vulnerability.
SP7	Encrypt connection strings.
SP8	Connection strings should be based in a configuration file.
SP9	Encrypt sensitive data using acceptable encryption algorithms.

Table 2.3: OWASP Secure Coding Practices

The secure coding practices shown in Table 2.3 are referred to using the numbering SP1 - SP9 throughout this dissertation. Considering the secure coding practices in Table 2.3, if one of them is not properly handled, it can be easy for an attacker to retrieve information that is in the database or to update the information. For example, if the connection string is found on other parts of the application and not locked in the configuration file, it can be easy for an attacker to access the information using the same connection

string to connect to the database. Or, if the expected values in an input field are not whitelisted in a system with concatenated SQL strings, attackers can use characters to manipulate the SQL string in the database and the information would be at risk.

These vulnerabilities cannot be prevented by programmers unless they know the types of flaws that exist in their code (Chi, Jones, & Brown, 2013). Similarly, they cannot implement the security controls unless they are taught how they work. However, it is essential that the secure coding practices be considered with the context of the application being developed.

2.8 Conclusion

Web applications are important in carrying out critical organisational activities. With these benefits come a variety of attacks, which target vulnerabilities in web applications. Vulnerabilities in web applications are often caused by software developers who do not adhere to secure coding practices or who lack the requisite knowledge relating to secure coding practices. Attacks which target these vulnerabilities in web applications can be addressed through the implementation of secure coding practices.

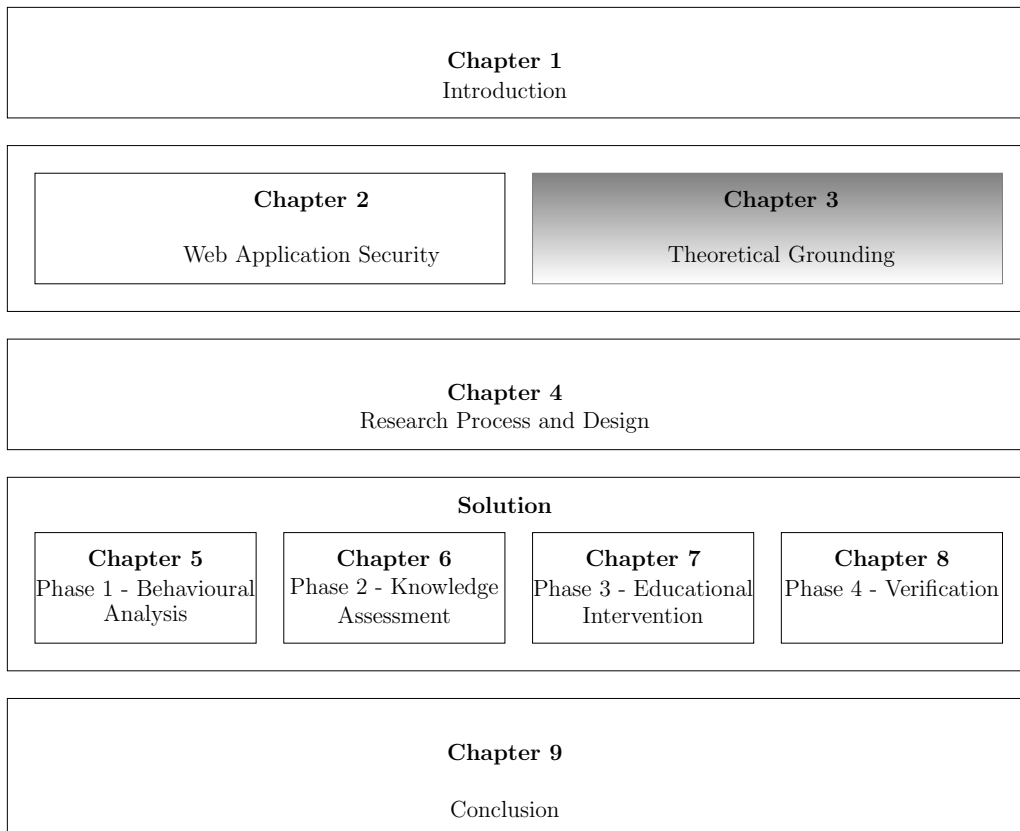
There are various standards and best practices related to secure coding practices for web applications as recommended by many organisations and institutes such as NIST, ISO/IEC, MSDN, and OWASP. The best practices provided by these organisations were evaluated and OWASP was considered the most relevant for identifying fundamental secure coding practices to be taught to software developers. Once software developers have been taught about secure coding practices, it is more likely that they will have the requisite knowledge. However, there has to be some form of compliance to monitor their behaviour, since it is known that people with the requisite knowledge do not always behave accordingly.

In order to educate and monitor the behaviour of software developers, underlying theories must be considered in order for the knowledge and monitoring to be effective. The next chapter provides the theoretical grounding for the educational and behavioural component of this research.

Chapter 3

Theoretical Grounding

This chapter provides an overview of the educational and behavioural compliance monitoring theories related to this research. The educational theory includes blended learning, and focusses specifically on online learning, which forms the basis of the educational part of the intervention. The behavioural compliance monitoring theories address the need for compliance monitoring and how it should be implemented.



3.1 Introduction

In South Africa, the increased demand for higher education has led to an increase in class sizes. This compels university lecturers to figure out how they can reach out to all the students as they did with small class sizes. Research shows that smaller class numbers have positive academic outcomes compared to large class numbers, since lecturers have less time to personalise the discussions in classes with large numbers (Van Niekerk & Webb, 2016; Blatchford, Bassett, & Brown, 2011).

One way that is commonly used to cater for large classroom sizes in higher educational institutions is to augment traditional classroom lectures with online experiences, which includes tutorials, laboratories, and assessments (Garrison & Vaughans, 2013). Integrating online learning material with the traditional classroom is referred to as blended-learning (Van Niekerk & Webb, 2016). In a blended learning environment, students participate in practicals where they can apply their learnt knowledge (Graham & Dziuban, 2008). They often use self-assessment tools to evaluate their progress in assigned practicals. Such tools can also provide lecturers with a means by which to monitor the behaviour of students.

The theoretical grounding of this research provides an understanding of educational and behavioural compliance monitoring theories, for students learning in a blended learning environment. The theoretical grounding informs the design of the intervention for this study, which is presented in Chapter 7.

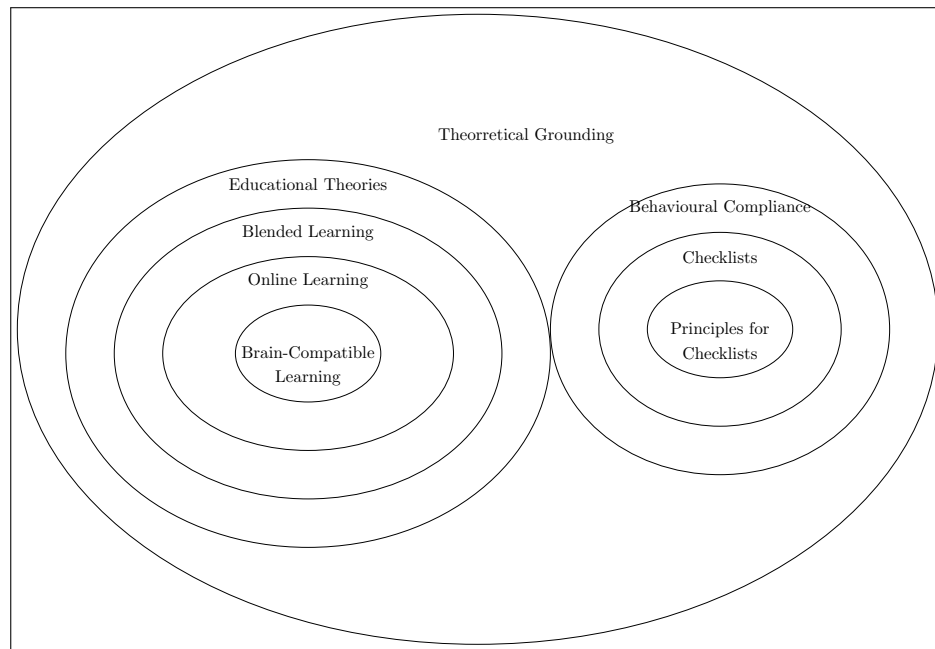


Figure 3.1: Theoretical Grounding

Figure 3.1 provides a roadmap of this chapter’s structure. This theoretical grounding chapter consists of two parts. The first part (Section 3.2) focusses on the theoretical grounding for educational theories, while Section 3.3 focusses on behavioural compliance monitoring theories. Finally this chapter is concluded in Section 3.4.

3.2 Educational Theories

This section provides an overview of the educational theories that inform the design of the educational intervention proposed in this research. In order to provide the reader with the necessary context, this chapter introduces the concept of blended learning, the role of online learning, and finally identifies the specific learning principles adhered to during the design of the educational intervention of this research.

3.2.1 Blended learning

For a long time, face-to-face lectures were the most commonly used teaching approach in higher educational institutions (Okaz, 2015). This has since

changed with the increased presence of innovations in technology. Technology has made it possible for students and lecturers to learn and communicate outside of the classroom. Rapid technological innovations have influenced the convergence between the traditional classroom and online learning environments (Graham & Dziuban, 2008). Traditionally, learning was facilitated fully through classroom experiences. This was done through face-to-face interaction between students and lecturers. Through advancements in technology, various technology-mediated instructions have been used to mediate learning experiences and interactions without requiring learners and instructors to be in the same location. More recently, in higher educational institutions, classroom-and technology-mediated instructions are often integrated such that the strengths of each are blended into a unique learning experience to improve the educational experience (Garrison & Vaughans, 2013). This integration of various learning experiences is a blended learning approach. Blended learning provides a place for collaborative learning and is flexible according to student preferences.

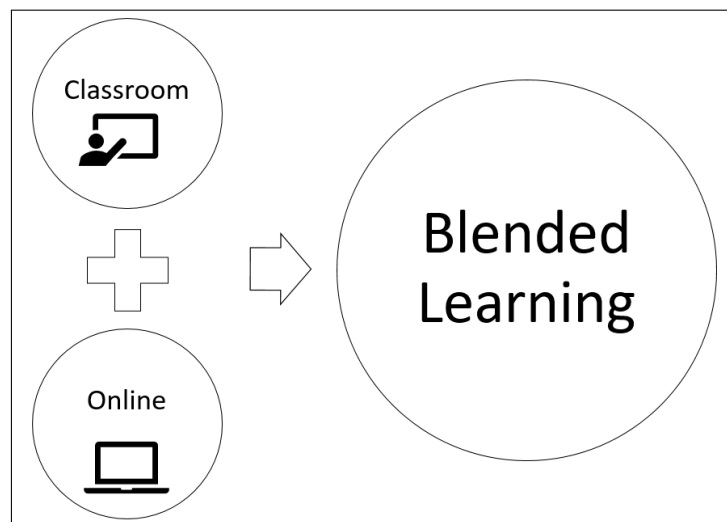


Figure 3.2: Blended Learning

Figure 3.2 illustrates the educational material that informs the design of the blended learning course for this research. In a blended learning environment, the lecturers have the decision of selecting from a variety of choices, depending on the learning context, and to determine the target skills that students should have at the end of the course (Okaz, 2015). It represents

a new approach and mix of classroom and online activities consistent with the goals of specific courses or programmes. This research focusses on the online learning component of blended learning because the proposed educational intervention uses online learning material to augment material that is currently taught in the classroom as is discussed in the next section.

3.2.2 What is Online Learning?

Higher education institutions are under pressure with the increased demands to maximise the number of students serviced by the education system, together with improving students' academic performance (Van Niekerk & Webb, 2016; Bakia, Shear, Toyama, & Lasseter, 2012). In South Africa there is a demand for institutions of higher learning to increase their annual student intake (Van Niekerk & Webb, 2016). As this demand increases, so do the technology needs (Torrise & Davis, 2000) as institutions of higher learning use various technologies to give instruction and to provide learning opportunities to students. In many cases, technologies provide a platform for online learning material which students and lecturers subscribe to.

The web has led to massive changes in the economy, in the way we communicate and relate to each other, and especially in the way we learn (Fawcett & Juliana, 2015). With the presence of the web, educators now use online platforms to assign activities to their students. Online learning has improved interaction between students and educators with the advances in technology (Reid & Van Niekerk, 2012).

Online learning was mostly used in distance learning environments. In a distance learning environment, students do not attend classes and they have the option to choose how and when they will learn, or do their activities (Al-Arimi, 2014). This type of learning is referred to as flexible learning, where students have flexibility in terms of the pace, and the time at which they would like to take assessments (Torrise & Davis, 2000). This is relevant to students in typical higher education institutions today, as these students are often adults who already have their own schedules and preferred time to complete their tasks (Van Niekerk & Webb, 2016).

Most university students do not perceive the classroom as motivating or supportive (Okaz, 2015). This has led to a decrease in participation and attendance in classroom learning and activities. The increased presence of

technology has been advantageous to university students, as they can learn and communicate in and out of the classroom environment. Higher educational lecturers need to redesign the learning process and to adjust their courses to accommodate such change. This will help students with different learning preferences and styles to get the right context in the correct form. This serves students better from a learning perspective, as it necessitates the need for an adoption of different learning pedagogies.

As online education is still a form of learning; it needs to adhere to sound pedagogical principles (Van Niekerk & Webb, 2016). There are many types of pedagogical principles, including personalised learning strategies, collaborative learning, constructivism, and **brain-compatible principles**. This research focusses on brain-compatible principles as is discussed in the next section.

3.2.3 Brain-compatible Educational Principles

Many higher education institutions have responded to the increase in student intake and improved academic performance with blended learning courses. This adds a further challenge on how best to deliver this content and to facilitate learning for students (Clemons, 2005). Student learning is mostly impacted by how their brains accept and process the information delivered. Therefore, brain-compatible educational principles should be adhered to in the design of online material. Brain-compatible educational principles promote a student centred approach, which is used to assure that the learning process is more effective and lasting (Tufekci & Demirel, 2009). Sound brain-compatible principles can provide a framework for learning and teaching that can assist in moving away from the traditional classroom approach (Caine & Caine, 1991). These principles provide a general understanding of the theoretical foundation for brain-compatible education (Caine & Caine, 1990). There is currently no single comprehensive list of brain-compatible principles. The researcher has combined various brain-compatible principles, which are provided by various researchers (Van Niekerk & Webb, 2016; Caine & Caine, 1990; Jensen, 2000) as listed in Table 3.1.

1. There is no long term retention without rehearsal.
2. Short, focused learning activities are best.
3. Learning is enhanced by challenge and inhibited by threat.
4. Emotions affect learning.
5. Learning involves both focused attention and peripheral perception.
6. The brain has a spatial memory and a set of systems for rote learning.
7. The brain simultaneously perceives parts and wholes.
8. Learning engages the entire physiology.
9. The brain is a parallel processor.
10. Learning is embedded in natural and social settings.
11. Each brain is unique.
12. The search for meaning is innate.
13. The search for meaning occurs through patterning.
14. Learning always involves both conscious and unconscious processes.
15. Learning with specific content is best.
16. Learning is a process of forming novel neural networks or patterns.
17. Learners need to recognise and connect patterns by themselves.
18. Novel patterns can only form as extensions of existing patterns.
19. Learning should be given choices to accommodate different learning styles.
20. Learning must apply to real life experiences of learning.
21. Immediate feedback amplifies learning.
22. Learning is collaborative and influenced by interactions with others.

Table 3.1: Comprehensive Brain-Compatible Educational Principles

Table 3.1 presents a more comprehensive list of brain-compatible educational principles. The brain-compatible principles that have been used in this study were selected from the list in Table 3.1. Table 3.2 presents the principles relevant to this study and are discussed further.

1. The search for meaning occurs through patterning.
2. Learning involves both focused attention and peripheral perception.
3. We understand best when facts and skills are embedded in natural, spatial memory.
4. Immediate feedback amplifies learning.
5. Learners should be given choices to accommodate different learning styles.
6. Learning with specific content is best.

Table 3.2: Brain-Compatible Principles Relevant to this Study

The brain-compatible principles in Table 3.1 are all effective for students learning in a blended learning environment. However, some of these principles can be impossible for a lecturer to control in a blended learning or a classroom environment. For example, principles like *Learning engages the entire physiology* and *The brain is a parallel processor* can be difficult to control, but it can be beneficial for students to understand in order to improve their learning. However, the principles in Table 3.2 guide the researcher when delivering the content, and guides the student to work through the content.

- **The search for meaning occurs through patterning:** In some way, students create patterns constantly with the information they learn. Lecturers should present information to the student in a way that allows them to extract patterns (Caine & Caine, 1990; Jensen, 2000). In order for the teaching process to be effective, the students should be able to create meaningful and personally relevant patterns (Caine & Caine, 1991). Lecturers should find out what knowledge the students already have at the beginning of the course, to adjust their lessons accordingly.
- **Learning involves both focussed attention and peripheral perception:** The brain absorbs the information to which it is directly paying attention, and also indirectly absorbs information signals that lie beyond the immediate focus of attention (Reid & Van Niekerk, 2012; Caine & Caine, 1990). Lecturers should organise the material that will be outside the focus of the learners’ attention. Art, illustrations and visuals should be added to present the content to the students (Jensen,

2000; Caine & Caine, 1990). The art and visuals should be changed frequently to reflect changes in the learning process. Music or subtle signals that emanate from the lecturer have an impact on learning.

- **We understand best when facts and skills are embedded in natural, spatial memory:** The teaching process should include real life activities, demonstrations or visual imaginary of certain experiences (Jensen, 2000).
- **Immediate feedback amplifies learning:** It is important to use early feedback to refine the students' patterning. Challenges and feedback are crucial ingredients for rewiring the brain (Jensen, 2000; Craig, 2003). Immediate feedback can increase student performance and produce more progress. This principle focusses on the lesson given. The lessons should be divided into smaller parts to allow students to move to the next lesson after understanding.
- **Learners should be given choices to accommodate different learning styles:** This principle enriches every learning experience and engages and motivates every student during the lesson or throughout the course (Reid & Van Niekerk, 2012). This includes giving the students time to problem solve, thereby allowing them to recognise and connect patterns for themselves instead of the lecturers having to provide all the answers (Jensen, 2000; Craig, 2003).
- **Learning with specific content is best:** Instructors should divide the course content into specific units and choose a concept that fits well with that block of content (Dunlosky, Rawson, Marsh, Nathan, & Willingham, 2013). These concepts should be planned well and it should be discussed how they fit together in order for the student to be able to process the information.

These brain-compatible educational principles have been identified to be key in designing the educational part of this research. The next section discusses the second key theory, which focusses on behavioural compliance monitoring.

3.3 Behavioural Compliance Monitoring

Since blended learning replaces some aspects of the traditional classroom with appropriate online experiences (including laboratories, tutorials, and assessments) through which students should be able to assess their own behaviour. These online experiences often do not come with immediate teacher feedback. That is why students should be able to assess their own behaviour. This allows them to check whether they have followed the correct steps when performing activities, since they would typically be working at their own at their own pace and at a time convenient to them. For example, during a practical session in a laboratory where students apply knowledge that has been learnt, they could use self-assessment tools to evaluate their progress and adherence to requirements.

These self-assessment tools can help in providing students with some immediate feedback. Student learning can be positively affected by providing feedback and evaluating students' performance (Mansour, 2015). Evaluating students' tasks and providing feedback can also affect the students' behaviour when performing a task, since they would be provided with correct guidelines on how to perform the task. The next section provides an overview of evaluation and feedback to students learning in a blended-learning environment.

3.3.1 Evaluation and Feedback

Providing students with feedback not only helps them to correct their mistakes, but also helps them to improve their learning and boosts their confidence when performing their tasks (Finn, Thomas, & Rawson, 2018; Mansour, 2015). Evaluating students and providing feedback is a core activity within the teaching and learning process (Mansour, 2015). In a blended-learning environment where students would mostly be working on their own, providing them with feedback could be challenging for the instructors. Therefore, developing reflections or self-assessment tools, which identify the desirable criteria or standards to which students should adhere when performing a task, would assist both students and instructors (Nicol & Macfarlane-Dick, 2006). Since students would have been taught the theoretical aspects of the tasks, the feedback criteria would have to be simple for them to understand and would act as a memory aid for them.

Providing students with self-assessment tools such as rubrics, scripts or checklists, which they can use to plan, monitor and self-assess, can improve their performance (Panadero, Jonsson, & Botella, 2017). Since checklists are a simple and effective form of self-assessment, researchers recommend them for students' self-assessment (Bishop, 2002; Cross, 2007; ISO/IEC, 2010). Checklists and their relevance to this research are discussed in the next section.

3.3.2 Checklist

Checklists have been considered to be one of the most common and simple forms of inspection for students used in software development (Jiménez, Merodio, & Sanz, 2017; Keil, Li, Mathiassen, & Zheng, 2008). Checklists can be given to students before they start their tasks, as it is important to establish a set of guidelines before engaging in a project or an activity (Jiménez et al., 2017). These guidelines help to determine the set of requirements that people should satisfy and the means by which to check behavioural compliance with such requirements.

Requirements that software development students should adhere to typically include functional, non-functional and security requirements. Whereas functional requirements specify the expected output of an end product, non-functional requirements specify the criteria that can be used to judge the operation of an end product. Security requirements specify the safety mechanisms that should be put in place for the product to perform tasks in a secure manner. In software development, software practitioners often use checklists to identify and assess risks in their project quickly (Keil et al., 2008; Bishop & Orvis, 2006).

There are many types of checklists that can be used for various tasks. These checklists include, but are not limited to, procedure checklists, feature checklists and evaluation checklists. Evaluation checklists are more prevalent in software inspection and are therefore considered most relevant to this research. The criteria in evaluation checklists are used to check against activities that have been completed (Jiménez et al., 2017). Various guidelines need to be followed when creating a checklist to produce one that is effective and usable (Wilson, 2013; Jiménez et al., 2017; Keil et al., 2008; Landoll, 2006), including the following:

- **Checklists should have a limited length:** Checklists should not be too long or too short. They should be limited to a single page, and should contain no more than 25 elements (Wilson, 2013; Jiménez et al., 2017). Checklists should only cover a certain range of aspects. This will help the users so as to have a sufficient list for verification (Jiménez et al., 2017). Checklist length should be considered in the context of understandability, clarity, and motivation of the users (Wilson, 2013; Jiménez et al., 2017). For checklists that exceed the limit, their items should be broken down into more specific objectives (Jiménez et al., 2017).
- **Checklist items should be phrased as questions:** A checklist can be developed in a manner that users ask themselves questions and provide answers. Items can also be phrased as imperative statements (Wilson, 2013), for example, a statement to verify that a certain activity has been completed. In addition, the questions in the checklist should not be too general, for example, the questions should contain Yes/No/Not Sure answers (Jiménez et al., 2017).
- **Provide an option for items not relevant:** In some situations, not all items of the checklist will be relevant (Wilson, 2013; Van Voorn, Verburg, Kunseler, Vader, & Janssen, 2016). Therefore, options such as "Not Applicable" or "Not Relevant" should be included in some items. This helps users not to go back to create the irrelevant items to satisfy the need of the checklist.
- **Checklists should be used as a memory aid:** Checklists should not be used to teach concepts or to provide information on how they relate. Rather, they should be a reminder to the users who already understand the content that is in the checklist (Landoll, 2006). Checklists should be simple and should focus on the known information or terms.
- **Checklists help in ensuring accuracy and completeness:** Checklists are predefined guidelines, questions and tasks that specify what is expected towards the end of an activity (Wilson, 2013; Landoll, 2006). Checklists can be used to check adherence to set objectives in an activity or project.

The above-mentioned guidelines were used in this research when designing the behavioural monitoring tool. The behavioural monitoring tool for this research will be the same throughout the different phases of the research, even though it will be used by different stakeholders.

3.4 Conclusion

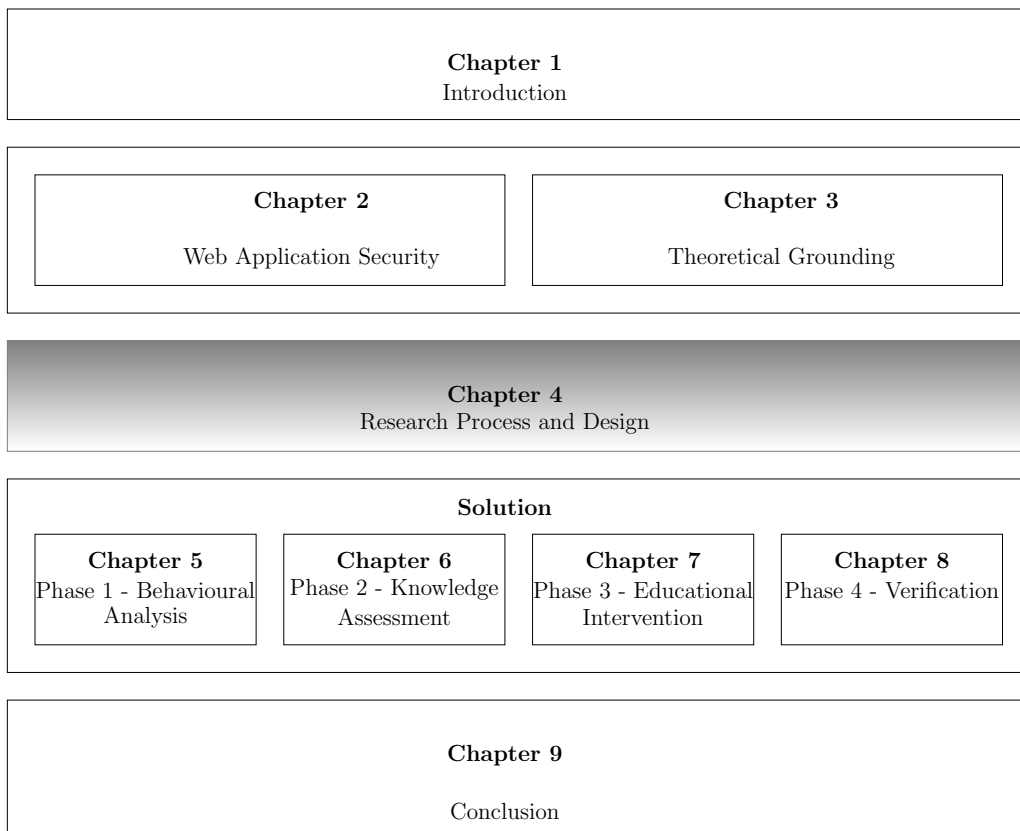
This chapter discussed the theoretical grounding for the intervention of this research. The first part of this chapter presented an overview of educational theories, while the second part focussed on an overview of behavioural compliance monitoring. The increased demands in higher education have led to the introduction of educational approaches that can suit various students' needs.

An educational approach that suits many students is blended learning, which integrates online and traditional classroom learning. Online education can be used to reach more students and to cater for various learning styles. Learning involves the human brain, therefore brain-compatible principles have been used in the design of the educational intervention proposed by this research. However, online learning alone cannot guarantee that students will perform accordingly. There is therefore a need for a behavioural compliance monitoring mechanisms for students in a blended learning environment since they would often be working independently. This behavioural compliance monitoring can be performed through the use of a checklist. Checklist guidelines need to be followed in order to create good usable and effective checklists. The next chapter presents the research process that has been followed in this research.

Chapter 4

Research Process and Design

This chapter provides an overview of the research process and design for this study. This includes a discussion of the research setting and an explanation of the various phases that comprise the contribution of this research.



4.1 Introduction

The research process and design provides an overview of the objectives for a research project, specifying sources for data collection, and discussing how the researcher plans to conduct the research (Saunders, Lewis, & Thornhill, 2009; Mafuwane, 2003). The research process and design presented in this chapter provides an overview of the steps followed in this study. It introduces the reader to the chapters that follow and provides the reader with the necessary context needed to understand the setting of the research undertaken. The next section presents the research approach that this research follows. Section 4.3 provides the research setting while Section 4.4 discusses the research process followed. Lastly, the conclusion of this chapter follows in Section 4.5.

4.2 Research Approach

Research entails finding out about information that is not known by people, thereby advancing the frontiers of knowledge (Walliman, 2011). In order to carry out research, there are tools that need to be involved, which are referred to as research methods. Research methods provide the researcher with ways to collect, sort and analyse data to come up with meaningful conclusions (Walliman, 2011; Macdonald & Headlam, 2009). In order to know which research methods to follow, the researcher needs to know which research approach they will follow. There are two types of research approaches, namely quantitative and qualitative research (Creswell, 2009). A quantitative research approach quantifies data, generalising results from a sample of the population of interest (Creswell, 2009; Macdonald & Headlam, 2009). With a quantitative research method, results are presented as numeric data, stating, for example *how long, how many, the degree to which, etc* (Moriarty, 2011; Macdonald & Headlam, 2009). The qualitative research approach, on the other hand, sets out to gain an understanding of the underlying reasons and motivation for actions and tries to establish how people interpret their experiences and the world around them (Macdonald & Headlam, 2009). Qualitative research collects visual and textual data for interpretation purposes.

In terms of this research, the data collection techniques used are quantitative in nature. However, the interpretation of the results take a more qualitative research approach using descriptive analysis. Therefore, the researcher does not prescribe either a strictly qualitative or strictly quantitative research approach. The researcher has chosen the research methods based on their sustainability for the specific task at hand (Creswell, 2007). Combining research methods in a research study is known as the mixed methods approach. This research adopted a mixed methods approach. The next section focusses on the research setting.

4.3 Research Setting

This research was conducted at the Nelson Mandela University. The Nelson Mandela University is a comprehensive institution, offering both academic degrees and vocational qualifications. In this case, the sample was drawn from students registered for the National Diploma: Software Development. This diploma is a vocational qualification focussing on providing prospective software developers with the requisite skills for professional work. In South Africa, there are no locally recognised curriculum guidelines for computing qualifications. The Nelson Mandela University therefore relies on the recommendations provided in global computing curriculum publications. The Association for Computing Machinery (ACM) has been producing such curriculum guidance for many decades.

4.3.1 Curriculum Guidelines

The ACM is an educational initiative which produces and updates curriculum recommendations utilised by computing disciplines world wide (ACM, 2018). This educational initiative continues to provide curriculum guidelines to the rapidly changing landscape of computing technology, and provides curriculum reports for updates in curriculum recommendations. The computing curriculum report overview (Shackelford, Cross, Davies, Impagliazzo, & Kamali, 2005) provides curriculum guidelines for sub-disciplines of computing, which includes computing engineering, computer science, information systems, software engineering and information technology. Additional to the

ACM educational initiative, the Cybersecurity Education (CSEC2017) curriculum guidelines assist academic departments across computing disciplines in providing cybersecurity knowledge to existing or newly formed computing qualifications (Burley et al., 2017).

Since the software development qualification at the Nelson Mandela University is part of the sub-disciplines of the ACM education initiative, the curriculum guidelines that this qualification has been based on is the Information Technology 2008 (IT2008) curriculum guidelines (Lunt et al., 2008). Security in this curriculum recommendation is broadly mentioned as **Information Assurance and Security** as a knowledge area with little detail on how to equip students with security knowledge. The updated version for this curriculum, which is the IT2017 (Lunt, Sabin, Hala, Impagliazzo, & Zhang, 2017), also broadly mentions security, but it emphasises the need to integrate cyber-security in the IT curriculum. This has been made easy through the ACM's expansion to include the first set of global curriculum guidelines for cyber-security education (Burley et al., 2017). The CSEC2017 curricula guidelines in **Chapter 4, Content of the cybersecurity curricula framework** in Section 4.2 **Knowledge Area: Software Security**, emphasises the importance of secure software and in Section 4.2.1 **Knowledge Units and Topics** provides knowledge units which align with OWASP. These knowledge units will help students to familiarise themselves with secure coding guidelines to developing web applications (Burley et al., 2017). The IT2008 and the more recent IT2017 curriculum guidelines also require students in computing and engineering disciplines to engage in a capstone project during their final year of study. The following section provides a discussion regarding the software development capstone project at the Nelson Mandela University.

4.3.2 Capstone Projects

Capstone projects are interactive assignments that students perform during their final year of study (Lunt et al., 2017). These capstone projects can take place over one or two semesters during a students' final year. Capstone projects should typically adhere to the following:

- Project groups of 3 to 5 students;

- Based on a real-world problem;
- Must be integrative;
- Students should have completed most of the curriculum before attempting the project;

Capstone projects in software development should consolidate and integrate all the skills gained throughout the software development qualification. Therefore, students engaging in the capstone project need first to have completed specific subjects. The subjects at the Nelson Mandela University introduce students to programming and business application systems development. The subjects for this qualification are all offered by the Department of Information Technology at the Nelson Mandela University (University Nelson Mandela, 2018). Figure 4.3.2 presents the subjects supporting the software development capstone project at the Nelson Mandela University.

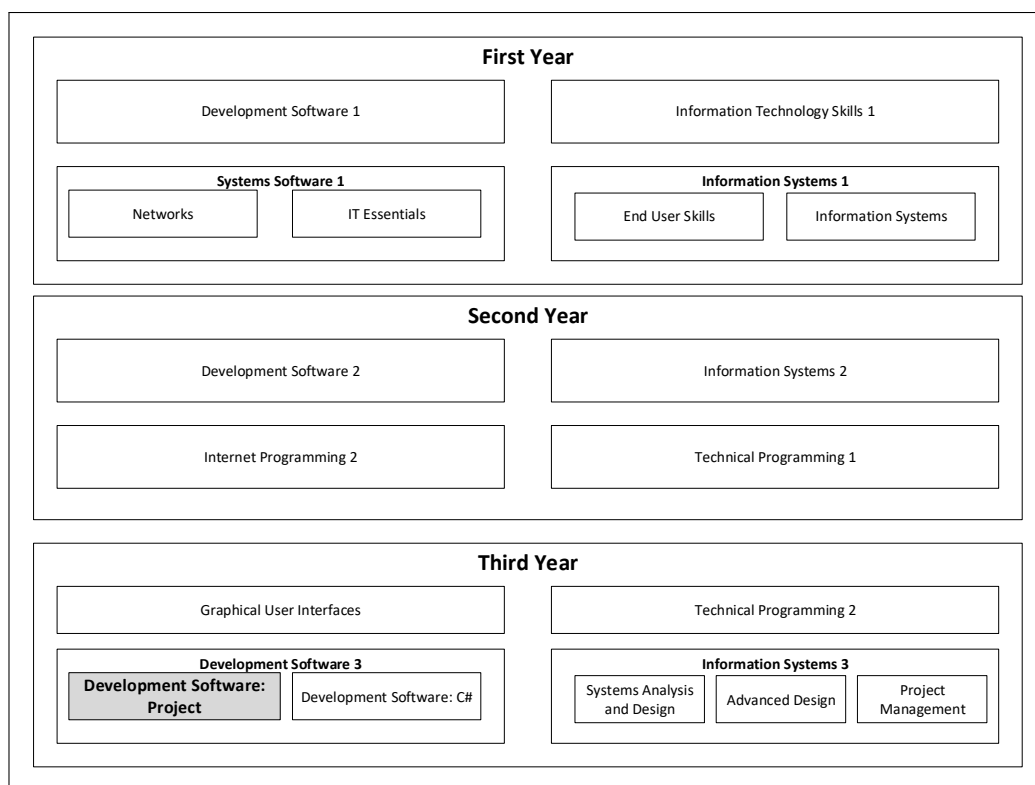


Figure 4.1: Subjects for the Diploma: Software Development

As shown in Figure 4.3.2, first year IT students register for Development

Software 1, which is a programming subject. This subject teaches students programming fundamentals, problem solving techniques and basic programming. This is achieved using the .NET framework's C# language. Information Technology Skills 1 equips students with the basic business skills, soft skills, self-management, and technical skills needed to provide effective services to customers and support in a technical environment. In the Networks subject, students are provided with a solid grounding in the fundamental networking concepts and technologies. The IT Essentials subject provides a fundamental understanding of hardware and software components for students, while Information Systems 1 focusses on teaching students programming logic, basics in systems analysis and design, and database design. The End User Skills subject equips students with the necessary computer literacy skills to be productive end users.

During the students' second year in the course, they register for Development Software 2 and Information Systems 2. Both of these subjects aim at teaching students database systems, advanced problem solving techniques, and allows them to use their programming skills when solving real-world problems. Additional to these second year subjects, students are also required to register for Internet Programming 2. This subject focusses on web-based application development techniques and technologies. It also focusses on programming from client and server perspectives. Students are taught in the .NET framework with ASP.NET.

In the students' third year of study, they register for Development Software 3, which is split into 2 modules; the capstone project and Development Software 3, which expands the students' knowledge regarding C# programming and the .NET framework. In the Technical Programming subject, students are introduced to other programming languages, such as PHP and Android reinforcing the techniques learnt in the other programming subjects. The graphical User Interfaces provide subject students with a solid foundation in human-computer interaction theory and an understanding of how to apply the theory to interaction design. Information Systems 3 has three sub-modules within it, namely System Analysis and Design, Advanced Design, and Project Management. System Analysis and Design provides an in-depth knowledge of information systems analysis and design, while Advanced Design provides students with an understanding of systems analysis

and design following an object oriented approach using Unified Modeling Language (UML) methodology. In Project Management, students are provided with an understanding of processes, tools and techniques, as well as areas of knowledge needed to manage IT projects successfully, which helps students to manage their capstone projects.

When students choose their capstone projects, many of them focus on mobile and web applications or gaming, while a few develop desktop applications. The capstone project is an ideal place to determine whether students adhere to secure coding practices when developing web applications. Although students are taught specifically to develop software in a Windows environment using the .NET framework, students may develop their capstone projects in the programming language of their choice. Most project students choose web applications in the .NET development environment as this is where their skills lie.

4.3.3 Participating Sampling

The participant sample for this study was drawn from the third year Software Development students using purposive and convenience sampling. In purposive sampling, the researcher groups the participants according to pre-selected criteria relevant to a particular research objective (Mack, Woodsong, M. MacQueen, Guest, & Namey, 2011). In convenience sampling, researchers select the cases that are easiest to obtain for the sample (Saunders et al., 2009; Bryman, 2013). The sample for this research was purposive in the sense that they met the criteria for the research project, and they were convenient in the sense that the researcher has access to the students. The criteria for selecting the samples of this research were split into two, as Phase 1 had different criteria from Phases 2, 3, and 4, which had the same criteria. The criteria for the different samples were as follows:

- **Phase 1:** behavioural analysis criteria required projects that were web applications, and developed in the .NET framework as this phase was conducted based on past capstone projects.
- **Phase 2, 3 and 4:** students had to be enrolled for the National Diploma: Software Development and they had to be doing their third

year capstone projects. In addition, these students' projects need to be web applications developed using the .NET framework.

4.4 Research Process

In order to address a problem or answer a question, researchers should follow a process. This helps to achieve the set research objectives. The process that a research study follows is determined by the research objectives and the nature of the research.

This research followed a phased approach with four main phases, as shown in Figure 4.4. These phases included a behavioural analysis, knowledge assessment, educational intervention, and verification. Sub-sections 4.4.1 to 4.4.4 provide a high-level overview of these phases. Each phase is discussed in depth in Chapters 5 to 8 respectively.

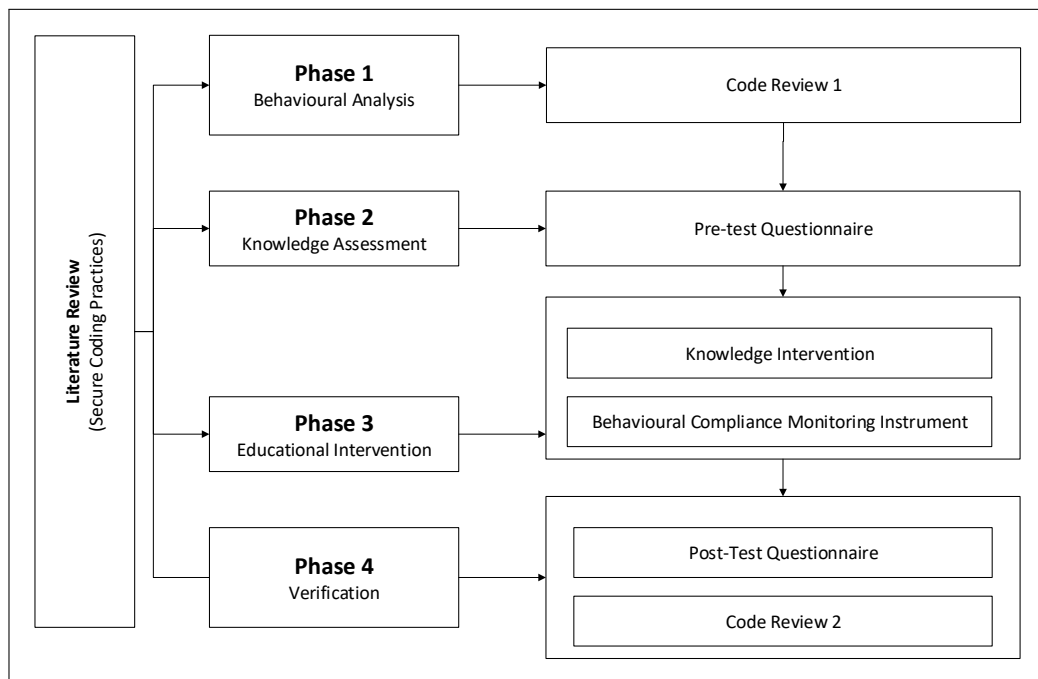


Figure 4.2: Phased Research Approach

The next section provides an overview of Phase 1, the behavioural analysis phase of this research.

4.4.1 Phase 1: Behavioural Analysis

In order to determine students' adherence to secure coding practices, a code review was conducted on past third year capstone projects using the identified secure coding practices. Code reviews are one of the most effective ways of improving software quality (Smartbear, 2009; Holzmann, 2009).

Performing a code review on software produces software with fewer defects (Smartbear, 2009). Code reviews are also used to assure that code complies with project coding guidelines and meets institutional coding standards (Holzmann, 2009). Different types of code review tools can be used depending on the sizes of the project being reviewed (Holzmann, 2009). For example, a project with millions of lines of code can require additional software to automate the code review process.

However, small projects such as capstone projects can easily be conducted using simple tools such as checklists and completed within a short space of time. Therefore, choosing tools for a code review depends on the scope of the project. Since checklists are simple to implement and are effective in code reviews (Cohen, 2010), they were used in this research to conduct a code review on past students' capstone projects.

The literature regarding checklists is discussed in detail in Chapter 3, Section 3.3.2. The results from the initial code review confirmed the problem statement for this research that *undergraduate software development students do not consistently adhere to secure coding practices when developing their third year capstone projects, thereby leading to vulnerabilities in their web applications*. This phase is discussed in detail in Chapter 5.

This phase of the study addressed both Secondary Objectives 1 and 2, which are *to determine what secure coding practices a web application developer should adhere to in the .NET environment and to determine the adherence of third year software development capstone projects to the identified secure coding practices*.

4.4.2 Phase 2: Knowledge Assessment

In order to determine whether students engaging in the capstone project have the requisite knowledge regarding secure coding practices, the researcher used a questionnaire. Questionnaires are one of the primary data collection

instruments used in research (Bowling, 2005). Questionnaires are mostly used to acquire information from large groups of respondents (Bird, 2009; Milne, 1994). They are developed for acquiring information about new products, for evaluating or redesigning existing products or processes (Giesen, Meertens, Vis-visschers, & Beukenhorst, 2012). Questionnaires are simple to implement and are quicker than other data collection methods such as interviews and focus groups. In questionnaires with closed-ended questions, collected data is more or less in the same format, which makes the collected data comparable within the data set (Bird, 2009).

A questionnaire was used in this research, which was given to third year capstone project students as a pre-test to determine their existing knowledge regarding secure coding practices. The results from the pre-test indicated that students generally lack the requisite knowledge regarding secure coding practices. This phase is discussed in detail in Chapter 6. This phase meets Secondary Objective 3 which is *to determine whether third year software development students have the requisite knowledge relating to secure coding practices*.

4.4.3 Phase 3: Educational Intervention

In order to address the problem of students' non-adherence and lack of knowledge regarding secure coding practices, the researcher designed an educational intervention to address both the students' knowledge and behavioural aspects using prototyping. Prototypes are built for testing concepts, processes and can also be used to learn from (Gero, 1990). In terms of this research, prototyping is used as a learning platform for students and is divided into two components, namely, knowledge and behavioral components. In order to address the knowledge aspect, the researcher created online lessons to educate students. The lessons adhered to brain-compatible learning principles as discussed in Chapter 3 Section 3.2.3. In order to address students' behaviour, a checklist was provided to the students for them to check their adherence to secure coding practices. The theory for the design of the checklist is discussed in Chapter 3, Section 3.3.2. This checklist was the same as that used for code reviews as conducted in Phase 1. This phase is discussed in detail in Chapter 7 and addresses Secondary Objective 4 which was *to design and implement an educational intervention to support software development*

students in the development of secure web applications.

4.4.4 Phase 4: Verification

In order to determine the effect of the educational intervention on students' knowledge and behaviour, the researcher designed a post-test, which also took the form of a questionnaire to test their knowledge. In addition, the checklist was used to conduct a further code review on the student projects. This phase is discussed in more detail in Chapter 8. This phase represents Secondary Objective 5 which was *to determine the effect of the educational intervention on both student adherence and their requisite knowledge regarding secure coding practices.*

The next section concludes this chapter.

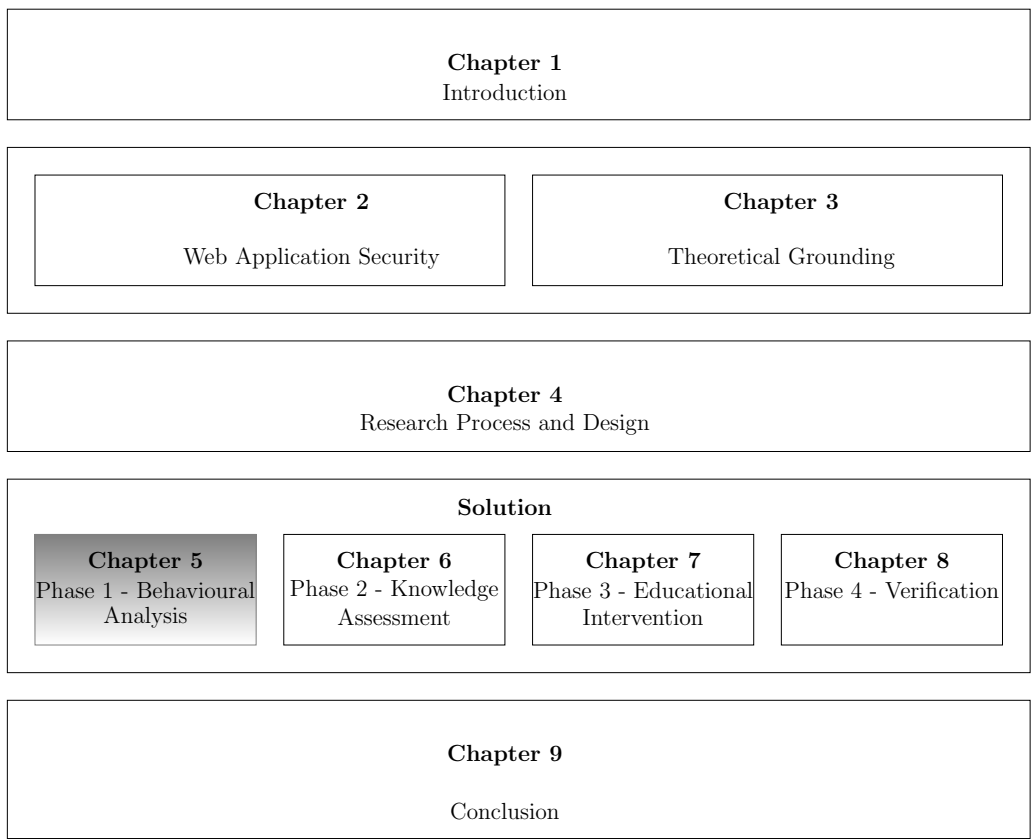
4.5 Conclusion

This chapter provided a road-map for the research process followed in this study. This research was conducted at the Nelson Mandela University with third year students registered for the National Diploma: Software Development. As mentioned in Section 4.4, each phase represents a chapter, which discusses that phase in detail. The next chapter discusses Phase 1 of the research, which includes determining secure coding practices and the code review of 2015 capstone projects.

Chapter 5

Behavioural Analysis

The purpose of this chapter is to provide an overview of the research process followed in the behavioural analysis phase of this research. This phase addresses the second secondary objectives, which is to determine the adherence of third year software development capstone projects to the identified secure coding practices.



5.1 Introduction

As discussed in Chapter 4, the main phases of this research are presented in Chapters 5 to 8. This chapter discusses the behavioural analysis phase. This was the first phase of this research, which set out to determine the extent to which third year students adhered to secure coding practices when developing their capstone projects. In order to establish what secure coding practices these students should adhere to, the researcher conducted a literature review to determine the secure coding practices for the data access layer of web applications in the .NET environment. These secure coding practices are discussed in Chapter 2, Section 2.7.

Firstly, this chapter provides the process flow for the behavioural analysis phase. Section 5.3 describes the design of the code review instrument, while conducting the behavioural analysis follows in Section 5.4. The results for the behavioural analysis follows in Section 5.5 and the discussion of the results in Section 5.6. Lastly, Section 5.7 concludes this chapter.

5.2 Behavioural Analysis Process Flow

The behavioural analysis phase for this research is explained in this section.

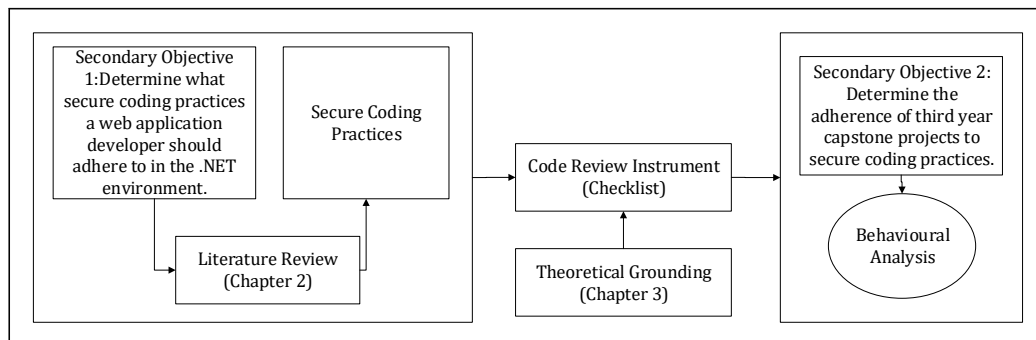


Figure 5.1: Phase 1 - Behavioural Analysis Process Flow

Figure 5.1 provides an overview of the steps followed in the first phase of this research. This phase started by addressing the first secondary objective, which was *to determine what secure coding practices a web application developer should adhere to in the .NET environment*. The secure coding practices that were determined in Secondary Objective 1 are discussed in Chapter 2,

Section 2.7.

These secure coding practices were used to design a checklist. This checklist was used to address Secondary Objective 2 of this research, which was *to determine the adherence of third year software development capstone projects to the identified secure coding practices*. In order to determine this adherence, a code review was conducted, using the checklist. The code review was conducted on 2015 third year capstone projects developed in the .NET environment, since the students were taught specifically in the .NET environment.

The researcher only focussed on the secure coding practices for the data access layer, since this layer is responsible for interacting with the database, where all the sensitive data is stored. Furthermore, it is important to pay attention to the data access layer since the manipulation of the data and the logic of the application is in this layer. The vulnerability list in Chapter 2, Section 2.7 shows that SQL injection is top on the list. Most of the vulnerabilities that result in SQL injection stem from poorly written code in the data access layer. The following section describes the design of the code review instrument.

5.3 Code Review Instrument Design

As already mentioned, the instrument that was used to conduct the code review was a checklist, using the identified secure coding practices. All these secure coding practices were converted to questions as recommended in the theory relating to checklists in Chapter 3, Section 3.3.2.

The questions asked in the code review are shown in Table 5.1. Table 5.1 indicates each Secure Coding Practice (SP) and the related question asked in the code review. There is a one-to-one match between the secure coding practices identified in Chapter 2, Section 2.7 and Table 2.3.

SP	Questions
SP1	Do they make use of parameterised SQL commands for all data access? <i>Yes / No (Number of Instances)</i>
SP2	Do they make use of concatenated strings in the queries? <i>Yes (Number of Instances) / No</i>
SP3	Are all input fields validated? <i>Validated / Not Properly Validated / Not applicable</i>
SP4	Do they make use of the Principle of Least Privilege when setting up their databases? <i>Yes / No</i>
SP5	Do they use integrated authentication or do they use SQL authentication? <i>Integrated Authentication / SQL Authentication</i>
SP6	Do they use stored procedures for their queries? <i>Yes / No / Inconsistent Use of Stored Procedures and Queries</i>
SP7	Do they encrypt their connection strings? <i>Yes / No</i>
SP8	Does the connection string only appear once in the web.config file? <i>Yes / No</i>
SP9	Is all the sensitive data being encrypted using the OWASP recommended methods? <i>Encrypted Using Acceptable Method / No Encryption / Encrypted Not using Acceptable Method</i>

Table 5.1: Code Review Checklist

Since some of the questions in the checklist would not be relevant to some forms in the capstone projects, the checklists also provided answers depending on the web form that was being reviewed. For example, when checking for validation, a form can either be *Validated*, *Not Properly Validated*, or *Not Applicable*. The questions asked in the checklist were closed-ended questions. For SP1 and SP2, the researcher recorded the number of instances where the secure coding practice was applied, since some of the students may not be consistent in adhering to these secure coding practices.

The data from the code review were recorded in a Windows application that the researcher developed and used to analyse the data further. The Windows application stored the details of each project and their related forms, as illustrated in the database in Figure 5.2.

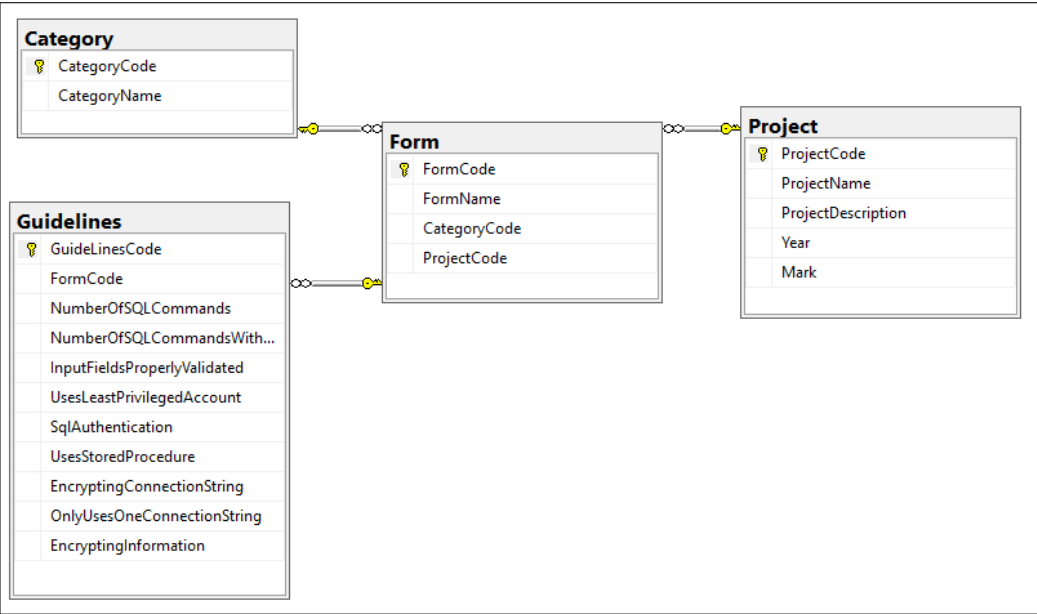


Figure 5.2: Database for Code Review

The researcher categorised the forms, for example, Register, Login, Add, Update, or Search, so that the researcher would have an idea of what the form does and what to look for when conducting the code review. The Category entity in Figure 5.2 was used to store these categories for the forms. The Guidelines entity was used to store the results of the form after it had been reviewed. The Form and Project entity stored the details of the forms for each project. The following section describes how the code review was conducted, as part of the behavioural analysis.

5.4 Conducting the Behavioural Analysis

As mentioned in Section 5.2, the code review was conducted on 2015 third year capstone projects. The researcher selected 15 capstone projects that were developed in the .NET environment. The code review was conducted manually by the researcher. During the code review process, for each chosen project, all the web forms that access the database and data access layer classes were reviewed for adherence to the nine checklist questions as shown in Table 5.1.

Each form in the code review was reviewed depending on the category it belonged to, as shown in Figure 5.2. For example, if the form being reviewed

was Register, then the researcher checked for encryption of passwords, identity number or account numbers depending on the function of the project. The following sections discuss how the code review was carried out for each secure coding practice.

5.4.1 Parameterised SQL Commands

In the case of parameterised SQL commands, as shown in Figure 5.3, the researcher looked at whether the students used parameterised SQL commands or whether they used concatenated SQL strings when writing their queries. The researcher counted the number of instances where the form used parameterised SQL commands.

```
using (SqlCommand cmd = new SqlCommand("InsertStudent", conn))
{
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.AddWithValue("@StudentNo", StudentId.Text);
    cmd.Parameters.AddWithValue("@FirstName", TextFirstName.Text);
    cmd.Parameters.AddWithValue("@LastName", TextLastName.Text);
    if (RadioButtonMale.Checked == true)
    {
        title = "Male";
        cmd.Parameters.AddWithValue("@gender", title);
    }
    else if (RadioButtonFemale.Checked == true)
    {
        title = "Female";
        cmd.Parameters.AddWithValue("@gender", title);
    }
    cmd.Parameters.AddWithValue("@parentID", DDParentID.SelectedItem.Value);
    s = DDownYears.SelectedValue.ToString() + "/" + DropDownMonth.SelectedValue.ToString() + "/" + DDownDay.SelectedValue.ToString();
    DateTime date = Convert.ToDateTime(s);
    cmd.Parameters.AddWithValue("@DOB", date);
    cmd.Parameters.AddWithValue("classCode", DropDownListClass.SelectedItem.Value);
    cmd.Parameters.AddWithValue("gradeCode", DropDownListGrade.SelectedItem.Value);
    cmd.ExecuteNonQuery();
    LabelError.Visible = true;
    LabelError.Text = "Added Successfully";
    Response.Redirect("AddStudents.aspx");
    conn.Close();
}
```

Figure 5.3: Parameterised SQL Command Example

5.4.2 Concatenated SQL Strings

In some projects, the students used concatenated SQL strings for parameters as shown in Figure 5.4. In that case, the researcher also counted the number of instances where concatenated SQL strings had been used. This method is unacceptable in terms of OWASP recommendations, as this method would cause an application to be vulnerable to SQL injection attacks.

```
try
{
    SqlConnection conn = new SqlConnection(connect);
    conn.Open();

    string sql = "UPDATE Student SET firstName='" + TextFirstName.Text + "',lastName='" + TextLastName.Text + "',gender=@gender WHERE studentNo=@studentNo";
    SqlCommand cmd = new SqlCommand(sql, conn);
    cmd.Parameters.AddWithValue("@studentNo", StudentId.Text);
    cmd.Parameters.AddWithValue("@firstName", TextFirstName.Text);
    if (RadioButtonMale.Checked == true)
    {
        title = "Male";
        cmd.Parameters.AddWithValue("@gender", title);
    }
    else if (RadioButtonFemale.Checked == true)
    {
        title = "Female";
        cmd.Parameters.AddWithValue("@gender", title);
    }
    cmd.ExecuteNonQuery();
    LabelError.Text = "Update Successfully";
    StudentId.Text = "";
    TextFirstName.Text = "";
    TextLastName.Text = "";
}
```

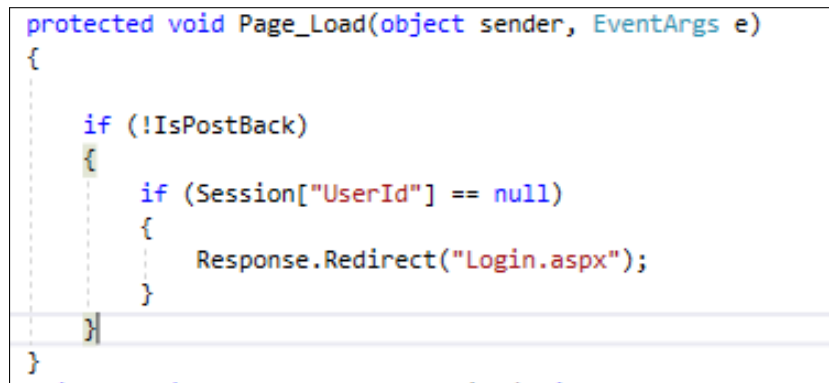
Figure 5.4: Concatenated SQL String Example

5.4.3 Input Validation

When validating user inputs, some projects only validated for empty fields, and did not validate for specific characters in some input fields. When reviewing a form for validation, the researcher would look at the category of the form to determine whether special characters should be allowed or whether they should be whitelisted, depending on the functionality of the form.

5.4.4 Principle of Least Privilege

In the Principle of Least Privilege, the researcher checked whether the page should be viewed by everyone, or whether certain users had rights to certain functionalities. The researcher checked for whether the students catered for users who would type the URL of the page without being thrown out since they would not have access rights.



```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        if (Session["UserId"] == null)
        {
            Response.Redirect("Login.aspx");
        }
    }
}
```

Figure 5.5: Principle of Least Privilege Example

Some of the projects evaluated applied the technique shown in Figure 5.5, which checks if a particular user is logged into the system and directs the user to the Login page if the session is not active.

5.4.5 Authentication

When students publish their projects to the server, they are provided with a server name, which they use to publish their web applications. The requirements for the capstone project are such that they should use SQL authentication, not Windows Authentication. However, the secure coding practices recommend that they should use Windows Authentication. Some capstone projects used Windows authentication, since they might not have published their web application to the university server, which means that they deviated from the capstone project requirements.

5.4.6 Stored Procedures

If the programmer has not made use of stored procedures, it potentially makes it easy for an attacker to see the data and tables in the database should they gain access to the application code. Some projects made use of concatenated strings, as shown in Figure 5.4 instead of stored procedures. However, some used stored procedures, as shown in Figure 5.6.

```
public DataTable Load()
{
    DataTable dt = new DataTable();
    SqlConnection conn = new SqlConnection(con);
    SqlDataAdapter data = new SqlDataAdapter("GetAllStudentDetails", conn);

    data.SelectCommand.CommandType = CommandType.StoredProcedure;
    DataSet ds = new DataSet();
    try
    {
        data.Fill(ds, "Students");
        return ds.Tables["Students"];
    }
    catch
    {
        throw;
    }
    finally
    {
        ds.Dispose();
        data.Dispose();
        conn.Close();
        conn.Dispose();
    }
}
```

Figure 5.6: Stored Procedure Example

5.4.7 Encrypting Connection Strings

The researcher checked whether the connection strings were encrypted in the projects. None of the projects had a connection string encrypted, as this was a requirement of the capstone project for evaluation purposes.

5.4.8 Connection Strings

Figure 5.7 shows a connection string in a config file. This connection string would have to be used as reference only in a helper file in the data access layer. In some projects, students incorrectly show the connection string in all the forms that connect to the database.


```
<connectionStrings>
  <add name="CNS" connectionString="server=sict-sql;
    database=CNS;Integrated Security=false; user id=cns;
    password=password;" providerName="System.Data.SqlClient"/>
</connectionStrings>
```

Figure 5.7: Connection String Example

5.4.9 Encryption using Acceptable Methods

When checking for encryption, the researcher checked whether passwords or identity numbers were being encrypted when stored to the database. The secure coding practices suggest that students should use certain encryption algorithms acceptable for .NET web applications. Some students did not encrypt data at all where they should have, while others used the acceptable secure coding practices, such as hashing algorithms and Data Protection Application Programming Interface (DPAPI). The following section presents the results of the code review.

5.5 Results of Code Review

After the code review was complete, the researcher calculated the extent to which the projects adhered to the secure coding practices as percentages. The 15 projects were labelled P1 to P15 respectively.

The percentage indicated that the level of adherence was calculated as the ratio between the number of web forms accessing the database that should have applied the secure coding practices and those that actually implemented them.

The formula is as follows: *Percentage = Number of Instances Adhered to / Overall Instances that should be adhered to * 100.*

	Form Number	Form Name	Number Of SQL Commands (Good)	Number Of SQL Commands With Conc Strings (Bad)	Input Fields Properly Validated	Uses Least Privileged Account	Sql Authentication	Uses Stored Procedure	Encrypting Connection String	Only Uses One Connection String	Encryption	Category Name
▶	32	AdmissionStat...	2	2	Input NOT Pro...	No	Integrated Au...	Uses Stored Pr...	No	Connection St...	No Encryption	View/List
	33	Application.as...	2	6	Input NOT Pro...	No	Integrated Au...	NOT Using Sto...	No	Connection St...	No Encryption	Create/Add
	34	ChangePassw...	1	0	Input NOT Pro...	No	Integrated Au...	Uses Stored Pr...	No	Connection St...	No Encryption	Edit/Update
	35	Contact_Us.as...	1	0	Input NOT Vall...	Yes	Integrated Au...	NOT Using Sto...	No	Connection St...	No Encryption	View/List
	36	Forgot_Passw...	1	1	Input NOT Vall...	No	Integrated Au...	Uses Stored Pr...	No	Connection St...	No Encryption	Edit/Update
	37	LearnerHome...	1	1	Input NOT Vall...	No	Integrated Au...	NOT Using Sto...	No	Connection St...	No Encryption	View/List
	38	Login.aspx.cs	0	2	Input NOT Pro...	No	Integrated Au...	NOT Using Sto...	No	Connection St...	No Encryption	Login
	39	ParentHome.a...	1	1	Input NOT Vall...	No	Integrated Au...	NOT Using Sto...	No	Connection St...	No Encryption	View/List
	40	Profiles.aspx.cs	0	6	Input NOT Vall...	No	Integrated Au...	NOT Using Sto...	No	Connection St...	No Encryption	View/List
	41	Register.aspx.cs	1	1	Input NOT Pro...	No	Integrated Au...	Uses Stored Pr...	No	Connection St...	No Encryption	Register
	42	TeacherHome...	0	1	Input NOT Vall...	No	Integrated Au...	NOT Using Sto...	No	Connection St...	No Encryption	View/List
*												
<div> <div><</div> <div>72.73%</div> <div>27.27%</div> <div>0%</div> <div>9.09%</div> <div>100%</div> <div>36.36%</div> <div>0%</div> <div>0%</div> <div>81.82%</div> <div>></div> </div>												

Figure 5.8: Captured Data Example (P14)

Figure 5.8 shows the results for the capstone project, labelled P14. This project had 11 forms accessing the database numbered 32 to 42. In order to calculate the adherence percentage, the researcher checked the number of violations by assigning colours to cells depending on the outcome in the form after the code review. Adherence to secure coding practices is indicated in green, while non-adherence to secure coding practices is indicated in orange. The calculations for the percentages are in Figure 5.8 where *Number of Instances actually adhered to (green)/Overall Instances that should be adhered to (total)*100*. For example, to calculate SP6 for P14, using the percentage formula: $\text{percentage} = 4/11 * 100 = 36\%$. The percentages were calculated and captured in a Microsoft Excel spreadsheet, as shown in Table 5.2. The results are shown according to the Secure Coding Practices (SPs) as shown in Table 5.1. The results from the code review are presented in Table 5.2.

ProjNo	SP1	SP2	SP3	SP4	SP5	SP6	SP7	SP8	SP9
P1	100%	100%	100%	100%	0%	85%	0%	100%	28%
P2	84%	100%	100%	100%	0%	100%	0%	65%	15%
P3	100%	100%	97%	100%	0%	27%	0%	42%	15%
P4	100%	100%	100%	100%	0%	0%	0%	100%	12%
P5	50%	50%	6%	100%	0%	50%	0%	0%	40%
P6	90%	80%	70%	81%	0%	13%	0%	81%	60%
P7	93%	93%	100%	92%	0%	67%	0%	57%	40%
P8	100%	100%	100%	0%	0%	27%	0%	100%	0%
P9	100%	100%	100%	55%	0%	45%	0%	100%	12%
P10	100%	100%	100%	5%	0%	40%	0%	100%	5%
P11	100%	100%	85%	0%	0%	57%	0%	100%	0%
P12	85%	85%	100%	0%	100%	33%	0%	87%	57%
P13	73%	70%	40%	0%	100%	13%	0%	53%	26%
P14	72%	27%	0%	9%	100%	36%	0%	0%	82%
P15	40%	57%	14%	92%	100%	0%	0%	35%	38%
Avg.	86%	84%	77%	60%	27%	38%	0%	68%	31%

Table 5.2: Adherence to Secure Coding Practices

Table 5.2 shows the results from the code review. For clarity, the percentages for all the forms per secure coding practice have been consolidated for each project. Table 5.2 also shows an average of all projects across each secure coding practice, which was calculated as **SP Average = sum of all SP percentages (per SP)/number of projects**. These indicate that SP1 and SP2 have the highest adherence, and that SP5 and SP7 have the lowest adherence.

5.6 Behavioural Analysis Discussion

The results presented in Table 5.2 indicate that there is some lack of adherence to the secure coding practices. This is indicated by the average adherence where only 4 of the 9 secure coding practices had an average of 70% or above. For web applications to be secure, as a minimum, these secure coding practices should be consistently adhered to.

For example, on SP1 (86%), which deals with parameterised SQL commands,

and SP2 (84%), which deals with the use of concatenated strings in SQL queries, the level of adherence is high in these secure coding practices. Even though some of the projects did not consistently adhere to these secure coding practices, it can be seen that implementing the secure coding practices is possible as most project groups had 100% adherence to these secure coding practices.

The other two secure coding practices where adherence seems to be high are SP6 (77%), use of stored procedures, and SP8 (71%), storing connection string in web.config file. However, for both these secure coding practices, there is an inconsistent adherence.

Another example can be seen in SP4 (60%), the Principle of Least Privilege, which refers to giving rights to certain individuals within an application. There was, to a large extent, a lack of adherence because some of the web pages should not have been visible to unauthorised users, but just by typing the full URL to the web page, access was granted. This violates the confidentiality, availability and integrity of information.

Considering the level of adherence to secure coding practices, SP9, which checks for the correct ways of encryption, it can be seen that students' adherence was poor in this regard. Only 3 of the 15 projects were above 50% adherence, showing that it is possible to adhere to this secure coding practice although in some instances students seem to have chosen to ignore them. Some capstone projects, such as P11, did not even encrypt any sensitive information.

For SP3, validating user input, only 5 of the 15 capstone projects were above 50% in adherence to this secure coding practice. P2 has proved that this secure coding practice can be fully implemented, and has an adherence of 100%.

For SP7, none of the students encrypted their connection strings. However, this is a requirement of the capstone project for evaluation purposes as the connection string must be visible in plain text. Regardless, it is necessary for students to know that, in reality, connection strings should be encrypted. Lastly, SP5 (27%), for the choice of authentication, the requirements of the capstone project are to use SQL authentication over the OWASP preferred Windows Authentication. Surprisingly, 4 projects deviated from the project requirements and used Windows Authentication.

The inconsistent adherence to secure coding practices, as indicated in Table 5.2 shows that there is generally a lack of behavioural compliance amongst the third year capstone projects. It is only for SP7 and SP9 where there is no project with 100% adherence. However, with SP7 it is an exception, since it is a capstone project requirement. Otherwise, all the other secure coding practices have at least 1 project with 100% adherence. This indicates that implementing these secure coding practices is possible; yet it is evident that some capstone project students chose not to comply with these secure coding practices.

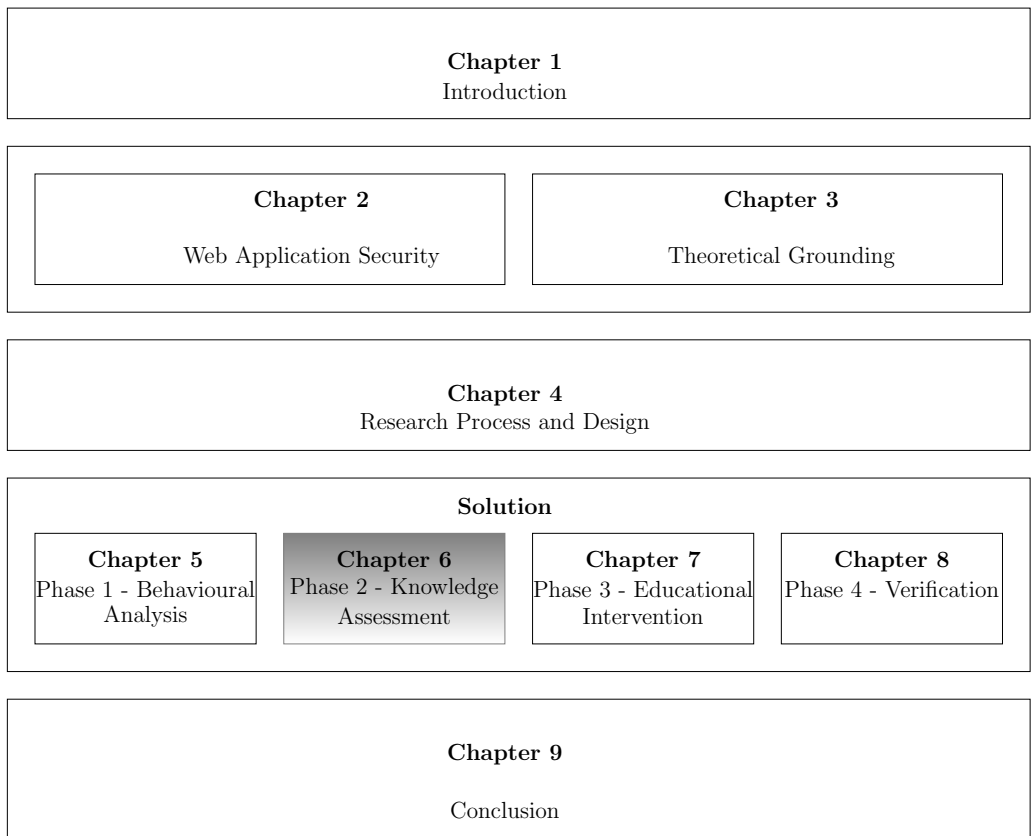
5.7 Conclusion

The aim of Phase 2, the Behavioural Analysis, was to determine whether students developing their third year capstone projects adhere to secure coding practices as recommended by OWASP. *The results from the code review indicate that students developing their capstone projects do not consistently adhere to secure coding practices when developing their projects.* This inconsistency indicates a general lack of behavioural compliance from the students. In order to determine whether students developing their capstone projects have the requisite knowledge regarding secure coding practices, the next chapter focusses on the assessment of students' knowledge with regard to secure coding practices.

Chapter 6

Knowledge Assessment

The purpose of this chapter is to provide an overview of the research process followed in Phase 2 of this study. This phase seeks to determine whether software development students have the requisite knowledge regarding secure coding practices. This was achieved through the use of a questionnaire.



6.1 Introduction

In the previous chapter, the behaviour of 2015 students programmers was analysed through their capstone projects to establish whether or not they adhered to the identified secure coding practices. This behavioural analysis indicated that students in general did not adhere to these practices. This general lack of behavioural compliance may result from the unwillingness of students to adhere to such practices or to their lack of requisite knowledge. The purpose of Phase 2 in the research process was to determine whether the students engaged in third year capstone projects at the Nelson Mandela University had the requisite knowledge regarding the identified secure coding practices.

This chapter discusses the process flow for Phase 2 of this research, followed by the knowledge assessment instrument design in Section 6.3. Section 6.4 addresses the conducting of the knowledge assessment, while Section 6.5 provides the results of the knowledge assessment, and Section 6.6 provides the discussion of the results. Section 6.7 is the conclusion of this chapter.

6.2 Knowledge Assessment Process Flow

This section provides the process flow for the knowledge assessment phase as shown in Figure 6.1.

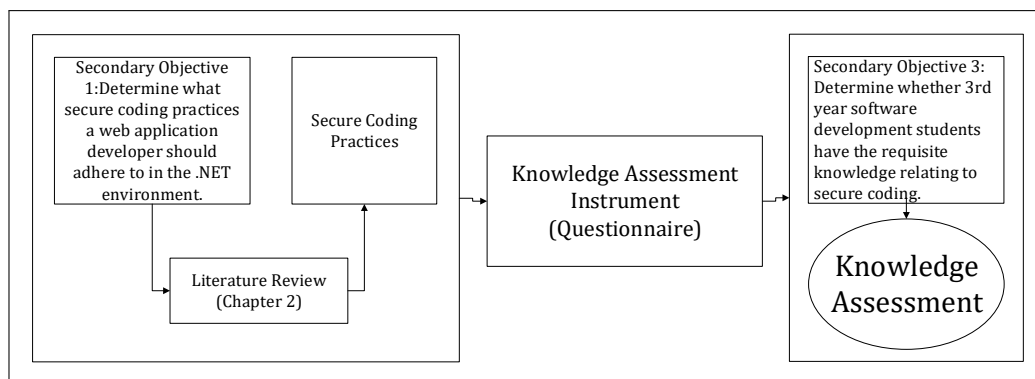


Figure 6.1: Phase 2 - Knowledge Assessment Process Flow

Figure 6.1 provides an overview of the steps followed in the knowledge assessment phase of this research. This was achieved through a questionnaire

that was distributed to 2017 capstone project students. This questionnaire served as a pre-test to determine existing knowledge regarding secure coding practices and as such was also used in the design of the questionnaire.

6.3 Knowledge Assessment Instrument Design

In order to formulate the knowledge assessment questionnaire, the identified secure coding practices were analysed and questions were formulated to determine students' knowledge regarding secure coding practices.

The questionnaire consisted of 15 multiple choice questions, and students were required to select the answer that best answered the question. Some questions had more than one correct answer. Some of the questions were based on specific scenarios and some secure coding practices were addressed by more than one question as shown in Table 6.1.

SP number	OWASP Secure Coding Practices	Questions (Q)
SP1	Use Parameterised SQL commands for all data access, without exception.	Q2, Q4
SP2	Do not use SQL command with string made up of a concatenated SQL string.	Q3
SP3	Properly validate input fields.	Q1
SP4	Apply the Principle of Least Privilege when setting up the database of your choice.	Q13, Q14, Q15
SP5	When using SQL Server, prefer integrated authentication over SQL authentication.	Q7, Q8, Q9
SP6	Using stored procedures is the most effective way to counter the SQL injection vulnerability.	Q5, Q6
SP7	Encrypt connection strings.	Q10
SP8	Connection strings should be based in a configuration file.	Q11
SP9	Encrypt sensitive data using acceptable encryption algorithms.	Q12

Table 6.1: OWASP Secure Coding Practices Related Questions

Questions 1 and 2 are based on Figure 6.2. Figure 6.2 is referred to as Figure 1 in the original questionnaire, and in the questions it will be referred to as Figure 1.

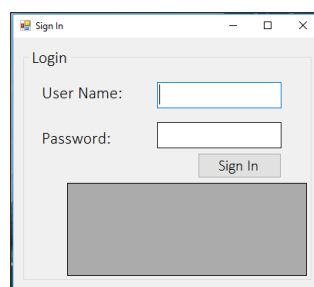


Figure 6.2: Sign In Scenario (Figure 1)

Question 1: Figure 6.3 illustrates Question 1 in the questionnaire. Question 1 relates to SP3, input validation, and its purpose is to assess students'

knowledge regarding implications caused by input fields that are not properly validated. This question has one correct answer which had 1 marks allocation. The correct answer for this question is Option D.

Question 1

Given User Input: Name' OR 1=1; DROP TABLE USERS;--

What would happen if the user enters this input in the 'User Name' input field regarded that the query *Select * FROM Users Where UserID = txtUserID.Text AND Password = txtPassword.Text?*

- A) User Would be required to enter a password first.
- B) Nothing will happen.
- C) The system will log the user in.
- D) User will be logged in and the **User** table will be deleted.
- E) None of these answers are correct.

Figure 6.3: Knowledge Assessment - Question 1

Question 2: Figure 6.4 illustrates Question 2 in the questionnaire. This question attempts to check whether students know which correct query to use in a query with parameters. This question relates to SP1 which deals with the use of Parameterised SQL commands. Options C and B in the question can both work; however, Option B is more secure than Option C. This question had 1 mark allocation.

Question 2 Which of the following SQL statements would be best to use in the Form shown in Figure 1 in order to avoid attacks?

- A) `SELECT * FROM USERS WHERE UserID = txtUserId.Text and Password = txtPassword.Text;`
- B) `SELECT UserID, Password FROM Users WHERE UserID = @UserID AND Password = @Password;`
- C) `SELECT UserID, Password FROM Users;`
- D) All of the above.

Figure 6.4: Knowledge Assessment - Question 2

Question 3: Figure 6.5 illustrates Question 3 in the questionnaire. This question is similar to Question 2, but Question 3 tries to point out why Option C in Question 2 is not the best to use with parameters, since it is not secure. In Option A, the SQL statement made use of concatenated SQL strings, which are vulnerable to SQL injections. Therefore, Option A is the correct answer for this question. This question has 1 mark allocated to it and relates to SP2, which is the use of concatenated SQL strings.

Question 3 Which of the following SQL statements would be the most vulnerable to attacks?
A) `SELECT * FROM USERS WHERE UserID = txtUserId.Text and Password = txtPassword.Text;`
B) `SELECT UserID, Password FROM Users WHERE UserID = @UserID AND Password = @Password;`
C) `SELECT UserID, Password FROM Users;`
D) All of the above.

Figure 6.5: Knowledge Assessment - Question 3

Question 4: Figure 6.6 illustrates Question 4 in the questionnaire. This question relates to SP1, which relates to the use of parameterised SQL commands. This question sets out to assess students' knowledge about the secure coding practice using parameterised SQL commands. Option A is the correct answer for this question, and 1 mark is allocated for this question.

Question 4 Parameterized SQL commands are a preferred way of coding in data access?
A) Yes
B) No

Figure 6.6: Knowledge Assessment - Question 4

Question 5: Figure 6.7 illustrates Question 5 in the questionnaire. This question relates to SP 6, stored procedures, and has more than one correct answer. This question assessed students' knowledge regarding secure ways of dealing with queries. Option A, B, and C can be used successfully, but Option C is not a secure way of dealing with queries. Therefore, Options A and B are correct answers for this question. This question has 2 marks allocated to it (1 mark for each correct option selected).

Question 5 When writing queries for a form, which of the following are acceptable ways for dealing which placeholders in the queries?
A) Parameterised SQL statements
B) Concatenated SQL strings
C) Stored Procedures.
D) Use Try-Catch and Concatenated SQL strings.

Figure 6.7: Knowledge Assessment - Question 5

Question 6: Figure 6.8 illustrates Question 6 in the questionnaire. This question also relates to SP6, and has one correct answer. The aim of this

question was to check whether students know that stored procedures are a recommended method for coding in the data access layer. The correct option for this question is Option A.

Question 6 Which of the following methods is most appropriate way for writing SQL statements?
A) Parameterized SQL statements.
B) Concatenated SQL strings.
C) Use Try-Catch and concatenated SQL strings.
D) Stored Procedures.

Figure 6.8: Knowledge Assessment - Question 6

Question 7: Figure 6.9 illustrates Question 7 in the questionnaire. Questions 7, 8 and 9 relate to SP5, which focusses on the use of authentication. This question (Question 7) seeks to determine whether students know the various ways of database authentication methods in SQL Server databases. This question has two correct options, Options B and D, and thus 2 marks in total (1 mark for each selected option).

Question 7 Which of the following are valid approaches to authentication when connecting to a SQL Server database?
A) Integrated Authentication (Windows Authentication).
B) Administrator and User login.
D) Mixed Authentication (SQL Authentication)
E) None of the above.

Figure 6.9: Knowledge Assessment - Question 7

Question 8: Figure 6.10 relates to Question 8 in the questionnaire. This question aims to find out whether students know which authentication method is secure and preferred by the secure coding practices when using an SQL Server database. Only 1 answer is correct for this question, which is Option E.

Question 8 Which is the most preferred method for authentication?
A) Integrated Authentication (Windows Authentication).
B) Administrator and User login.
D) Mixed Authentication (SQL Authentication)
E) None of the above.

Figure 6.10: Knowledge Assessment - Question 8

Question 9: Figure 6.11 illustrates Question 9 in the questionnaire. For Question 9, students were assessed as to whether they know the security implications for the authentication method chosen in Question 8. This question only had 1 correct option, and 1 mark allocated to it. Option A is the correct answer for this question.

Question 9 Why the authentication method selected in Question 8 preferred?
A) Requires no password because Windows takes care of security issues.
B) Secure with your own user name and password.
C) Easiest method of all the methods.
D) Administrator and User Login is simplified.
E) Secure with an encrypted username and password.

Figure 6.11: Knowledge Assessment - Question 9

Question 10: Figure 6.12 illustrates Question 10 in the questionnaire. This question seeks to identify students' understanding of how encryption algorithms work. Option B is the correct answer for this question.

Question 10 When sensitive data has been encrypted, only the authorized parties can analyze the data.
A) True.
B) False.

Figure 6.12: Knowledge Assessment - Question 10

Question 11: Figure 6.13 illustrates Question 11 in the questionnaire. Question 11 relates to SP7, encryption of connection strings. This question aims to determine whether students know what should be encrypted. The focus of this question was to see if they would select Option A (Connection Strings), showing that they know that it should be encrypted, and select

other options. The correct options for this question are Option A, B, and C and it has 3 marks allocated to it (1 mark for each correct option).

Question 11 Which of the following data should be encrypted?
A) Connection strings
B) Sensitive parts of the web.config file.
C) Password.
D) Sensitive information such as ID number only.
E) Application Code snippets
F) SQL statements

Figure 6.13: Knowledge Assessment - Question 11

Question 12 Figure 6.14 illustrates Question 12 in the questionnaire. Since the secure coding practices provide recommendations on which acceptable encryption algorithms should be used, this question tries to see if the student knows those recommendations, as suggested in SP9. This question has 3 marks allocated to it, with B, C, and D as correct answers (1 mark for each correct option).

Question 12 Which are the recommended ways of encryption?
A) Windows Data Protection (DPAPI).
B) Using a hashing algorithm.
C) ASP.NET RegIIS.
D) Writing your own encryption using an algorithm with random numbers and alphabets.
E) 64-bit algorithm for the right system.

Figure 6.14: Knowledge Assessment - Question 12

Question 13: Figure 6.15 illustrates Question 13 in the questionnaire. Questions 13, 14 and 15 relate to SP4, which is the use of the Principle of Least Privilege. This question seeks to determine whether students know whether the Principle of Least Privilege applies. The correct answer for this question is Option C, and has 1 mark allocated to it.

Question 13 Where does the Principles of Least Privilege apply?
A) 64-bit algorithm for the right system.
B) Admin User rights of the system.
C) In the code behind for assigning rights.
D) Login.

Figure 6.15: Knowledge Assessment - Question 13

Question 14: Figure 6.16 illustrates Question 14 in the questionnaire. This question tries determine whether students know why they use the Principle of Least Privilege. This question has 1 mark allocated to it, and the correct answer for this is Option A.

Question 14 Why is the Principle of Least Privilege seen as important in application development?
A) To give rights to certain people depending on the positions in an organization.
B) For users of the system to be guided on every action they take on the system.
C) For security reasons.
D) For users to be able to complete their tasks.

Figure 6.16: Knowledge Assessment - Question 14

Question 15: Figure 6.17 illustrates Question 15 in the questionnaire. This question aimed to determine whether students know how the Principle of Least Privilege can be used in a scenario. This question had one correct option, Option C, and 1 mark allocated it.

Question 15 How could the principles of Least Privilege be applied to employees in an organization?
A) Having a person look at them whilst they sign in the system.
B) Create a password for users depending on a department.
C) Giving employees rights according to the department they belong to.
D) Give rights to the manager of a department and allow the manager to deal with logging of users.

Figure 6.17: Knowledge Assessment - Question 15

The questionnaire was prepared and distributed to the 2017 capstone project students to answer. The conducting of the questionnaire is described in the following section.

6.4 Conducting the Knowledge Assessment

Conducting the knowledge assessment on 2017 project students was deemed acceptable as the capstone project is an integration of modules which have already been covered in the curriculum. The students who were engaged in the capstone project in 2015 would have had the same programming knowledge as those students engaging in the capstone project in 2017.

The questionnaires were administered during a 2017 capstone project lecture where 86 students were present. The lecturer for the capstone project was

not present. The researcher briefly provided students with the context of the study before distributing the paper-based questionnaire for completion. The students were given a question paper, and an answer sheet. The students were given 30 minutes to complete the questionnaire. The answer sheet was marked by an automated computer marking system. The answer sheet contained Options A to F that the student could select by shading the correct option. The students had to use a dark shading for their choices so that the computer marker could recognise their choice. The answer sheet is provide as Appendix B. The questionnaire together with the answer sheets were collected and marked, and the results collected and analysed as discussed in Sections 6.5 and 6.6 respectively.

6.5 Results of Knowledge Assessment

This section provides the results for the knowledge assessment. These results are illustrated using graphs and tables. Figure 6.18 presents the results from the questionnaire completed by 2017 capstone project students to determine their knowledge regarding secure coding practices.

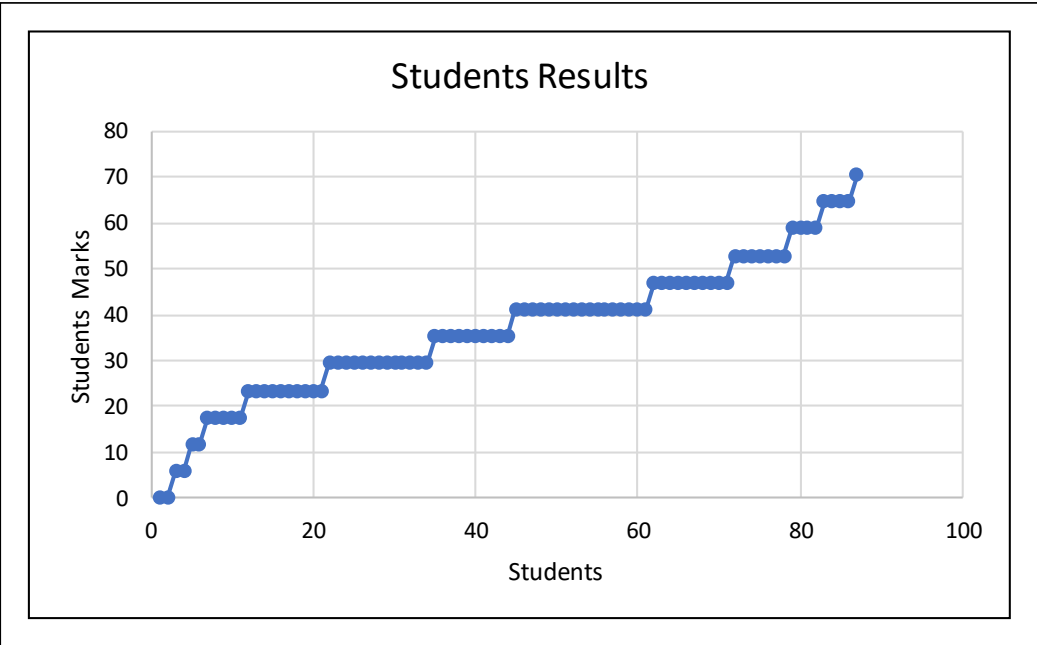


Figure 6.18: Student Marks

Figure 6.18 shows the graph of student marks for the results of the knowl-

edge assessment. Of the 86 students who took the questionnaire, 88% received a mark of less than 50%, and only 12% of them scored between 60% and 70%. This shows that there was a lack of secure coding knowledge amongst the third year capstone project students.

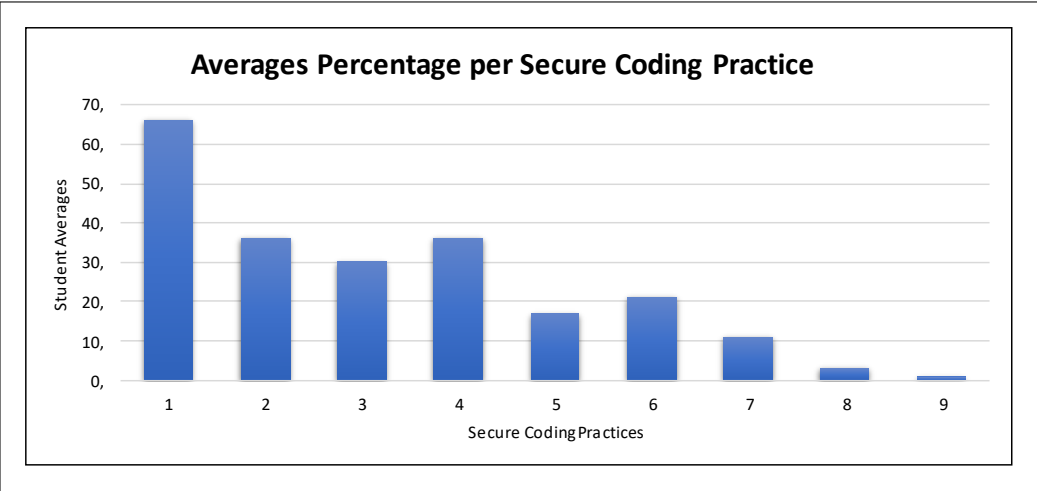


Figure 6.19: Average percentage per Secure Coding Practice

Figure 6.19 illustrates student averages per secure coding practice. Of the 9 secure coding practices, only SP1 and SP4 had an average above 50%.

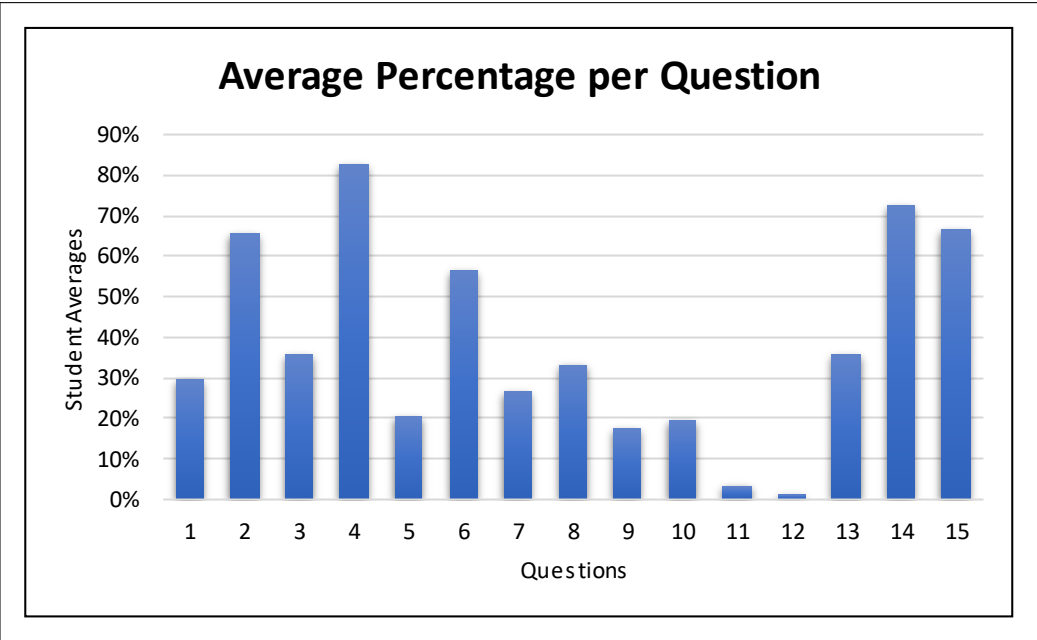


Figure 6.20: Average Percentage per Question

Figure 6.20 shows the average percentage obtained by students in each of the questions asked in the knowledge assessment. The results for each question also shows a lack of knowledge regarding secure coding practices, as these results show that only 5 of the 15 questions had an average above 50%. Questions with averages above 50% are Questions 2 and 4, which relate to SP1, and Question 6, which relates to SP6. Lastly, Questions 14 and 15, which relate to SP4. For the rest of the questions, student averages were lower than 50%. This shows that few students were familiar with these secure coding practices. A detailed discussion of the knowledge assessment results follow.

6.6 Knowledge Assessment Discussion

As can be seen in Figures 6.18, 6.19, and 6.20, students generally lacked knowledge regarding secure coding practices.

In Figure 6.18, 70 students out of the 86 (81%) students received lower than a 50% mark for their knowledge assessment.

In Figure 6.19, only in 2 (SP1 and SP4) secure coding practices did students obtain an average of more than 50%. For the other secure coding practices, students performed poorly.

In terms of results per question, as shown in Figure 6.20, the students performed poorly (less than 40%) in 10 out of the 15 questions. Especially in Questions 11 and 12, which relate to SP8 and SP9 respectively, where the averages for students were lower than 10%.

Therefore, many of these students lack the requisite knowledge regarding secure coding practices. It is therefore important that students acquire the knowledge of these secure coding practices when developing web applications.

Phases	SP1	SP2	SP3	SP4	SP5	SP6	SP7	SP8	SP9
Phase 1	86%	84%	77%	60%	27%	38%	0%	68%	31%
Phase 2	74%	36%	30%	58%	26%	39%	20%	3%	1%

Table 6.2: Results from Phase 1 and 2 Compared

If one considers the results of both Phases 1 (Behavioural Analysis) and 2 (Knowledge Assessment) collective shown in Table 6.2, the results suggest

that only 1 of the secure coding practices (SP1), students both behavioural and knowledge aspects. In most cases, they performed poorly. For example, in *SP1* and *SP4*, the students seem to have behaved well and to some extent had the requisite knowledge regarding secure coding practices. In *SP7* and *SP9*, students lacks both the behavioural and knowledge aspects. In many cases, the level of behavioural adherence is higher than the actual level of the underlying security knowledge. This could be an indication that students are being taught the appropriate ways of using the code without linking that to the underlying security context. This poses a long-term risk because if the programmers do not understand why they are doing something, they still disregard it in future implementations.

6.7 Conclusion

As can be seen in the in the two phases of the research, students in general do not consistently adhere to secure coding practices, and to some extent do not have the requisite knowledge regarding secure coding practices. Therefore, at this stage it was determined that the research intervention for this study should consist of two parts. The first part would be an educational intervention to ensure that students have the knowledge, and the second part would be a behavioural intervention to try to ensure compliance. The next chapter will describe the educational intervention.

Chapter 7

Educational Intervention

This chapter provides an overview of the educational intervention for this research, which consists of two components, and sets out to address Phase 3 of this research. The first component is the knowledge component and the second component is the behavioural component.

<div>Chapter 1 Introduction</div>			
<div>Chapter 2 Web Application Security</div>		<div>Chapter 3 Theoretical Grounding</div>	
<div>Chapter 4 Research Process and Design</div>			
<div>Solution</div> <div><div><div>Chapter 5 Phase 1 - Behavioural Analysis</div></div><div><div>Chapter 6 Phase 2 - Knowledge Assessment</div></div><div><div>Chapter 7 Phase 3 - Educational Intervention</div></div><div><div>Chapter 8 Phase 4 - Verification</div></div></div>			
<div>Chapter 9 Conclusion</div>			

7.1 Introduction

As mentioned in Chapter 5, Section 5.6, students, in general did not adhere to secure coding practices in their capstone software development projects. In addition, the researcher found that students generally also lack the requisite knowledge regarding secure coding practices, as discussed in Chapter 6, Section 6.5. Therefore, the need for an educational intervention was established. The educational intervention contained elements that addressed both the knowledge and the behaviour of students regarding secure coding practices.

Owing to the lack of knowledge on the part of the students, the researcher realised the need to create a knowledge component that could assist students in acquiring the requisite knowledge regarding secure coding practices. The need to address behavioural compliance was also realised since it is known that having knowledge does not necessarily ensure that people would behave accordingly (Vroom & Von Solms, 2004).

This chapter is divided into two main parts. The first part describes the knowledge component in Section 7.4, and the second part describes the behavioural compliance monitoring component in Section 7.5. The following section describes the process flow of the educational intervention.

7.2 Educational Intervention Process Flow

Figure 7.1 illustrates the steps followed to design and implement the educational intervention for this research.

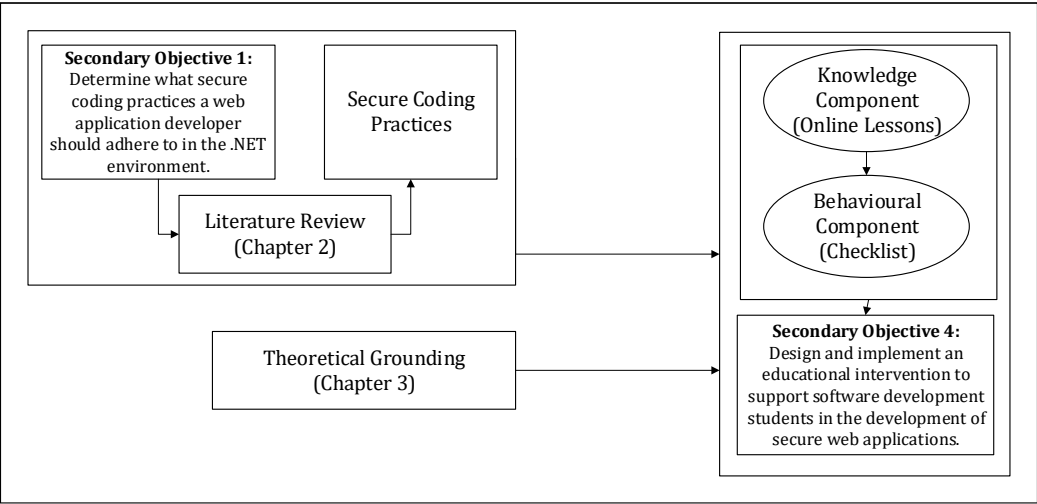


Figure 7.1: Phase 3 - Educational Intervention Process Flow

Phase 3 of this research is the educational intervention, which addresses Secondary Objective 4 *to design and implement an educational intervention to support software development students in the development of secure web applications.*

The knowledge component for this research took the form of a blended learning course that students worked through to improve their knowledge regarding secure coding practices. The behavioural compliance monitoring component for this research took the form of a checklist to monitor students’ adherence to secure coding practices.

The same checklist used in Phase 1, as discussed in Chapter 2, Section 2.7 was used by students to monitor their adherence to these practices. The next section describes the focus of the educational intervention.

7.3 Focus of Educational Intervention

The design of the educational intervention stems from the secure coding practices, which were discussed in detail in Chapter 2, Section 2.7. Table 7.1 provides the secure coding practices that informed the design of the educational intervention of this research. Both the knowledge and behavioural components of this research were designed using the identified secure coding practices in Table 7.1.

SP Number	OWASP Secure Coding Practices
SP1	Use Parameterised SQL commands for all data access, without exception.
SP2	Do not use SQL commands with a string made up of a concatenated SQL string.
SP3	Properly validate input fields.
SP4	Apply the Principle of Least Privilege when setting up the database of your choice.
SP5	When using SQL Server, prefer integrated authentication over SQL authentication.
SP6	Using stored procedures is the most effective way to counter the SQL injection vulnerability.
SP7	Encrypt connection strings.
SP8	Connection strings should be based in a configuration file.
SP9	Encrypt sensitive data using acceptable encryption algorithms.

Table 7.1: OWASP Secure Coding Practices

The following section discusses the knowledge component, which forms the first part of the educational intervention for this research.

7.4 Knowledge Component

As discussed in Chapter 3, it is necessary to educate students in order to equip them with the knowledge required to perform a task. This section discusses the knowledge component which was used to educate students regarding secure coding practices. This section first discusses the design of the knowledge component in Section 7.4.1, which provides detail on how the educational theory in Chapter 3 was used to design the knowledge component. Section 7.4.2 provides an overview of the curriculum for the knowledge component, and Section 7.4.3 provides detail on how the knowledge component was administered.

7.4.1 Knowledge Component Design

The knowledge component for this research included online lessons that the researcher designed using the identified secure coding practices as seen in Figure 7.1. For each of the lessons, the researcher explained their importance and the security implications if they were ignored. The online learning platform that was used to distribute the lessons was the *Moodle Learning Management System* that runs on the learn.mandela.ac.za site at the Nelson Mandela University. Moodle is a learning management system used by educators to create effective blended learning material for students in various higher educational institutions. Moodle has been adopted by many institutions for its cost effectiveness, its ability to expand with increased student populations, and its ability to meet the needs of institutions, students and educators (Florian & Zimmerman, 2015). A course was created on Moodle, was called the *Web Application Security* course, and the content was distributed.

Figure 7.2 provides an overview of the process followed by the students when completing the online lessons on Moodle.

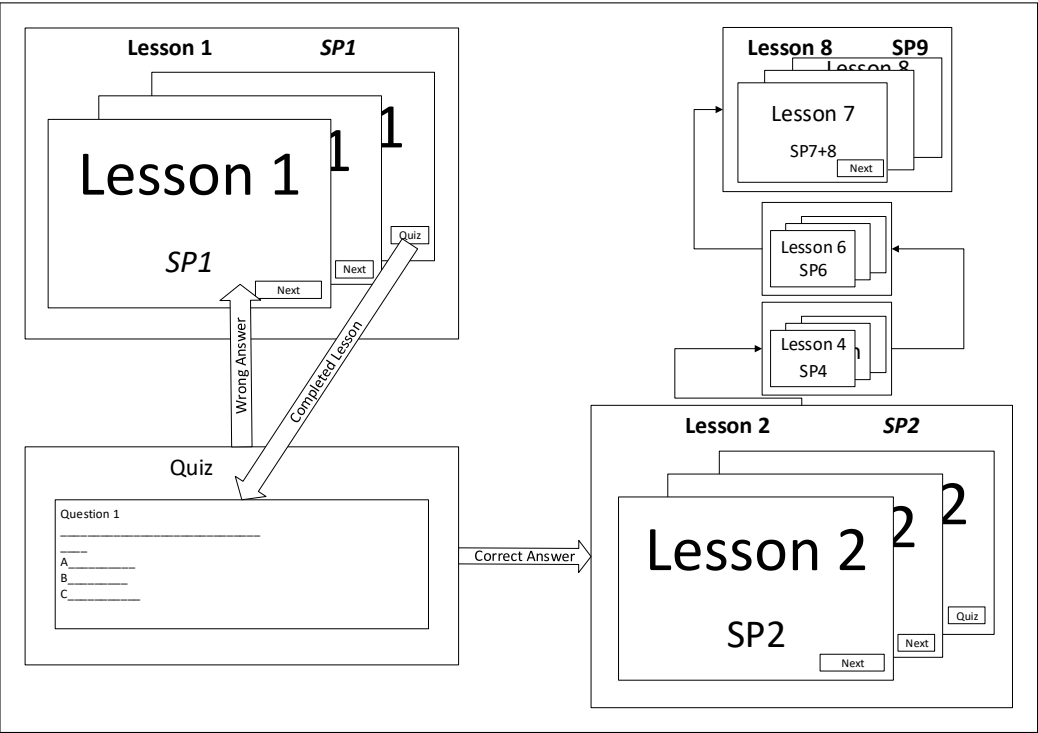


Figure 7.2: Lesson Content Process Flow

Lessons took the form of interactive Microsoft PowerPoint slides, which were converted to videos, for students to work through. Each secure coding practice was addressed in one lesson either by a video or by interactive PowerPoint slides.

After completing each lesson, the students were required to take a quiz, which allowed them to reflect on the content of the lesson. The quiz had four questions assigned to each lesson. The students had to answer only one randomly selected quiz question before continuing to the next lesson. If the students selected the incorrect answer, they were required to work through the lesson again, and if they selected an incorrect answer once again, a different question would be randomly selected. Alternatively, if they selected the correct answer, they were allowed to continue to the next lesson.

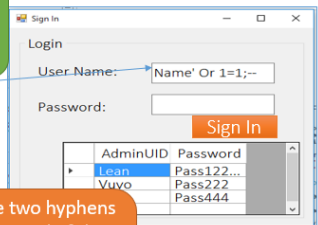
These lessons adhered to brain-compatible principles as discussed in Chapter 3, Section 3.2.3. These brain-compatible principles were integrated into the design of the knowledge component, as discussed below:

- **The search for meaning occurs through patterning:** This principle requires the instructors to find out what knowledge students already have at the beginning of the course, to adjust the lessons accordingly. These students were being taught programming in the .NET environment. Therefore, the content for the course was based on .NET, with a specific focus on web applications in the data access layer as discussed in Chapter 2.
- **Learning involves both focussed attention and peripheral perception:** In this principle, instructors are required to use visuals to present the content to the students. The use of animation or music is encouraged by this principle. The lessons were designed using Microsoft PowerPoint and included animations. In addition, the content used various colours and visuals to explain concepts in each lesson.

Concatenated SQL Strings

```
private void btnSignIn_Click(object sender, EventArgs e)
{
    string connection = "Data Source=localhost;Initial Catalog=Intervention;Integrated Security=True;User=root;Password=root";
    SqlConnection someConnection = new SqlConnection(connection);
    SqlCommand someCommand = new SqlCommand();
    someCommand.Connection = someConnection;

    someCommand.CommandText = "SELECT AdminUID, Password FROM Admin " +
        "WHERE AdminUID=" + txtUID.Text +
        " AND Password=" + txtPassword.Text + " ";
    SqlDataAdapter da = new SqlDataAdapter(someCommand);
    DataTable dt = new DataTable();
    da.Fill(dt);
    if(dt.Rows.Count > 0)
    {
        dataGridView1.DataSource = dt;
    }
    someConnection.Open();
}
```



Becomes:

```
"Select Admin, Password FROM Admin
WHERE AdminUID = Vuyo
OR 1=1; --
AND Password = Pass222"
```

The text from the inputs used as placeholders to the SQL string.

These two hyphens at the end of the query comments all that comes after the SQL statement.

--

Figure 7.3: Focused Attention and Peripheral Perception Example

Figure 7.3 illustrates a lesson containing visuals. This lesson provides an example to students on why they should adhere to concatenated SQL strings.

- **We understand best when facts and skills are embedded in natural, spatial memory:** This principle requires instructors to embed real-life activities, demonstrations or visual imaginary experiences for the students to be able to relate to the concepts being taught. This was considered in the design of the lessons for this course, as shown in Figure 7.4.

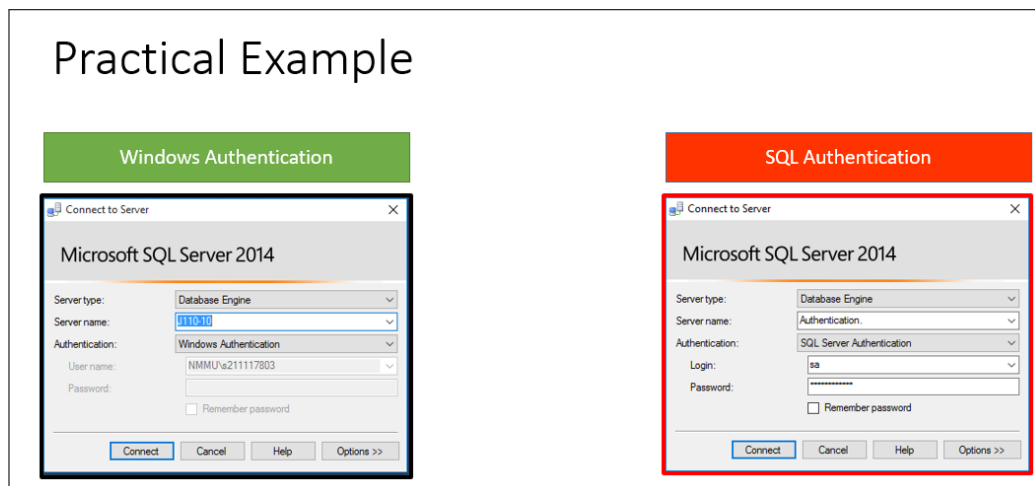


Figure 7.4: Natural, Spatial Memory Example

Figure 7.4 shows an example of a lesson with a real-world example familiar to them, to illustrate to students which authentication is best to use, as suggested by SP5 in the secure coding practices.

- **Immediate feedback amplifies learning:** This principle focusses on the lesson being given. Lessons should be divided into smaller parts to allow students to move to the next lesson only after understanding the previous one. For the lessons provided in this course, students were required to work through a single lesson for each secure coding practice. After working through each lesson, they were required to complete a quiz for that specific lesson, before moving on to the next lesson. If the students chose the incorrect answer, they were required to repeat the lesson content, as shown in Figure 7.5.

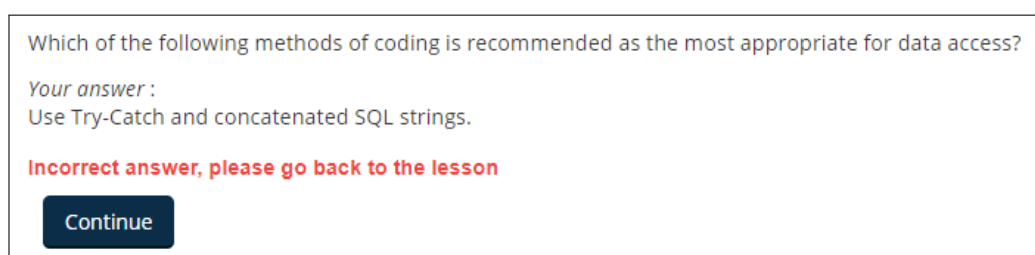


Figure 7.5: Immediate Feedback Example

- **Learners should be given choices to accommodate different**

learning styles: This principle encourages instructors to provide students with enough time to work through the lessons, in order to cater for different learning styles. This principle responds to one of the goals of blended learning courses, which promotes a student-centered learning approach. The Web Application Security course was made available to the students for a week, giving them sufficient time to work through the online lessons.

As seen from this discussion, the knowledge component adhered to all the brain-compatible principles mentioned in Chapter 3, Section 3.2.3. The next section discusses the overview of the curriculum for the knowledge component of the education intervention for this research. .

7.4.2 Overview of the Curriculum

This section provides details of how each secure coding practice was addressed in the knowledge component of the educational intervention for this research. This section first provides details on how a student would work through a single lesson for a secure coding practice. The example of a secure coding practice that will be used is SP2, use of *concatenated SQL strings*. Students would be presented with a video, which they would watch containing the secure coding practice as shown in Figure 7.6.

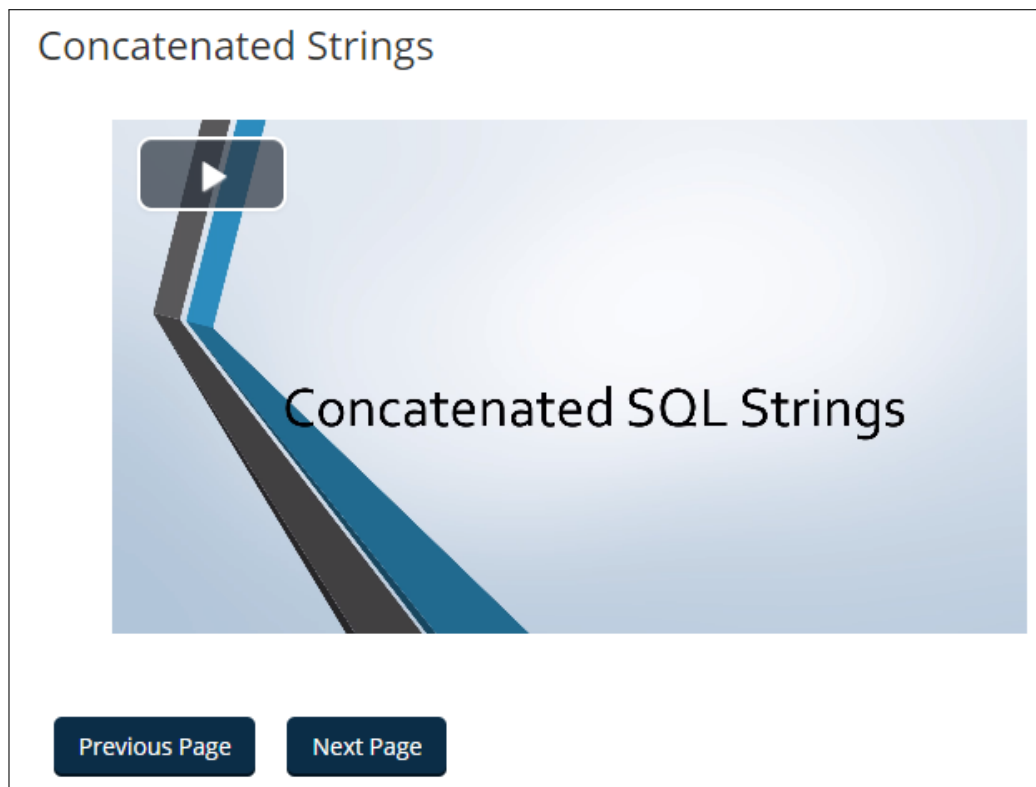


Figure 7.6: Lesson Introductory Slide (SP2)

Once the student clicks on the video shown in Figure 7.6, the video starts, and does not contain audio. Only *Lesson 6* which contained a YouTube video contained audio for which students were informed to bring headphones.

Do not use SQL Commands with a string parameter made up of a concatenated SQL string.

- A SQL Command with a string parameter made up of a concatenated SQL string is also a type of a SQL query that requires at least one parameter for execution.
- The difference between this type of SQL command and the normal SQL Command is that the placeholder is a string parameter that is built from user input.
- The following is an example of a SQL command with a string:
 - `"SELECT UserName, Password FROM User WHERE UserName = ' " + txtUserName.Text + " ' AND Password = ' " + txtPassword.Text + " '";`
- *txtUserName and txtPassword are user input fields.*

Figure 7.7: Definition of Concepts and Examples (SP2)

Figure 7.7 provides the description about the concatenated SQL strings secure coding practice and an example. The contents in this slide helped the students understand the examples that follow in the lesson, as shown in Figure 7.8.

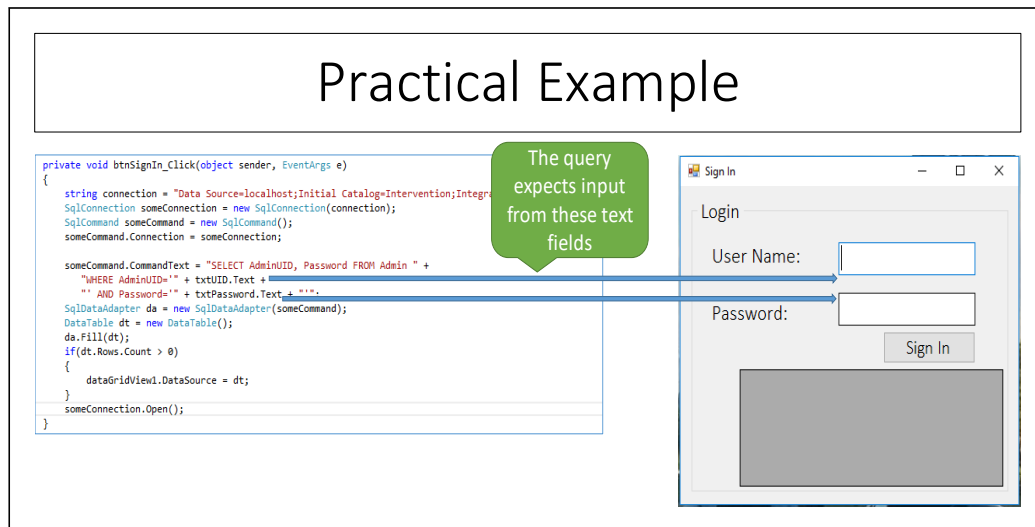


Figure 7.8: Lesson Practical Example (SP2)

Figure 7.8 provides an example of a scenario where it would be possible for students to use *concatenated SQL strings*. The example used illustrates a *Sign In* simulation for an administrator user. The Admin table would consist of login details for administrators such as AdminUID and Password. Once the administrator inserts their details in the Username and Password input fields, the administrator will be granted access if the login details are correct. Access granted would be indicated by loading the data grid view with their Username and Password. The data grid view is represented by the grey shaded area below the *Sign In* button.

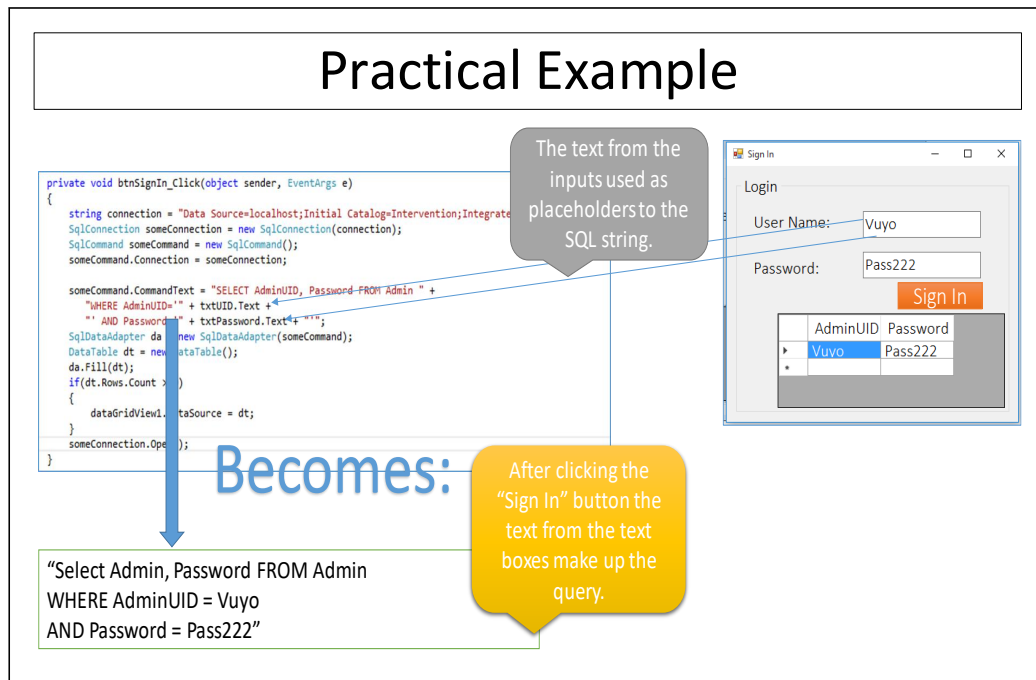


Figure 7.9: Lesson Practical Example - Continued (SP2)

Figure 7.9 is a continuation from Figure 7.8, and shows the action that will follow after the user clicks the *Sign In* button. Figure 7.9 shows what the SQL statement will look like after the button has been clicked and the data grid view showing that the correct login details have been entered, and displays the details.

Practical Example

```
private void btnSignIn_Click(object sender, EventArgs e)
{
    string connection = "Data Source=localhost;Initial Catalog=Intervention;Integrated Security=True;";
    SqlConnection someConnection = new SqlConnection(connection);
    SqlCommand someCommand = new SqlCommand();
    someCommand.Connection = someConnection;

    someCommand.CommandText = "SELECT AdminUID, Password FROM Admin " +
        "WHERE AdminUID=" + txtUID.Text +
        " AND Password=" + txtPassword.Text + " ";
    SqlDataAdapter da = new SqlDataAdapter(someCommand);
    DataTable dt = new DataTable();
    da.Fill(dt);
    if(dt.Rows.Count > 0)
    {
        dataGridView1.DataSource = dt;
    }
    someConnection.Open();
}
```

The text from the inputs used as placeholders to the SQL string.

These two hyphens at the end of the query comments all that comes after the SQL statement.

Becomes:

"Select Admin, Password FROM Admin
WHERE AdminUID = Lean'
OR 1=1; --
AND Password = Pass222"

--

AdminUID	Password
Lean	Pass122...
Vuvo	Pass222
	Pass244
	Pass444

Figure 7.10: Lesson Practical Example - Vulnerability Illustration (SP2)

Figure 7.10 illustrates how an application can be attacked when using concatenated SQL strings. The data grid view shows that the attacker would have access to all the login details. The complete SQL statement is shown at the bottom corner of the slide to make students aware of how it works.

Which ONE of the following SQL statements would be the most vulnerable to SQL injection?

☐ SELECT UserID, Password FROM Users WHERE UserID = txtUserID AND Password = txtPassword

☐ SELECT UserID, Password FROM Users WHERE UserID = @UserID AND Password = @Password

☐ All of the above.

☐ SELECT UserID, Password FROM Users

Submit

Figure 7.11: Lesson Question Example

Once the student has worked through the lesson, they are required to take one quiz question before they move to the next lesson, as discussed in

Section 7.4.1. Figure 7.11 illustrates a student quiz where a student chooses the wrong answer, which means that they will have to work through the lesson again. After they click the *Continue* button, a message will be displayed to them as shown in Figure 7.12.

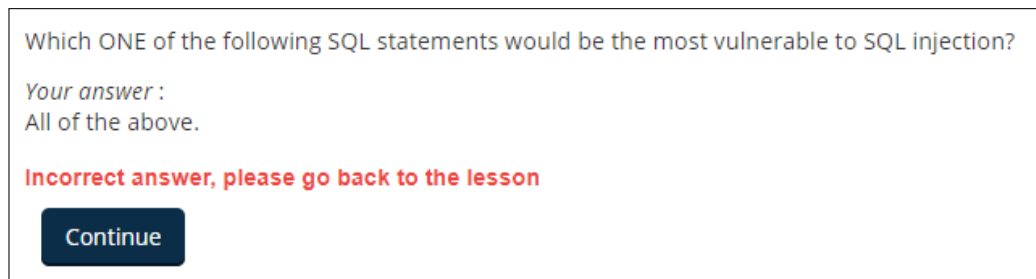


Figure 7.12: Example of Student's Incorrect Answer

Next, an overview of what each secure coding practice focused on in the knowledge component, is discussed. The secure coding practices are discussed as listed in Table 7.1.

- SP1 (Using Parameterised SQL commands): The content for this secure coding practice firstly provides the background relating to parameterised SQL commands in order to equip students with the necessary information relating to this secure coding practice. The remainder of the lesson shows the students how parameterised SQL commands can be implemented in their code, and why it is necessary to use them.
- SP2 (Concatenated SQL strings): Content for this secure coding practice begins by introducing what is meant by concatenated SQL strings. The lesson proceeds by showing how programmers make use of concatenated SQL strings and the negative implications of using them. This lesson also provides a way in which to avoid using concatenated SQL strings, which is by means of parameterised SQL commands.
- SP3 (Input validation): The content for this secure coding practice begins by discussing validation in general. It also discusses the various types of validation, such as blacklisting and whitelisting, and why they are important. The content also provides suggestions on what to use when dealing with validation, for example, ASP.NET Regular Expressions to tell input fields which values to accept.

- SP4 (Principle of Least Privilege): This secure coding practice content explains what the Principle of Least Privilege is and why it is important when developing web applications. This content also provides a scenario where the use of this secure coding practice is shown and how it can be implemented.
- SP5 (Authentication): The content of this secure coding practice was addressed by means of a video adapted from YouTube. The video was embedded in the slides and distributed as a single lesson to the students to listen to and to watch. The link to the video used from YouTube can be found on the following link <https://www.youtube.com/watch?v=Vo4iAxo0rJE>.
- SP6 (Using Stored Procedures): The content for this secure coding practice focusses on how stored procedures are used and why they are important, providing examples on how they should be implemented in a web application.
- SP7 + SP8 (Connection strings): These two secure coding practices both deal with connection strings, and have been addressed collectively in the same lesson. The content first provides detail about the importance of connection strings, and how they should be handled when developing web applications, providing detail on how to implement both the secure coding practices.
- SP9 (Encryption): In this secure coding practice, the researcher uses an analogy to explain the concept of encryption to the students. The lesson further explains the analogy to clarify the concepts for the students. Since OWASP provides recommendations relating to acceptable encryption algorithms, the content for this lesson also emphasises the use of the encryption algorithms recommended by OWASP when developing web applications.

All the lessons were followed by a quiz question to check students' understanding of the content contained in the lesson they had worked on. The results for the short content quizzes were not recorded, since answers were simply used to ensure that students do not move to the next lesson without understanding the content in the previous lesson.

7.4.3 Administering the Knowledge Component

The Web Application Security lessons were prepared by the researcher and distributed to the students on Moodle between 28 July and 4 August 2017. The students were permitted to work through the lessons as often as they wanted. During a lecture, the researcher explained the process that the students needed to follow when completing the online content. Most students worked through the content as soon as it was made active and available to them. The students worked through the online content in the computer laboratories at the university. A total of 120 students completed the online lessons. The students had to work through all the lessons, because they were encouraged to take a quiz for which marks were awarded. The quiz is the verification of the knowledge component for this research, and is discussed in Chapter 8.

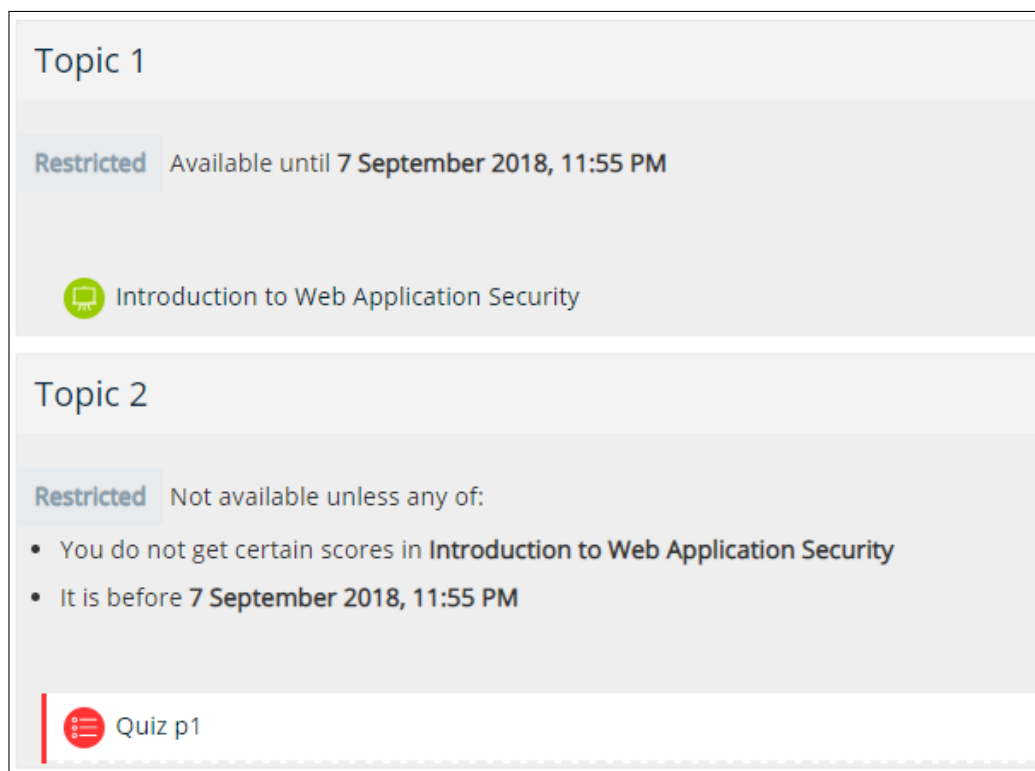


Figure 7.13: Web Application Security Course and Quiz

Figure 7.13 shows that students had to work through the lessons before they could take the final quiz. The dates have changed as shown in Figure 7.13 as this same material was distributed to 2018 students to use to improve

their knowledge relating to secure coding practices.

7.5 Behavioural Compliance Monitoring Instrument

The first part of this chapter provides details relating to the knowledge component of the educational intervention. Although it is possible for a student to have the knowledge and not perform accordingly when developing their web applications, it is most unlikely for them to behave accordingly when they do not have the requisite knowledge. Therefore, it was deemed necessary to educate the students on secure coding practices and then to monitor their adherence to these practices. This section provides details on how the behaviour of students was monitored when developing their web applications. It first discusses the design of the instrument in Section 7.5.1, while conducting the behavioural compliance instrument is discussed in Section 7.5.2. As mentioned previously, this is the same checklist used for the code reviews of the 2015 capstone projects, as discussed in Chapter 2, Section 2.7.

7.5.1 Design of Behavioural Compliance Instrument

The behavioural compliance instrument took the form of a checklist. Table 7.2 provides the checklist that was used by the students to monitor their behavior relating to secure coding practices.

SP	Questions
SP1	Do they make use of parameterised SQL commands for all data access? <i>Yes / No (Number of Instances)</i>
SP2	Do they make use of concatenated strings in the queries? <i>Yes (Number of Instances) / No</i>
SP3	Are all input fields validated? <i>Validated / Not Properly Validated / Not applicable</i>
SP4	Do they make use of the Principle of Least Privilege when setting up their databases? <i>Yes / No</i>
SP5	Do they use integrated authentication or do they use SQL authentication? <i>Integrated Authentication / SQL Authentication</i>
SP6	Do they use stored procedures for their queries? <i>Yes / No / Inconsistent Use of Stored Procedures and Queries</i>
SP7	Do they encrypt their connection strings? <i>Yes / No</i>
SP8	Does the connection string only appear once in the web.config file? <i>Yes / No</i>
SP9	Is all the sensitive data being encrypted using the OWASP recommended methods? <i>Encrypted Using Acceptable Method / No Encryption / Encrypted Not using Acceptable Method</i>

Table 7.2: Code Review Checklist

The conducting of the behavioural compliance instrument is discussed in the next section.

7.5.2 Conducting the Behavioural Compliance Instrument

The code review checklist in Table 7.2 was provided to the students electronically via Moodle. In addition, during a lecture the researcher explained to the students how they should go about using the checklist to review their capstone projects. They were required to check all web forms accessing the data access layer against the secure coding practices for SP1 to SP9. Having worked through the knowledge component, as discussed in Section 7.4, the students should have acquired the relevant knowledge relating to the secure coding practices that should be implemented in their web applications.

Since most students worked in groups when developing their web applications, they were also required to conduct a peer code review on each other's web forms using the checklist provided. The peer code review helped the students to double check whether they had really adhered to the secure coding practices as indicated in their own code reviews. Feedback from the students was positive and most students found the checklist helpful in their code and to ensure compliance to the secure coding practices. The next section concludes the chapter.

7.6 Conclusion

As indicated in Phases 1 and 2 of this research, there was a general lack of knowledge and behavioural compliance regarding secure coding practices. Phase 3 set out to address the problem of non-adherence and lack of knowledge through an educational intervention.

The knowledge component of the educational intervention was responsible for providing students with knowledge regarding secure coding practices. Having completed the online course, the students were expected to implement the learnt secure coding practices in their capstone projects.

The behavioural aspect of the educational intervention was responsible for monitoring the students' behavioural compliance regarding secure coding practices.

Once students had completed the educational intervention, it was necessary to determine the effectiveness of the educational intervention. A tool was developed to measure students' knowledge and behavioural compliance after completing the educational intervention. The next chapter focusses on the verification of the educational intervention.

Chapter 8

Verification of Educational Intervention

The purpose of this chapter is to provide an overview of the verification phase of this research. The verification addresses both components of this research, namely knowledge and behavioural compliance. The verification phase determines the effectiveness of Phase 3, which is the educational intervention for this research.

<div>Chapter 1 Introduction</div>			
<div>Chapter 2 Web Application Security</div>		<div>Chapter 3 Theoretical Grounding</div>	
<div>Chapter 4 Research Process and Design</div>			
<div>Solution</div>			
<div>Chapter 5 Phase 1 - Behavioural Analysis</div>	<div>Chapter 6 Phase 2 - Knowledge Assessment</div>	<div>Chapter 7 Phase 3 - Educational Intervention</div>	<div>Chapter 8 Phase 4 - Verification</div>
<div>Chapter 9 Conclusion</div>			

8.1 Introduction

This research focused on two aspects, namely knowledge and behavioural compliance of students engaged in their third year software development capstone projects. These students were generally lacking in both these aspects, as seen in Phases 1 and 2. Phase 3 addressed both these aspects through an educational intervention. The purpose of this phase is to determine the effectiveness of this educational intervention in both the students' knowledge and behavioural compliance regarding the identified secure coding practices. The verification of the knowledge component was achieved through an online quiz distributed to the students through the Moodle site. Verification of the behavioural compliance component took the form of a code review by the researcher on the 2017 capstone projects.

The verification of both components are discussed in this chapter. Firstly, the verification of the knowledge component is described in Section 8.3, then the verification of the behavioural compliance component is described in Section

8.4.

8.2 Verification Process Flow

The steps followed for the verification process of this research are illustrated in Figure 8.1.

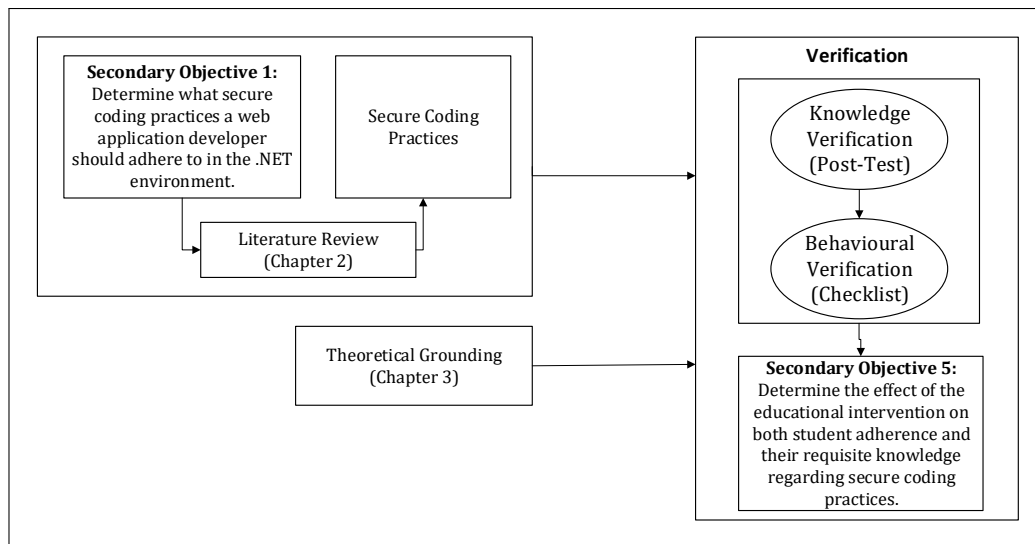


Figure 8.1: Phase 4 - Verification Process Flow

Phase 4 is the final phase for this research which is the verification of the educational intervention for this research. The knowledge and behavioural verification instruments were developed using the secure coding practices identified in the literature review in Chapter 2.

The knowledge verification took the form of an online quiz that was distributed to the students through the Moodle site. The researcher conducted a code review on the 2017 capstone projects, using the same checklist as that used for the code review in Phases 1 and Part 2 (Behavioural Compliance Monitoring) of the educational intervention.

The verification of the components address Secondary Objective 5, which is *to determine the effect of the educational intervention on both student adherence and their requisite knowledge regarding secure coding practices*. The next section provides the knowledge verification part for this research.

8.3 Knowledge Verification

The knowledge part of this chapter provides detail regarding the verification of the knowledge component discussed in the educational intervention phase in Chapter 7, Section 7.4. Section 8.3.1 first describes the design of the knowledge verification instrument, while conducting the instrument, and the results of the knowledge verification follow in Sections 8.3.2 and 8.3.3 respectively.

8.3.1 Knowledge Verification Instrument Design

The knowledge verification for this research took the form of a questionnaire, which was distributed to the students using the Moodle site. The theory which informed the design of the questionnaire, as discussed in Chapter ??, Section ??, was used in the design of this questionnaire. Some of the questions for this post-test questionnaire were adopted from the questionnaire in Phase 1 (pre-test) of this study. This questionnaire consisted of 23 questions, which the students were given one hour to answer. The full set of the questions are presented in Appendix B.

8.3.2 Conducting the Knowledge Verification Instrument

The setup for the post-test questionnaire was such that students were only allowed to work through the post-test after they had completed the lessons in the knowledge component of the educational intervention referred to as the Web Application Security Course. Students completed the post-test between 28 July and 4 August 2017, after completing the educational intervention. The students were only allowed to work through the post-test once. There were 113 students who completed the post-test. The results for the post-test questionnaire were automatically recorded on the Moodle site, where the researcher was able to export the data to an Excel spreadsheet for analysis. The next section provides the results of the knowledge verification (post-test).

8.3.3 Knowledge Verification Results

The student percentages for the knowledge verification are shown in Figure 8.2.

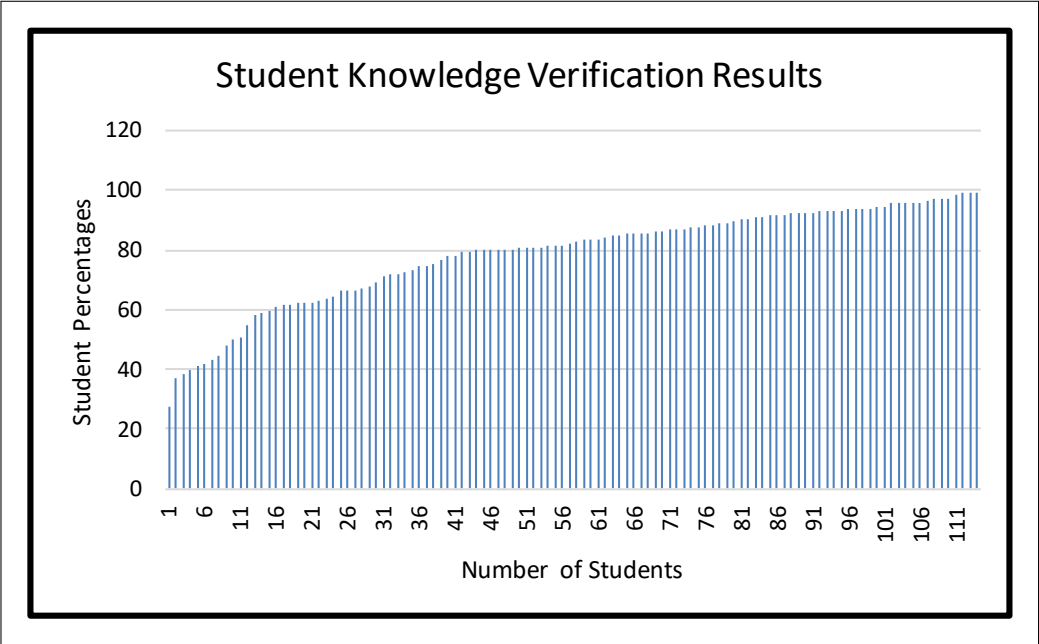


Figure 8.2: Knowledge Verification Results (Percentages per Student)

The results indicate that the students’ knowledge regarding secure coding practices improved having completed the Online Web Application Security course. Only 13 students of the 113 (11%) scored a mark lower than 50% with most students having scored above 60%.

SP Number	OWASP Secure Coding Practices	Questions(Q)
SP1	Use Parameterised SQL commands for all data access, without exception.	Q1, Q2
SP2	Do not use SQL command with string made up of a concatenated SQL string.	Q3, Q4
SP3	Properly validate input fields.	Q16, Q17, Q18
SP4	Apply the Principle of Least Privilege when setting up the database of your choice.	Q13, Q14, Q15
SP5	When using SQL Server, prefer integrated authentication over SQL authentication.	Q7, Q8, Q9
SP6	Using stored procedures is the most effective way to counter the SQL injection vulnerability.	Q5, Q6
SP7	Encrypt connection strings.	Q19, Q20
SP8	Connection strings should be based in a configuration file.	Q21, Q22
SP9	Encrypt sensitive data using acceptable encryption algorithms.	Q10, Q11, Q12

Table 8.1: OWASP Secure Coding Practices Related Questions (Post-Test)

Table 8.1 presents the questions relating to each secure coding practice, while Figure 8.3 depicts the average student marks per question and related secure coding practice.

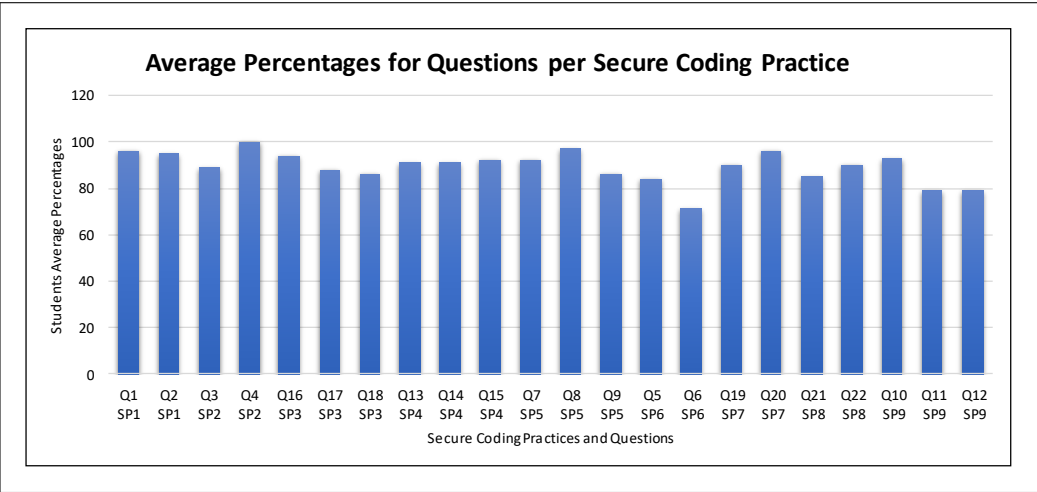


Figure 8.3: Knowledge Verification Results per Secure Coding Practice and Related Questions

Figure 8.3 shows the results for the questions relating to each secure coding practice. The results are grouped according to the secure coding practices (SP1 to SP9) rather than by question order. The results indicate that students understood the content provided in the educational intervention. The lowest average was obtained for *Q6 SP6*, where the average percentage was 70%. The highest average was obtained for *Q4 SP2* where the average was 100%. Of the average percentages for questions, only three questions recorded an average mark less than 80%; including *Q6 SP6*, *Q11 SP9* and *Q12 SP9*. Whereas *SP6* deals with the use of stored procedures, *SP9* focusses on encryption and OWASP recommended encryption methods.

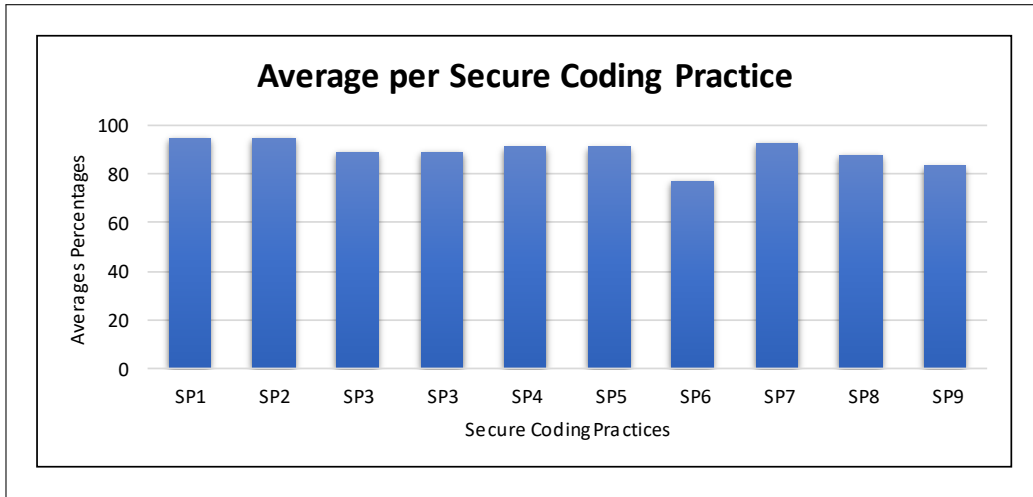


Figure 8.4: Knowledge Verification Results per Secure Coding Practice

Figure 8.4 shows the results for the knowledge verification per secure coding practice. These results consolidate the results shown in Figure 8.3, for each secure coding practice. Of the 9 secure coding practices, 5 (SP1, SP2, SP4, SP5 and SP7) have averages of 90% and above, and 3 (SP3, SP8 and SP9) have averages between 80% and 89 %. Only one secure coding practice (SP6) is between 70% and 80%, which is SP6.

Phases	SP1	SP2	SP3	SP4	SP5	SP6	SP7	SP8	SP9
Phase 2	74%	36%	30%	58%	26%	39%	20%	3%	1%
Phase 4	95%	95%	89%	91%	91%	77%	93%	88%	83%

Table 8.2: Knowledge Assessment and Verification Results (Pre-test vs Post-Test)

Table 8.2 shows the results for Phase 2, Knowledge Assessment (pre-test), and Phase 4, Knowledge Verification (post-test). As can be seen in Table 8.2, there is a substantial improvement in the students' knowledge as indicated in the second row, Phase 4. Students' knowledge has improved in all of the secure coding practices (SP1 to SP9).

These results indicate the effectiveness of the Web Application Security Course. As mentioned previously, knowledge acquisition, however, does not guarantee a change in behaviour. In order to monitor the adherence of the students' to these secure coding practices when developing their web appli-

cations, a behavioural compliance monitoring instrument was designed, as is discussed in the next section.

8.4 Behavioural Verification

The behavioural verification section firstly provides the design of the verification instrument in Section 8.4.1. Section 8.4.2 discusses how the verification was conducted, and the results for the behavioural instrument are discussed in Section 8.4.3.

8.4.1 Behavioural Verification Design

The behavioural verification instrument also took the form of a checklist, which is the same as the checklist used in Phases 1 and 3. The checklist was used by the researcher to conduct a code review on the 2017 third year capstone projects. The checklist with the nine secure coding practices (SP1 to SP9) is shown in Table 8.3.

SP	Questions
SP1	Do they make use of parameterised SQL commands for all data access? <i>Yes / No (Number of Instances)</i>
SP2	Do they make use of concatenated strings in the queries? <i>Yes (Number of Instances) / No</i>
SP3	Are all input fields validated? <i>Input Properly Validated / Input not Properly Validated / No Validation</i>
SP4	Do they make use of the Principle of Least Privilege when setting up their databases? <i>Yes / No</i>
SP5	Do they use integrated authentication or do they use SQL authentication? <i>Integrated Authentication / SQL Authentication</i>
SP6	Do they use stored procedures for their queries? <i>Yes / No / Inconsistent Use of Stored Procedures and Queries</i>
SP7	Do they encrypt their connection strings? <i>Yes / No</i>
SP8	Does the connection string only appear once in the web.config file? <i>Yes / No</i>
SP9	Is all the sensitive data being encrypted using the OWASP recommended methods? <i>Encrypted Using Acceptable Method / No Encryption / Encrypted Not using Acceptable Method</i>

Table 8.3: Code Review Checklist

The design for the checklist adhered to the checklist guidelines, as discussed in Chapter 3, Section 3.3.2. The following section describes how the behavioural verification was conducted.

8.4.2 Conducting the Behavioural Verification

The verification for the behavioural compliance was a code review on the 2017 capstone projects. The code review was conducted by the researcher before the final submission of the software development projects. The researcher first informed the students about the code review process scheduled to take place during a session in the computer laboratory. Students filled in their group names and were required to be in the computer laboratory in order for their projects to be reviewed. The code review was conducted during the students' practical sessions. For each of the capstone projects, the researcher

selected five web forms per project, which connected to the database, to be reviewed. The web forms selected were related to the capstone projects' main functionality. There were 17 groups present for the code review, and they were all reviewed successfully, in the presence of the students who belong to the group being reviewed. The data from the code review in the checklist was recorded on the Windows Application database discussed in Chapter 5, Section 5.3, (Figure 5.2). The results for the code review are discussed in the next section.

8.4.3 Behavioural Verification Results

The data from the code review was stored in the Windows Application and the percentage explained in Chapter 5, Section 5.5 was used to calculate the adherence of the students to secure coding practices, where *Percentage* = *Number of Instances actually adhered to (green)/ Overall Instances that should be adhered to (total)*100*.

	Form Number	Form Name	Number Of SQL Commands (Good)	Number Of SQL Commands With Conc Strings (Bad)	Input Fields Properly Validated	Uses Least Privileged Account	Sql Authentication	Uses Stored Procedure	Encrypting Connection String	Only Uses One Connection String	Encryption	Category Name
▶	293	LearnerRegist...	0	0	Input Property	Yes	SQL Authentic...	Uses Stored Pr...	No	Only Uses One	Uses Window...	Register
	293	LearnerRegist...	0	0	Input Property	Yes	SQL Authentic...	Uses Stored Pr...	No	Only Uses One	Uses Window...	Register
	294	Login.aspx.cs	0	0	Input Property	Yes	SQL Authentic...	Uses Stored Pr...	No	Only Uses One	Uses Window...	Login
	295	Submission.as...	0	0	Input Property	Yes	SQL Authentic...	Uses Stored Pr...	No	Only Uses One	Uses Window...	Create/Add
	296	AddNewAdmi...	0	0	Input Property	Yes	SQL Authentic...	Uses Stored Pr...	No	Only Uses One	Uses Window...	Create/Add
	297	BlockLearnera...	0	0	Input Property	Yes	SQL Authentic...	Uses Stored Pr...	No	Only Uses One	Uses Window...	Edit/Update
*												

Figure 8.5: Captured Data Example (P1)

Figure 8.5 shows the results of a single project (P1) in the verification for a particular capstone project group. The consolidated results for all the capstone projects were recorded in a Microsoft Excel spreadsheet for representation and are shown in Table 8.4.

ProjNo	SP1	SP2	SP3	SP4	SP5	SP6	SP7	SP8	SP9
P1	100%	100%	100%	100%	0%	100%	0%	100%	100%
P2	100%	100%	100%	84%	0%	100%	0%	100%	100%
P3	100%	100%	100%	80%	0%	100%	0%	100%	100%
P4	100%	100%	100%	100%	0%	100%	0%	100%	100%
P5	100%	100%	100%	100%	0%	100%	0%	100%	100%
P6	100%	100%	100%	100%	0%	100%	0%	100%	100%
P7	100%	100%	100%	100%	0%	100%	0%	100%	100%
P8	100%	100%	100%	100%	0%	100%	0%	100%	100%
P9	84%	65%	100%	65%	0%	100%	0%	100%	100%
P10	100%	100%	100%	100%	0%	100%	0%	100%	100%
P11	100%	100%	100%	100%	0%	100%	0%	100%	100%
P12	100%	100%	100%	70%	0%	100%	0%	100%	100%
P13	48%	60%	100%	56%	0%	24%	0%	100%	100%
P14	100%	100%	100%	100%	0%	100%	0%	100%	100%
P15	100%	100%	100%	100%	0%	100%	0%	100%	100%
P16	100%	100%	100%	100%	0%	100%	0%	100%	100%
P17	100%	100%	100%	100%	0%	100%	0%	100%	100%
Avg.	96%	96%	100%	91%	0%	96%	0%	100%	100%

Table 8.4: Adherence to Secure Coding Practices

Table 8.4 shows the percentages for each project and the average percentages per secure coding practice for the 2017 capstone projects according to the code review. These results indicate high levels of adherence, although two project groups (P9 and P13) did not fully adhere to some of the secure coding practices. As for two of the secure coding practices (SP5 and SP7), it is the capstone project's requirement for the student not to implement the secure coding practices in the projects since the evaluators need to have access to the capstone projects. However, in order to keep the behavioural assessment in line with the knowledge assessment, these were not removed. This was deemed necessary since the behavioural checklist must be suited to various contexts. A discussion of the results follows in the next section.

8.5 Verification Discussion

The results from the knowledge verification indicate a high level of knowledge regarding the taught secure coding practices. Fewer than 20 of the 113 students scored lower than 50% in the online post-test. This indicates that students' knowledge regarding secure coding practices improved between pre- and post-tests and most students were able to implement the secure practices.

The results from the code review indicated high levels of adherence, with 12 of the 17 capstone project groups having adhered to all the secure coding practices, except for those that the capstone project requirements permits them not to implement (SP5 and SP7). All averages per secure coding practice were between 90% and 100%.

However, five project groups deviated from one or more secure coding practice. The secure coding practices that the project groups deviated from were SP1, SP2, SP4, and SP6.

For SP1 and SP2, only P9 and P13 did not adhere entirely to these secure coding practices. There were five groups which did not fully adhere to all secure practices; these groups are P2, P3, P9, P12 and P13.

Looking closer to SP6, only one group did not adhere to this secure coding practice; that group is P13, with a percentage score of 24%.

The code review verification results indicate that most students adhered to the secure coding practices, having only two groups who did not fully adhere to three or more secure practices. SP4 also lacked full adherence, having five project groups not fully adhering to secure coding practices. The twelve other groups adhere to this secure practice, which indicates that this secure practice can be implemented.

8.6 Conclusion

The results for the verification of the knowledge and behavioural adherence indicates a positive impact on the software development capstone projects. In Phase 1, students generally lacked knowledge relating to secure coding practices and did not behave accordingly when developing their capstone projects. In Phase 3, the researcher had developed an educational interven-

tion to improve students' knowledge and their behaviour regarding secure coding practices. In terms of their knowledge, the online lessons were designed for students to go through, and for their behaviour, the students were given a checklist to check their adherence when developing their capstone projects.

The students had to work through the knowledge component first, and then to use the checklist to monitor their adherence to secure coding practices. It is known in information security that it is possible to have knowledge and not to behave accordingly (Van Niekerk & Von Solms, 2010). Therefore, the checklist served as an aid to monitor students' adherence to secure coding practices whether they forgot to implement or whether they chose to ignore it completely.

Students have shown good adherence, except for five project groups, which did not adhere to one or more secure practice. It can be concluded that it is good to educate students and to provide them with a behavioural monitoring instrument to help them with their adherence because security is important when developing web applications.

Chapter 9

Conclusion

This chapter concludes the dissertation. It provides a summary of the findings for this research. This chapter also indicates the limitations for this research, providing suggestions for future research.

<div>Chapter 1 Introduction</div>			
<div>Chapter 2 Web Application Security</div>		<div>Chapter 3 Theoretical Grounding</div>	
<div>Chapter 4 Research Process and Design</div>			
<div>Solution</div> <div><div><div>Chapter 5 Phase 1 - Behavioural Analysis</div></div><div><div>Chapter 6 Phase 2 - Knowledge Assessment</div></div><div><div>Chapter 7 Phase 3 - Educational Intervention</div></div><div><div>Chapter 8 Phase 4 - Verification</div></div></div>			
<div>Chapter 9 Conclusion</div>			

9.1 Introduction

Information is one of the most important assets of any modern day organisation (Van Niekerk & Von Solms, 2010). This information would need to be communicated using software applications such as desktop, mobile and mostly web applications. In order for the users of these software applications to be confident in using them, they have to be confident in their carrying out the information in a secure manner with the confidentiality, integrity and availability of information not being compromised.

However, this is often not the case, as in 2014 more than 76% of web applications scanned by Semantic contained known vulnerabilities (Razzaq et al., 2014). Many of these vulnerabilities are caused by programmers while developing such applications. These could be corrected easily through adhering to secure coding practices (Razzaq et al., 2014; Conklin & White, 2005).

Since programmers who develop these web applications are trained at universities, this research was conducted on student capstone projects to evaluate students' adherence to secure coding practices, and assessed students' knowledge regarding secure coding practices. This research is about teaching these secure coding practices through a blended learning approach.

The purpose of this chapter is to conclude this dissertation. Section 9.2 provides an overview of the chapters for this dissertation while Section 9.3 provides the research summary, arguing how the research objectives for this research were met. Section 9.4 reflects on the contribution of this research; Section 9.5 highlights the research limitations of this research, and Section 9.6 provides suggestions for future research. Section 9.7 provides the publications for this research, and the epilogue is discussed in Section 9.8.

9.2 Summary of Chapters

This dissertation consists of nine chapters. The first chapter is the introduction while Chapters 2 and 3 are literature review chapters. Chapter 4 presents the research process and design. The solution was split according to the four phases of this research which make up Chapters 5 to 8, and Chapter 9 concludes this dissertation.

Chapter 1 introduced the research problem and objectives for this research.

This chapter also provided the outline structure for the research.

Chapter 2 provided the background information and relevance for the study. This was achieved by examining web application security and introducing the reader to the larger field of the study with the aim of highlighting the role of programmers in secure web application development.

Chapter 3 provides an overview of the educational theory and behavioural theories, which informed the design of the educational intervention for this research. The educational theories focussed on blended learning with a specific focus on online learning and brain-compatible principles. The behavioural theory addressed the need for compliance, focussing more on evaluating feedback and on the use of a checklist.

Chapter 4 presents the research process and design for this research. It provides a discussion regarding the research setting and an explanation of the various phases that comprise the contribution of this research. This chapter also provides details regarding the research methods and approaches used in this research.

Chapter 5 presents Phase 1 of this research which addresses the behavioural analysis phase of this research. This chapter provides detail regarding the code review which took place on 2015 capstone projects, addressing the second secondary objective of this research.

Chapter 6 presents the knowledge assessment providing detail about Phase 2 which is the third secondary objective for this research. This chapter discusses the design, the conducting of the instrument and the results from the questionnaire.

Chapter 7 provides the process flow of the design and implementation of the educational intervention for this research, which addresses the fourth secondary objective. This chapter is split into two parts; firstly the knowledge component, and then followed by the behavioural component. This chapter represents Phase 3 of this research.

Chapter 8 presents Phase 4, which is the final phase for this research and the fifth secondary objective. It provides the verifications for this research, and provides results for the effectiveness of the educational intervention. This chapter is verification for both the knowledge and behavioural aspects.

Finally, *Chapter 9*, concludes this dissertation. This chapter summarises the research findings and provides the limitations of this research providing

suggestions for future research.

9.3 Meeting the Research Objectives

This section provides a summary of how the research objectives for teaching secure coding practices through a blended learning approach were met. The primary objective for this research was as follows: **Primary Objective:** *To develop a framework for teaching secure coding practices through a blended learning approach.* In order to address the primary objective, the following five secondary objectives were met as follows:

Secondary Objective 1: *To determine what secure coding practices a web application developer should adhere to in the .NET environment.* In order to determine these secure coding practices, the researcher conducted a literature review on various organisations and previous research regarding secure coding practices which included OWASP, ISO/IEC, MSDN and NIST, as discussed in Chapter 2, Section 2.6. OWASP was chosen for this research as it suited the needs of this research, in terms of what is recommended for guidance in applications development, and focusses on the .NET environment, as discussed in Chapter 2, Section 2.6.

Secondary Objective 2: *To determine the adherence of third year software development capstone projects to the identified secure coding practices.* The purpose of this objective was to measure the adherence of capstone project students to the secure coding practices identified in Secondary Objective 1. The capstone projects should at a minimum have these secure coding practices implemented when developed by the students. Adherence to these secure coding practices was determined by conducting a code review on 2015 capstone projects developed in the .NET environment. The code review indicated that students' developing their capstone project lacked behavioural compliance regarding these secure coding practices. This is in line with the problem statement of this research, which states that *undergraduate software development students do not consistently adhere to secure coding practices when developing their third year capstone projects, thereby leading to vulnerabilities in their web applications.*

Secondary Objective 3: *To determine whether third year software development students have the requisite knowledge relating to secure coding practices.*

This objective was achieved through a questionnaire, which was distributed to the students in class for them to answer. The secure coding practices identified in Secondary Objective 1 informed the design of the questionnaire. The results from the questionnaire indicated that students lacked the requisite knowledge relating to secure coding practices.

Secondary Objective 4: *To design and implement an educational intervention to support software development students in the development of secure web applications.* The educational intervention for this research addressed both the knowledge and behaviour of the students. The theoretical grounding in Chapter 3 informed the design of the educational intervention. For their knowledge, the researcher developed blended learning lessons using the Moodle Course Management System, which is an online platform. In terms of their behaviour, the students were given a checklist to monitor their adherence to secure coding practices whilst developing their third year capstone projects. The secure coding practices were used to design both the knowledge and behaviour instruments. The lessons were made available to the students before they were given the checklist, so that they had the requisite knowledge relating to secure coding practices.

Secondary Objective 5: *To determine the effect of the educational intervention on both student adherence and their requisite knowledge regarding secure coding practices.* The verification included both knowledge and behavioural components. In terms of the knowledge component, the researcher designed a questionnaire using the secure coding practices identified in Secondary Objective 1. For the behavioural component, the same checklist used in Secondary Objectives 2 and 3 was used. The checklist was used by the researcher to conduct a code review on 2017 capstone projects before the students submitted their web applications for final assessment. The results from both the knowledge and behavioural verification showed a positive impact on both student adherence and their knowledge relating to secure coding practices.

9.4 Research Contribution - The Framework

This section provides the research contribution by outlining how the framework was developed. This research consisted of key elements which were

important in the development of the framework. Section 9.4.1 provides the key elements for this research, and Section 9.4.2 provides an overview of the framework.

9.4.1 Research Key Elements

Figure 9.1 illustrates the key elements and their relationships which resulted from this research.

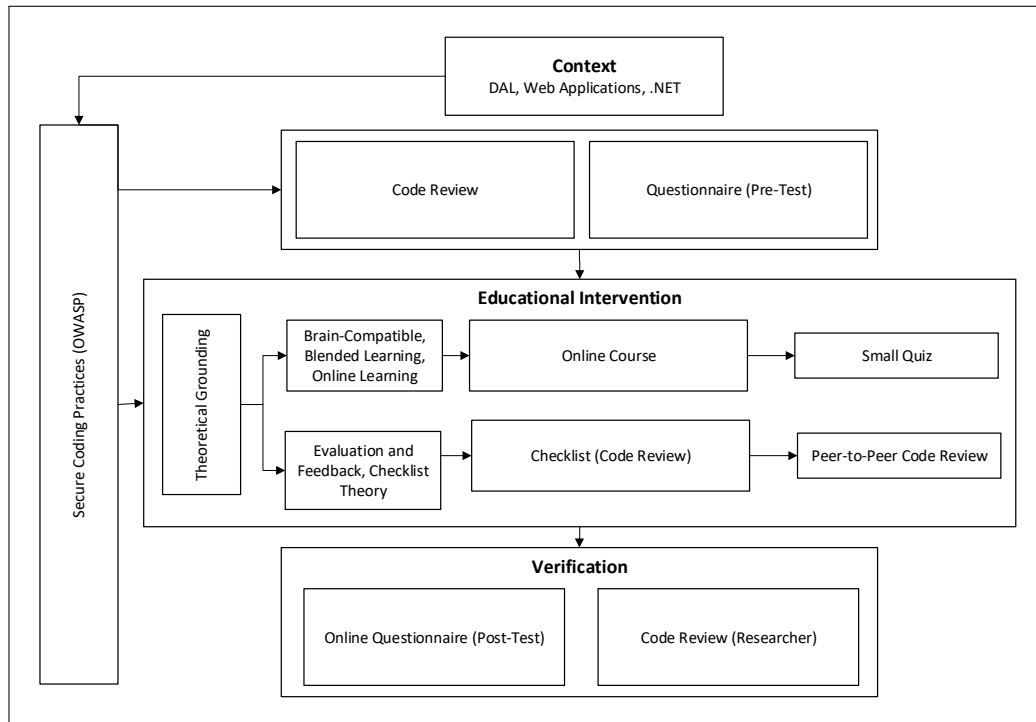


Figure 9.1: Key Elements

This research is about teaching secure coding practices to software development students through a blended learning approach. In order to know what secure coding practices should be taught, the researcher defined the context that the secure coding practices had to address. This research focussed specifically on the data access layer of web applications developed in the .NET environment. A literature review followed, which was to identify secure coding practices that align with the context for the research focus. The secure coding practices for this research were identified from OWASP. These secure coding practices have been used through out this research. Firstly, the secure coding practices were used to conduct a code review to

determine the adherence of 2015 capstone projects to the identified secure coding practices. This was conducted using a checklist based on secure coding practices. The aim of the code review was to determine the behaviour of students when developing their web applications. After the code review, the results were analysed, showing low levels of adherence and inconsistencies when developing their web applications.

The next step in the research was to assess the students' knowledge relating to these secure coding practices. This was achieved by using a questionnaire which served as a pre-test. The results from the knowledge assessment showed that most students lacked the requisite knowledge relating to secure coding practices.

After the knowledge and behaviour results were analysed there was a need to address the lack of knowledge and low levels of adherence through an educational intervention. The knowledge component was addressed through online lessons and the behavioural component was addressed by using checklist provided to the students to monitor their compliance with secure coding practices.

This research included a theoretical grounding, which informed the design of the educational intervention. The theoretical grounding for the knowledge component included theory on brain-compatible principles, blended learning and online learning. For the behavioural component, the theory included evaluation and feedback, and theory on how to create checklists.

For each lesson in the knowledge component, students were required to complete a simple quiz to ensure that students had worked through the lesson. For the behavioural component, students were required to review fellow group members' work to be sure that they implemented secure coding practices.

In order to determine the effectiveness of the educational intervention, verification for both the knowledge and behavioural component were required. The knowledge verification took the form of an online questionnaire, which served as a post-test provided to 2017 students to complete. For the behavioural verification, the researcher conducted a code review on the 2017 capstone projects to measure their adherence to secure coding practices.

The results from both the knowledge and behaviour verification showed an improvement in knowledge and a high level of adherence to secure coding practices. These key elements have been effective in teaching secure coding

practices through a blended learning approach; and therefore, informed the development of the proposed framework, as is discussed in the next section.

9.4.2 A Generic Framework for Teaching Secure Coding Practices

Figure 9.2 illustrates a graphical representation of the proposed framework resulting from this research.

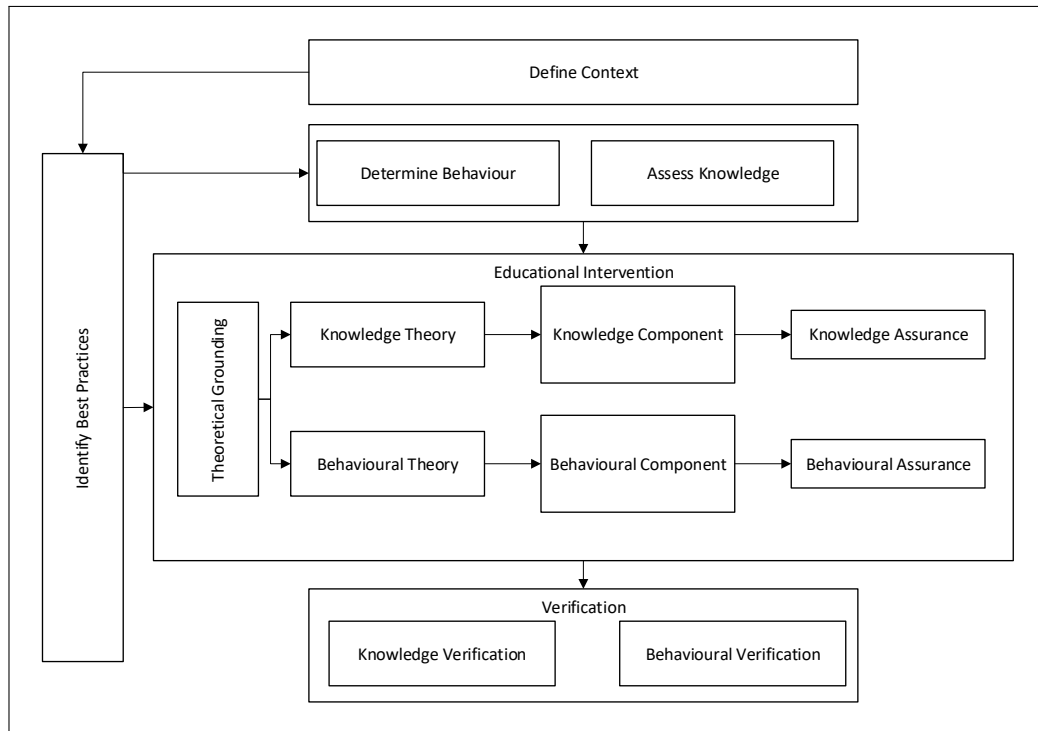


Figure 9.2: A Generic Framework for Teaching Secure Coding Practices

- **Define Context:** When teaching secure coding practices through a blended learning approach, the context in which the secure coding practices will be taught needs to be defined.
- **Identify Best Practices:** The defined context would influence the best practices to consider when conducting a literature review for identifying secure coding practices.
- **Assess Knowledge and Analyse Behaviour:** The identified secure coding practices can be used to determine the adherence and to assess

the knowledge of students relating to those secure coding practices. After the analysis of the results of both the knowledge and behavioural aspects, the researcher can address these aspects through an educational intervention.

- **Educational Intervention:** The educational intervention should address the knowledge and behavioural aspects. For the knowledge component, the students should be educated. The traditional classroom education can be augmented with any form of educational approach that will be effective in improving students knowledge relating to secure coding practices.
- **Identify Theory for Educational Intervention:** The necessary theory required to design the educational approach would need to be gathered so that the content for the knowledge and behavioural components is delivered in an effective manner. The developer of the educational intervention would have to employ some techniques to ensure that the students work through and apply the learnt knowledge when developing their applications.
- **Verification:** After the students have worked through the educational intervention, it is necessary to determine the effectiveness of the educational intervention. In doing so, the verification will depend on the nature of the entire educational intervention. The verification should also be based on the identified secure coding practices, which will depend on the context of the study.

The next section provides an overview of the limitations of this research.

9.5 Research Limitations

This research was conducted at the Nelson Mandela University. The students selected for this research were third year students studying towards the National Diploma: Software Development. The students who participated had to be engaged in their third year capstone projects. In terms of the capstone projects, the researcher used the 2015 and 2017 projects to conduct a code review. The code review only focussed on the data access layer of

web applications developed in the .NET environment, using the identified secure coding practices. These secure coding practices were used to teach students using a blended learning approach. The following section provides suggestions for future research.

9.6 Suggestions for Future Research

As stated in the research limitations section, this study only focussed on the secure coding practices for the data access layer in the .NET environment, within the Nelson Mandela University. This research therefore focussed only on a small segment of the secure coding practices.

As this research focussed on capstone projects, future research can focus on educating students in introductory programming classes whilst they are still learning programming. More educational approaches to teach these secure coding practices could be employed to improve the secure coding knowledge of software development students. Future research can focus on more secure coding practices focussing on other layers, such as the presentation and business logic layers.

9.7 Publication

The following publication stemmed directly from this dissertation and is attached in Appendix A:

- Mdunyelwa, V. S., Van Niekerk, J. F., & Futch, L. A. (2017). Secure Coding Practices in the Software Development Capstone Project. Human Aspects of Information Security & Assurance (HAISA 2017), Adelaide, Australia, 28 - 30 November 2017.

9.8 Epilogue

Most universities that teach programming focus on the functionality that the application provides, with little consideration for security resulting in web applications containing vulnerabilities. These vulnerabilities can easily be solved, by introducing secure coding practices within the curriculum.

It is argued that these secure coding practices should be introduced at a university level to undergraduate students. Various behavioural monitoring mechanisms can be used to monitor their compliance with these secure coding practices, such as checklists which are normally used in software development to evaluate software developers' code. Educating and monitoring students can equip them with secure coding knowledge and help them to monitor their behaviour relating to secure coding practices. The framework proposed by this research could provide the necessary guidance to help educators teach secure coding practices.

References

- ACM. (2018). *Key ACM Education Activities*.
- Aggarwal, N., Gupta, R., & Saxena, P. (2014). Comparative Study of OSI & TCP / IP Reference. 2(Xi), 64–68.
- Al-Arimi, A. M. A.-K. (2014). Distance Learning. *Procedia - Social and Behavioral Sciences*, 152, 82–88.
- Bakia, M., Shear, L., Toyama, Y., & Lassetter, A. (2012). Understanding the implication of Online Learning for Educational Productivity.
- Beal, V. (1999). *The 7 Layers of the OSI Model*. (https://www.webopedia.com/quick_ref/OSI_Layers.asp, Last Accessed 2018-04-08)
- Beal, V. (2018). *Application (Application software)*. (<https://www.webopedia.com/TERM/A/application.html>, Last Accessed 27 April 2018)
- Benbasat, I. (2010). Information Security Policy Compliance: An Emperical Study of Rationality-Based Beliefs and Information Security Awareness. 34(3), 523–548.
- Bird, D. K. (2009). The use of questionnaires for acquiring information on public perception of natural hazards and risk mitigation – a review of current knowledge and practice. 1307–1325.
- Bishop, M. (2002). *Computer Security: Art and Science*. Addison Wesley.
- Bishop, M. (2003). *Software protection and application security: Understanding the battleground*.

- Bishop, M., & Orvis, B. J. (2006). A Clinic to Teach Good Programming Practices. *Proceedings of the 10th Colloquium for Information Systems Security Education*, 168–174.
- Blatchford, P., Bassett, P., & Brown, P. (2011). Examining the effect of class size on classroom engagement and teacher-pupil interaction: Differences in relation to pupil prior attainment and primary vs. secondary schools. *Learning and Instruction*, 21(6), 715–730.
- Bowling, A. (2005). Mode of questionnaire administration can have serious effects on data quality. *Journal of Public Health*, 27(3), 281–291.
- Bryman, A. (2013). Social research methods. *Journal of Chemical Information and Modeling*, 53(9), 1689–1699.
- Burley, D., Bishop, M., Buck, S., Ekstrom, J., Fitcher, L., & Gibson, D. (2017). *Cybersecurity Curricula 2017 Version 0.75* (Vol. Version 0.).
- Caine, R., & Caine, G. (1991). *Making Connection: Teaching and the Brain*.
- Caine, R. N. R., & Caine, G. (1990). Understanding a brain-based approach to learning and teaching. *Educational Leadership*(October), 66–70.
- Chandrasekar, K., Cleary, G., Cox, O., & O Gorman, B. (2017). *Symantec* (Tech. Rep. No. April).
- Chi, H., Jones, E. L., & Brown, J. (2013). Teaching Secure Coding Practices to STEM Students. *Proceedings of the 2013 on InfoSecCD '13 Information Security Curriculum Development Conference - InfoSecCD '13*, 42–48.
- Christopher, L., Choo, K. K. R., & Dehghantanha, A. (2016). *Honeypots for Employee Information Security Awareness and Education Training: A Conceptual EASY Training Model*. Elsevier Inc.
- Chung, S., Hansel, L., Bai, Y., Moore, E., Taylor, C., Crosby, M., Heller, R., Popovsky, V., & Endicott-Popovsky, B. (2014). What approaches work best for teaching secure coding practices? *2014 HUIC Education & STEM Conference*.

- Clemons, S. A. (2005). Brain Based Learning : Possible Implications for Online Instruction. (September 2005), 1–10.
- Cohen, J. (2010). Lightweight Code Review Episode 6 : Checklists - You build me up just to knock me down.
- Commission, I. E., & Division, S. S. (2013). *SANS 27034-1 : 2013 SOUTH AFRICAN NATIONAL STANDARD Information technology — Security techniques — Application security Part 1 : Overview and concepts*.
- Conklin, A., & White, G. (2005). A Graduate Level Assessment Course : A Model for Safe Vulnerability Assessment. (June), 6–9.
- Craig, D. I. (2003). Brain-compatible learning: principles and applications in athletic training... including commentary by Blackley SB with author response. *Journal of Athletic Training*, 38, 342–350.
- Creswell, J. W. (2007). *Qualitative enquiry & research design, choosing among five approaches* (Vol. 2nd ed).
- Creswell, J. W. (2009). *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*.
- Cross, M. (2007). *Developer's Guide to Web Application Security*.
- Customs Solutions Group. (2012). A CISO 's Guide to Application Security.
- Deepa, G., & Thilagam, P. S. (2016). Securing web applications from injection and logic vulnerabilities: Approaches and challenges. *Information and Software Technology*, 74, 160–180.
- Dunlosky, J., Rawson, K. A., Marsh, E. J., Nathan, M. J., & Willingham, D. T. (2013). Improving students' learning with effective learning techniques: Promising directions from cognitive and educational psychology. *Psychological Science in the Public Interest, Supplement*, 14(1), 4–58.
- Fawcett, G., & Juliana, M. (2015). Teaching in the Digital Age. *Designing Instruction for Technology-Enhanced Learning*, 71–82.

- Finn, B., Thomas, R., & Rawson, K. A. (2018). Learning more from feedback: Elaborating feedback with examples enhances concept learning. *Learning and Instruction*, 54, 104–113.
- Florian, T. P., & Zimmerman, J. P. (2015). Understanding by Design , Moodle , and Blended Learning : A Secondary School Case Study. *MERLOT Journal of Online Learning and Teaching*, 11(1), 120–128.
- Fujitsu. (2006). The TCP / IP Protocol Suite. *Fujitsu*, 1–74.
- Garrison, D., & Vaughans, N. D. (2013). *Blended Learning in Higher Education* (Vol. 53).
- Gero, J. (1990). Design Prototypes: A Knowledge Representation Schema for Design. *AI Magazine*, 11(4), 26.
- Giesen, D., Meertens, V., Vis-visschers, R., & Beukenhorst, D. (2012). Questionnaire development 12 1. (201210), 1–82.
- Graham, C. R., & Dziuban, C. (2008). Blended Learning Environments. *Handbook of research on educational communications and technology* (3rd ed.)(February 2015), 269–276.
- Grassi, P. A., Garcia, M. E., & Fenton, J. L. (2017). Digital identity guidelines: revision 3.
- Holzmann, G. J. (2009). S CRUB : a tool for code reviews. 6(4), 311–318.
- Hudli, A. V., & Pidaparti, R. M. V. (1995). Distributed finite element structural analysis using the client server model. *Communications in Numerical Methods in Engineering*, 11(3), 227–233.
- Hugoson, M. Å. (2008). Centralized versus Decentralized Information Systems: A Historical Flashback. *IFIP Advances in Information and Communication Technology*, 303, 106–115.
- ISO/IEC. (2010). *SANS 23026 : 2010 SOUTH AFRICAN NATIONAL STANDARD Software Engineering — Recommended Practice for the Internet — Web Site Engineering , Web Site Management , and Web Site Life Cycle*.

- ISO/IEC. (2011). *SANS 1105-2 : 2011 SOUTH AFRICAN NATIONAL STANDARD Information and documentation — Open systems interconnection — Interlibrary loan application protocol specification Part 2 : Protocol implementation conformance statement (PICS) proforma.*
- ISO/IEC. (2013). *Sans 27002 2013.*
- Jeff, S. (2017). *Desktop Applications Vs. Web Applications.* (<http://www.streetdirectory.com/travelguide/114448/programming/desktopapplicationsvswebapplications.html>, Last Accessed 2017-08-28)
- Jensen, E. (2000). Brain-based learning: A reality check. *Educational Leadership*, 57(7), 76–80.
- Jiménez, J. A., Merodio, J. A. M., & Sanz, L. F. (2017). Checklists for compliance to DO-178C and DO-278A standards. *Computer Standards and Interfaces*, 52(January), 41–50.
- Keil, M., Li, L., Mathiassen, L., & Zheng, G. (2008). The influence of checklists and roles on software practitioner risk perception and decision-making. *Journal of Systems and Software*, 81(6), 908–919.
- Landoll, D. J. (2006). *The Security Risk Assessment Handbook: A Complete Guide for Performing Security Risk Assessments.*
- Lee, G. (2014). *Cloud Networking : Developing Cloud-Based Data Center Networks* (Vol. 33).
- Li, X. (2013). A Survey on Server-side Approaches to Securing Web Applications. *V*(November), 1–30.
- Li, Y., Li, D., Cui, W., & Zhang, R. (2011). Research based on OSI model. *2011 IEEE 3rd International Conference on Communication Software and Networks, ICCSN 2011*, 554–557.
- Lunt, B., Ekstrom, J., Gorka, S., Hislop, G., Kamali, R., Lawson, E., LeBlanc, R., Miller, J., & Reichgelt, H. (2008). Curriculum Guidelines for Undergraduate Degree Programs in Information Technology. 139.

- Lunt, B., Sabin, M., Hala, A., Impagliazzo, J., & Zhang, M. (2017). Information Technology Curricula 2017. 1–92.
- Macdonald, S., & Headlam, N. (2009). Research Methods Handbook. 72.
- Mack, N., Woodsong, C., M. MacQueen, K., Guest, G., & Namey, E. (2011). Qualitative Research Methods Overview. *Family Health International*(January), 1–12.
- Mafuwane, B. (2003). Research Design and Methodology. In (pp. 152–168).
- Mansour, H. F. (2015). Enhancing first year management students' engagement: An action research project to explore the use of the Essay Feedback Checklist (EFC). *International Journal of Management Education*, 13(3), 218–226.
- Milne, J. (1994). Questionnaires : Some Advantages and Disadvantages. *Centre for CBL in Land Use and Environmental Sciences*, 5248.
- Moriarty, J. (2011). Methods Review 1. 1–43.
- Nicol, D. J., & Macfarlane-Dick, D. (2006). Formative assessment and self-regulated learning : A model and seven principles of good feedback practice . Formative assessment and self-regulated learning : A model and seven principles of good feedback practice . *Studies in Higher Education (2006)*,, 31(2), 199–218.
- Okaz, A. A. (2015). Integrating Blended Learning in Higher Education. *Procedia - Social and Behavioral Sciences*, 186(May), 600–603.
- OWASP. (2017). *The OWASP Foundation*. (https://www.owasp.org/index.php/Main{_}Page, last accessed on 2017-08-08)
- Panadero, E., Jonsson, A., & Botella, J. (2017). Effects of self-assessment on self-regulated learning and self-efficacy: Four meta-analyses. *Educational Research Review*, 22, 74–98.
- Ravali, P. (2013). A Comparative Evaluation of OSI and TCP/IP Models. *International Journal of Science and Research (IJSR) ISSN (Online Index Copernicus Value Impact Factor*, 14(7), 2319–7064.

- Razzaq, A., Latif, K., Ahmad, H. F., Hur, A., Anwar, Z., & Bloodsworth, P. C. (2014). Semantic security against web application attacks. *Information Sciences*, 254, 19–38.
- Reid, R., & Van Niekerk, J. (2012). A comparative analysis of HTML 5 and Silverlight for the implementation of brain-compatible education in web-based learning environments. (September), 1–15.
- Reid, R., Van Niekerk, J., & Von Solms, R. (2011). Guidelines for the creation of brain-compatible cyber security educational material in Moodle 2.0. *2011 Information Security for South Africa - Proceedings of the ISSA 2011 Conference*.
- Rouse, M. (2018). *Confidentiality, Integrity, and Availability (CIA triad)*. (<https://whatistechtarget.com/definition/confidentiality-integrity-and-availability-CIA>, Last Accessed 2018-04-06)
- Saunders, M., Lewis, P., & Thornhill, A. (2009). *Research methods for business students*.
- Shackelford, R., Cross, J., Davies, G., Impagliazzo, J., & Kamali, R. (2005). *Computing Curricula 2005* (Vol. 1).
- Smartbear. (2009). Peer Code Review : An Agile Process. *Review Literature And Arts Of The Americas*(November), 1–6.
- Torrise, G., & Davis, G. (2000). Online Learning as a catalyst for reshaping practice – The experiences of some academics developing online learning materials. *International Journal for Academic Development*, 5(2), 166–176.
- Tufekci, S., & Demirel, M. (2009). The effect of brain based learning on achievement, retention, attitude and learning process. *Procedia - Social and Behavioral Sciences*, 1(1), 1782–1791.
- University Nelson Mandela. (2018). FACULTY OF ENGINEERING, THE BUILT ENVIRONMENT AND INFORMATION TECHNOLOGY PROSPECTUS. *Ieee*, 2(7), 1–8.

- Van Niekerk, J., & Webb, P. (2016). The effectiveness of brain-compatible blended learning material in the teaching of programming logic. *Computers and Education*, 103, 16–27.
- Van Niekerk, J. F., & Von Solms, R. (2010). Information security culture: A management perspective. *Computers and Security*, 29(4), 476–486.
- Van Voorn, G. A., Verburg, R. W., Kunseler, E. M., Vader, J., & Janssen, P. H. (2016). A checklist for model credibility, salience, and legitimacy to improve information transfer in environmental policy assessments. *Environmental Modelling and Software*, 83, 224–236.
- Von Solms, R., & Van Niekerk, J. (2013). From information security to cyber security. *Computers and Security*, 38, 97–102.
- Vroom, C., & Von Solms, R. (2004). Towards information security behavioural compliance. *Computers and Security*, 23(3), 191–198.
- Wagner, B., & Wenzel, M. (2015). *C# Coding Conventions (C# Programming Guide)*.
- Walliman, N. (2011). *Research methods*. Routledge.
- Wilson, C. (2013). *Credible Checklists and Quality Questionnaires: A User-Centered Design Method*.
- Zhu, J., Xie, J., Lipford, H. R., & Chu, B. (2014). Supporting secure programming in web applications through interactive static analysis. *Journal of Advanced Research*, 5(4), 449–462.

Part I

Appendices

Appendix A

Academic Publication

Secure Coding Practices in the Software Development Capstone Project

V.S. Mdunyelwa, J.F. van Niekerk and L.A. Fletcher

Centre for Research in Information and Cyber Security, Nelson Mandela University,
Port Elizabeth, South Africa
e-mail: {s211117803, Johan.VanNiekerk, Lynn.Fletcher}@nmmu.ac.za

Abstract

Web applications play an important role in many organisations, but could also expose these organisations to cyber security risks. Organisations use a variety of cyber security controls to mitigate risks. Currently, most organisational security spending focus on reducing network security related risks. However, most attacks focuses on vulnerabilities existing at the web application layer. Security breaches in web applications are mostly caused by programmers' failure to adhere to secure coding practices, such as those recommended by the Open Web Application Security Project. The purpose of this paper is to determine whether software development students know about secure coding practices and whether they implement them when developing web applications as part of their capstone projects.

Keywords

Secure Coding, Web Application Security, OWASP, Capstone Project, Knowledge, Behaviour

1. Introduction

Nowadays many organisations use information to achieve their daily activities. Both the private and public sectors rely on information systems to carry out their key operational activities. Previously, many software applications were desktop applications which could be used on standalone computers. The arrival of the internet has seen web application development gain a significant amount of attention. Media players and word processors are typical examples of desktop applications, whilst internet banking and online shopping carts on e-commerce websites can be considered as web applications (Jeff 2017).

Desktop applications are usually installed on a single computer for a specific task, whereas web applications are made available on servers (Jeff 2017). Thus, web applications are exposed to a number of threats on the internet that desktop applications may not be exposed to. In addition, web applications often handle very sensitive data, used for carrying out critical tasks such as banking, online shopping and online tax filing (Deepa & Thilagam 2016). These applications are trusted by billions of users for performing such daily activities.

Web applications have received attention from both academia and industry to initiate some defence mechanisms to protect them from security threats (Deepa & Thilagam 2016). Handling risks related to the security of web applications is a major challenge for many organisations. To this effort, information security has become one of the top managerial priorities in many organisations (Benbasat 2010). The need for securing organisation's network resources has increased over the past decade. Many organisations address this need by including costs for firewalls and other network security controls in their budgets. However, 75% of all attacks that are on the internet are executed through the application layer of the OSI Model (Customs Solutions Group 2012). This indicates that more focus needs to be put on securing web applications.

More than 75% of web applications have vulnerabilities (Chandrasekar et al. 2017). Many of these web applications have common vulnerabilities which can be easily corrected (Zhu et al. 2014). Addressing many of these web application vulnerabilities is relatively straightforward through introducing secure coding practices. Academic institutions may teach proper programming, but because of the limited resources they only focus on the correctness of code and end up overlooking security in the code (Bishop & Orvis 2006). This oversight in addressing security issues is felt most during the maintenance phase of the Software Development Life Cycle, which is costly (Customs Solutions Group 2012). Therefore, if academic institutions would integrate secure coding practices in undergraduate software development courses, students would be more likely to implement them and in turn help improve security in web applications in industry.

Human aspects in information and cyber security usually consists of two major elements, namely knowledge-based and behavioural elements. Knowledge deals with the underlying requisite knowledge that would allow a person to behave in accordance to security guidelines, whilst the behavioural elements deal with everything else which if one can assume requisite knowledge, would actually make them apply it (Van Niekerk & Von Solms 2010). It is possible for a software development student to have the requisite knowledge and not behave appropriately, but it is unlikely for someone to behave appropriately if they do not have the requisite knowledge.

The purpose of this paper is firstly to demonstrate that software development students generally do not adhere to secure coding practices when performing their capstone projects. Secondly, it tries to determine whether the lack of adherence is caused by purely behavioural aspects or an underlying lack of knowledge. It therefore focuses on both knowledge and behaviour relating to secure coding practices. The next section discusses secure coding practices according to the Open Web Application Security Project (OWASP) guidelines. It further discusses what the Association for Computing Machinery (ACM) curricular guidelines for Information Technology states in terms of secure coding practices in Section 3. Section 4 presents the sample and setting, while Section 5 addresses the behavioural aspects of students and Section 6 focuses on the knowledge of students. Future work and conclusion follows in Section 7 and 8 respectively.

2. Secure coding practices

The secure processing of information consists of the implementation of various controls in order to reduce risks caused by a specific security vulnerability (von Solms & van Niekerk 2013). In the case of web applications there are various types of controls that could play a role in protecting information. The types of controls are:

- Physical controls, for example, providing a lock on the computing facility.
- Technical controls, for example, an authentication mechanism.
- Operational controls deal with human behaviour, for example, having to use the lock to prevent access to the computing facility or requiring authentication before allowing access to computing resources.

As demonstrated above, if these control types are not collectively addressed, it could lead to security breaches. Some of the controls may deal with end-user behaviour, while others deal with the behaviour of Information Technology staff in relation to configuring technical controls (Chung et al. 2014). This includes programming staff who develop web-based applications and who should adhere to secure coding practices to mitigate risks associated with web application security. While the technical control would be a secure coding practice, an operational control would be a policy and procedures for a programmer to adhere to secure coding practices.

In the case of web application security, one of the best sources for securing web applications is the Open Web Application Security Project (OWASP) (owasp.org). OWASP is a non-profit organisation with a goal to improve the security of web applications. This organisation focuses on making security visible, so that individuals and organisations such as governments, companies and educational institutions have more knowledge when developing web applications (OWASP 2017).

OWASP focuses on many development security platforms. This research study uses OWASP security practices which apply to the Microsoft .NET framework. There are several ways in which programmers could introduce security flaws when using the .NET framework especially when working at the data access layer.

The controls OWASP proposes are technical measures, while the programmers themselves having to adhere to them would be an operational measure. Operational controls fail when the human does not adhere to them. Humans will often not adhere to such a control due to the lack of knowledge or because there is no procedural requirement forcing them to adhere to the control. Therefore, programmers should be educated on secure coding practices to ensure that they are aware of the vulnerabilities that may otherwise be missed or realised at a stage where it might be too late. This behaviour should also be audited to ensure that programmers adhere to such secure coding practices.

Table 1 presents a sample set of secure coding practices extracted from the OWASP guidelines. Those selected are specifically for the Windows environment used in the .NET framework. In addition, these secure coding practices were chosen because

they are specific to the data access layer where attackers could easily access information if it is not secure.

SP	OWASP secure coding practices
SP1	Use Parameterised SQL commands for all data access, without exception.
SP2	Do not use SQL command with string parameter made up of a concatenated SQL string.
SP3	Whitelist allowable values coming from the user. Use Enums, TryParse or lookups to ensure that the data coming from the user is as expected.
SP4	Apply the Principle of Least Privilege when setting up the Database of your choice.
SP5	When using SQL Server, prefer integrated authentication over SQL authentication.
SP6	Using stored procedures is the most effective way to counter the SQL injection vulnerability.
SP7	Encrypt sensitive data in the database including connection strings.
SP8	Connection strings should be based in a configuration file.
SP9	Never write your own encryption.

Table 1: OWASP secure coding practices

Since this research focuses on the knowledge and behaviour of software development students regarding secure coding practices, it is important to consider curricular guidelines. These are discussed in the next section.

3. Curricular guidelines within the software development curriculum.

Traditionally, most software developers are educated in tertiary institutions. Tertiary institutions follow various curricula to determine what to teach. According to the ACM curricular guidelines for Information Technology (Lunt et al. 2017), the focus of a software development curriculum is to educate students with programming and database fundamentals. During the students' final year it is important for them to consolidate and integrate all the skills gained throughout the qualification. Further, the ACM curricular guidelines state that security should be integrated into the software development life cycle throughout the curriculum. This means that secure coding practices should be taught from the early stages in the curriculum.

Capstone experiences are projects that are performed during the final year of a qualification, mostly in engineering qualifications (Lunt et al. 2017). However, the capstone experience has gained a lot of support in the computing discipline. Capstone projects normally come with 3 major elements, namely; 1) working in teams of 3 to 5; 2) choosing a real-world problem to solve using the project and the project must be integrative and students who are engaged in this experience should have completed most of the curriculum before attempting this project (Lunt et al. 2017). Capstone projects are therefore the best place to determine whether secure coding practices are being taught and applied during the software development life cycle.

4. Sample and setting

This study was conducted within a comprehensive tertiary institution in South Africa. Comprehensive institutions offer both academic degrees as well as vocational qualifications. In this case, the sample was drawn from students registered for the National Diploma: Software Development. The National Diploma: Software Development is a vocational qualification focusing on providing prospective software developers with the requisite skills for professional work. Students from this program were selected as a research sample for this study. The sample was selected both *purposively* and because it was *convenient*.

They were selected *purposively* in the sense that these students were selected specifically because the researchers believe that they should have requisite web application development skills. They are a *convenient* sample because the researchers have access to these specific students. This research project received ethics approval from the university research ethics committee (Ref H15-ENG-ITE-009).

The Software Development National Diploma focuses on software development skills and therefore has a large portion of the curriculum dedicated to the development of web applications. Subjects include Internet Programming and Development Software as part of the curriculum. The department where the students reside is well known for research in information security. It should therefore be reasonable to assume that these students have some form of secure coding knowledge. To a certain extent this qualification has also been modelled according to the ACM curricular guidelines for Information Technology (Lunt et al. 2017). In the National Diploma: Software Development, final year students are expected to develop a capstone project as described in the ACM curriculum (Lunt et al. 2017).

This capstone project is a software development orientated project which consolidates and integrates all the skills gained throughout the software development qualification. The researchers therefore believe that this capstone project is an ideal place to determine whether students adhere to the secure coding practices for the development of secure web applications as proposed by OWASP.

Students at the university within this qualification are trained specifically to develop software in a Windows environment using the .NET framework. The students therefore, as a minimum, if they were taught the appropriate secure web application principles should be aware of the OWASP guidelines for the .NET framework.

5. Behaviour regarding secure coding practices

The first phase in this research sets out to determine whether students doing their capstone project actually adhere to the secure coding practices as recommended by OWASP. In order to determine the degree of behavioural compliance, the researchers assume that the students would already have all the requisite knowledge regarding secure coding practices. Therefore, this research could focus only on

whether or not completed capstone project implemented these practices. Since the capstone projects are carried out after all formal programming education subjects have been completed in the curriculum, this assumption was deemed appropriate. For this purpose, the researchers focuses specifically on completed projects from a prior year.

5.1. Data generating instruments for behaviour

The questions in Table 2 relate to the OWASP secure practices (SP1 to SP9) previously illustrated in Table 1. These questions were used to create an instrument in the form of a checklist that was used to perform a code review for final year capstone projects submitted during the 2015 period. All web-based application projects were included in this code review. The data used in this study was specifically for 2015, mainly because data from 2017 is incomplete and will only be available at the end of the year. 2016 proved to be an unfavourable year for South African university students, since there were wide spread student protests during the year which disrupted classes and student projects. From the 2015 data, fifteen projects were included in this study and every web form was reviewed and all data was recorded on the instrument as shown in Table 2. The code review was conducted to determine whether students' adhere to secure coding practices when developing their capstone projects.

SP	Questions asked based on OWASP practices
SP1	Are they making use of parameterised SQL commands for all data access?
SP2	Do they make use of concatenated strings in the queries?
SP3	Are all input fields validated?
SP4	Do they make use of the Principle of Least Privilege when setting up their databases?
SP5	Do they use integrated authentication or do they use SQL authentication?
SP6	Do they use stored procedure for their queries?
SP7	Are they encrypting their connection strings?
SP8	Does the connection string only appear once in the web.config file?
SP9	Is all the sensitive data being encrypted using the OWASP recommended methods?

Table 2: Checklist to use for code review

The code review was conducted by the primary researcher using the checklist provided in Table 2. All results were captured in a database for ease of analysis.

5.2. Data collection and analysis

Table 3 shows the results of the code review for each project (P1 to P15) and every secure coding practice (SP1 to SP9). Each project consisted of a number of webforms. The percentage to which a project adhered to a secure coding practice was calculated as the ratio between the number of webforms that should have used the practice and those that actually did use the practice. For example, Project P12 had 19 forms and only 9 of those accessed the database, of those that accessed the database,

only 3 forms used stored procedures as in SP6, therefore the percentage would calculate as $3/9 \times 100 = 33\%$.

Proj#	SP1	SP2	SP3	SP4	SP5	SP6	SP7	SP8	SP9
P1	100%	100%	100%	100%	0%	85%	0%	100%	28%
P2	84%	100%	100%	100%	0%	100%	0%	65%	15%
P3	100%	100%	97%	100%	0%	27%	0%	42%	15%
P4	100%	100%	100%	100%	0%	0%	0%	100%	12%
P5	50%	50%	6%	100%	0%	50%	0%	0%	40%
P6	90%	80%	70%	81%	0%	13%	0%	81%	60%
P7	93%	93%	100%	92%	0%	67%	0%	57%	40%
P8	100%	100%	100%	0%	0%	27%	0%	100%	0%
P9	100%	100%	100%	55%	0%	45%	0%	100%	12%
P10	100%	100%	100%	5%	0%	40%	0%	100%	5%
P11	100%	100%	85%	0%	0%	57%	0%	100%	40%
P12	85%	85%	100%	0%	100%	33%	0%	87%	57%
P13	73%	70%	40%	0%	100%	13%	0%	53%	26%
P14	12%	27%	0%	9%	100%	0%	0%	36%	40%
P15	40%	57%	14%	92%	100%	0%	0%	35%	38%
Avg.	82%	84%	74%	56%	27%	37%	0%	70%	29%

Table 3: Adherence to secure coding practices

As shown in Table 3 above, there was generally a lack of adherence to secure coding practices. Where adherence did occur it was often inconsistent. For example, SP5 was about using Windows authentication. 11 of the projects did not use Windows authentication, however 4 of them did use the desired authentication model. This clearly indicates that some of them knew how to do this. This is probably a behavioural issue caused by a lack of knowledge as to **why** they should choose one authentication model above the other. As another example, SP7 deals with the encryption of a connection string. None of the projects encrypted their connection strings. However, it was determined that the capstone project requirement is that they should not encrypt the connection strings because values used in the connection string are used as part of the project evaluation. In this case, additional measures should be taken to make sure that students do know how to encrypt these connection strings. SP9 deals with the Principle of Least Privilege. Students are required to apply this principle. The fact that it is partly adhered to it shows that they either cannot use it or they do not know why they should use it. Therefore, the lack of behavioural compliance might in fact be caused by an underlying lack of knowledge. The next phase of the research focused on determining whether current capstone students have this requisite knowledge.

6. Knowledge regarding secure coding practices

This phase of the research set out to determine whether students doing their capstone projects have the requisite knowledge regarding secure coding practices as recommended by OWASP.

6.1. Data generating instruments for knowledge

The data generating instrument for determining the students' knowledge regarding secure coding practices took the form of a questionnaire. The questionnaire was structured according to the OWASP secure coding practices as shown in Table 1. Questions for each SP were constructed to determine whether students would be able to apply the requisite knowledge. Due to space limitations, the full set of questions cannot be depicted in their entirety, as many of these questions contained code snippets and schematic representations as part of phrasing of the question. Figure 1 shows an example of a question asked to determine students' knowledge.

Question 3 Which ONE of the following SQL statements would be the most vulnerable to SQL injection? (1)

☐ A. SELECT UserID, Password FROM Users WHERE UserID = @UserID AND Password = @Password

☐ B. SELECT * FROM Users WHERE UserID = txtUserID AND Password = txtPassword

☐ C. All of the above.

☐ D. C. SELECT Password, UserID FROM Users

Figure 1: Example question for knowledge questionnaire

The questions were multiple choice questions where students were required to select the answer that best answered the question. These results were collected and analysed as discussed in the next section.

6.2. Data collection and analysis

In the case of secure coding practices, students should at the very least be able to apply the requisite knowledge. The questions were thus structured appropriately. The questionnaire was administered during a 2017 project lecture where 86 students were present. The lecturer for the capstone project was not present. The researchers briefly provided students with the context of the study before distributing the paper-based questionnaire for completion.

The questionnaire was marked by an automated computer marking system. Table 4 shows the consolidated results of the questions for each SP by showing the percentage of correct answers, and the averages from the behavioural instrument.

SP	SP1	SP2	SP3	SP4	SP5	SP6	SP7	SP8	SP9
K	66%	36%	30%	36%	17%	21%	11%	3%	1%
B	82%	84%	74%	56%	27%	37%	0%	70%	29%

Table 4: Underlying knowledge related to the ability to apply secure coding practices

If one considers the results of both parts of the study collectively as demonstrated in Table 4, in some cases students do have both the behavioural and knowledge aspects. However, in most cases they performed poorly. For example, in SP4, Principle of

Least Privilege, some students did not adhere to this practice, and this is important in terms of web application security. From SP4, it is evident that students also did not have all the requisite knowledge as to how to use this and/or why the Principle of Least Privilege should be used. Another interesting observation is when looking at SP1, students generally behaved well and only a few did not comply with this secure practice. In many cases, the level of Behavioural adherence is higher than the actual level of the underlying security knowledge. This could quite possibly be an indication that lectures are teaching students the appropriate ways of using the code without linking it to the underlying security context. This still poses a long-term risk because if the programmers do not understand why they are doing something, they might still disregard it in future implementations. Also by looking at SP1, these students generally have the knowledge about the use of SQL commands but do not perform accordingly because they possibly do not know why they should adhere to secure practices. Therefore, it can be seen that these students lack both behavioural compliance and the requisite knowledge.

7. Future work

Future work will focus on both the knowledge and the behavioural aspects of compliance to secure coding best practices. In order to address the lack of knowledge, the primary researcher will create an educational intervention in the form of a blended learning program. It is the intention that students who complete this educational intervention will be tested again using the same instrument as used during this research to provide the researcher with pre and post test data.

Addressing the lack of underlying knowledge might have an impact on behavioural compliance. However, in order to further encourage behavioural compliance an instrument in the form of a checklist will form part of formal project documentation for future capstone project students.

8. Conclusion

We live in a world where information is important. Therefore, none adherence to secure coding practices by developers puts information at risk. These programmers cannot adhere if they are not educated regarding the secure coding best practices in the first place. This research has shown that such a lack in knowledge exists among current software development students at a specific South African university. More attention therefore needs to be placed on educating students in this regard during the course of their studies.

9. Acknowledgements

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

10. References

- Benbasat, I., 2010. SPECIAL ISSUE INFORMATION SECURITY POLICY COMPLIANCE: AN EMPIRICAL STUDY OF RATIONALITY-BASED BELIEFS. , 34(3), pp.523–548.
- Bishop, M. & Orvis, B.J., 2006. A Clinic to Teach Good Programming Practices. *Proceedings of the 10th Colloquium for Information Systems Security Education*, pp.168–174. Available at: <http://nob.cs.ucdavis.edu/~bishop/papers/2006-cisse-2/clinic.pdf>.
- Chandrasekar, K., Cleary, G., Cox, O. and O Gorman, B., 2017. *Internet Security Threat Report*, Available at: <https://www.symantec.com/security-center/threat-report>.
- Chung, S., Hansel, L., Bai, Y., Moore, E., Taylor, C., Crosby, M., Heller, R., Popovsky, V. and Endicott-Popovsky, B., 2014. What approaches work best for teaching secure coding practices? *2014 HUIC Education & STEM Conference*.
- Customs Solutions Group, 2012. A CISO 's Guide to Application Security. Available at: <http://h30528.www3.hp.com/Security/CISOGuideToApplicationSecurity.pdf>.
- Deepa, G. & Thilagam, P.S., 2016. Securing web applications from injection and logic vulnerabilities: Approaches and challenges. *Information and Software Technology*, 74, pp.160–180. Available at: <http://dx.doi.org/10.1016/j.infsof.2016.02.005>.
- ISO/IEC, 2013. Sans 27002 2013.
- Jeff, S., 2017. No Title. Available at: http://www.streetdirectory.com/travel_guide/114448/programming/desktop_applications_vs_web_applications.html [Accessed August 28, 2017].
- Lunt, B., Sabin, M., Hala, A., Impagliazzo, J. and Zhang, M., 2017. Information Technology Curricula 2017. , pp.1–92.
- Van Niekerk, J.F. & Von Solms, R., 2010. Information security culture: A management perspective. *Computers and Security*, 29(4), pp.476–486. Available at: <http://dx.doi.org/10.1016/j.cose.2009.10.005>.
- OWASP, 2017. No Title. Available at: https://www.owasp.org/index.php/Main_Page [Accessed August 8, 2017].
- von Solms, R. & van Niekerk, J., 2013. From information security to cyber security. *Computers and Security*, 38, pp.97–102. Available at: <http://www.sciencedirect.com.ezproxy.unisabana.edu.co/science/article/pii/S0167404813000801>.
- Zhu, J., Xie, J., Lipford, H. & Chu, B., 2014. Supporting secure programming in web applications through interactive static analysis. *Journal of Advanced Research*, 5(4), pp.449–462. Available at: <http://dx.doi.org/10.1016/j.jare.2013.11.006>.

Appendix B

Questionnaire (Pre-Test)

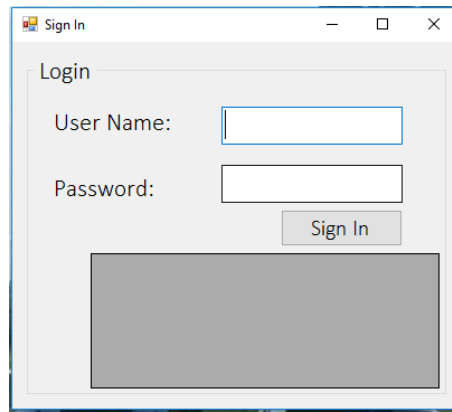


Figure 1: Sign In Simulation

QCM

TEST

OWASP pre-test*10 Minutes*

- Answer all questions on the answer sheet provided at the back of this paper.
- All questions that use the sign ♣ may have zero, one or several correct answers.
- Other questions have only a single correct answer
- Incorrect answers might be penalized so do not guess!
- No cellphone or other mobile device can be used during this session.
- This question sheet **MUST** be handed in with your answer sheet

Question 1 ♣ Use Figure 1 to answer the following question:

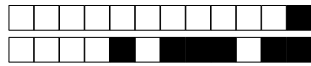
User Input : **Name' OR 1=1; DROP TABLE USERS;**–

What would happen if the user enters the above input in the 'USER NAME' input field? (3)

- ☐ A User will be logged on to the system and the **User** table will be deleted.
- ☐ B Nothing will happen.
- ☐ C The system will log the user on.
- ☐ D User would be required to enter a password first.
- ☐ E None of these answers are correct.

Question 2 Which of the following SQL statements would be the best to use in the Form shown in Figure 1? (1)

- ☐ A A. SELECT UserID, Password FROM Users WHERE UserID = @UserID AND Password = @Password
- ☐ B D. All of the above.
- ☐ C B. SELECT * FROM Users WHERE UserID = txtUserID AND Password = txtPassword
- ☐ D C. SELECT Password, UserID FROM Users



Question 3 Which ONE of the following SQL statements would be the most vulnerable to SQL injection? (1)

- ☐ A. SELECT UserID, Password FROM Users WHERE UserID = @UserID AND Password = @Password
- ☐ B. SELECT * FROM Users WHERE UserID = txtUserID AND Password = txtPassword
- ☐ C. D. All of the above.
- ☐ D. C. SELECT Password, UserID FROM Users

Question 4 Parameterized SQL Commands are a preferred way of coding data access? (1)

- ☐ A. No
- ☐ B. Yes

Question 5 ♣ When using SQL in a form, which of the following are acceptable ways for dealing with placeholders in the SQL statements?

- ☐ A. Stored Procedures
- ☐ B. Parameterized SQL statements.
- ☐ C. Concatenated SQL Strings.
- ☐ D. Use Try-Catch and concatenated SQL strings.
- ☐ E. *None of these answers are correct.*

Question 6 Which of the following methods of coding is recommended as the most appropriate for data access? (1)

- ☐ A. Parameterized SQL statements.
- ☐ B. Stored Procedures
- ☐ C. Use Try-Catch and concatenated SQL strings.
- ☐ D. Concatenated SQL Strings.

Question 7 ♣ Which of the following are valid approaches to authentication when connecting to a SQL database.(2)

- ☐ A. Login.
- ☐ B. Integrated Authentication (Windows Authentication).
- ☐ C. Administrator and User login.
- ☐ D. Mixed Authentication (SQL Authentication).
- ☐ E. Portion Authentication.
- ☐ F. *None of these answers are correct.*

Question 8 Which is the most preferred method of Authentication? (1)

- ☐ A. Portion Authentication.
- ☐ B. Mixed Authentication (SQL Authentication).
- ☐ C. Integrated Authentication (Windows Authentication).
- ☐ D. Administrator and User login.
- ☐ E. Login.



Question 9 Why is the method Select in Question X the best method to authenticate? (1)

- ☐ A Administrator and User login.
- ☐ B Requires no password because Windows takes care of security issues.
- ☐ C Secure with a username and password.
- ☐ D Easiest method of all the methods.
- ☐ E Secure with an encrypted username and password.

Question 10 When sensitive data has been encrypted, only the authorized parties can analyze the data. (1)

- ☐ A False
- ☐ B True

Question 11 ♣ Which of the following data should always be encrypted? (2)

- ☐ A Sensitive parts of the web.config file.
- ☐ B SQL statements.
- ☐ C Application Code
- ☐ D Sensitive information such as ID numbers only.
- ☐ E Connection Strings
- ☐ F Passwords
- ☐ G *None of these answers are correct.*

Question 12 ♣ Which are the recommended ways of encryption? (1)

- ☐ A Using a hashing algorithm.
- ☐ B Windows Data Protection (DPAPI).
- ☐ C 64-bit Algorithm for best encryption.
- ☐ D Writing your own encryption using an algorithm with random numbers and alphabets.
- ☐ E ASP.NET RegIIS.
- ☐ F *None of these answers are correct.*

Question 13 Where does the **Principle of Least Privilege** apply? (1)

- ☐ A Login.
- ☐ B 64-bit Algorithm for best encryption.
- ☐ C In the code behind for assigning rights.
- ☐ D Admin and User rights of the system.

Question 14 Why is the **Principle of Least Privilege** seen as important in application design? (1)

- ☐ A For security reasons.
- ☐ B To give certain rights to certain people and security reasons.
- ☐ C For user be able to complete their tasks in time.
- ☐ D For users of the system to to be guided on every action the do on the system.



Question 15 How could the **Principle of Least Privilege** be applied to employees? (1)

- ☐ **A** Give rights to the manager of a department and allow the manager to deal with logging in of users.
- ☐ **B** Giving employees rights according to the departments they belong to.
- ☐ **C** Having a person looking at them whilst the sign in to the system.
- ☐ **D** Create a password for users depending on a department.

DRAFT



<input type="checkbox"/> 0	<input type="checkbox"/> 0	<input type="checkbox"/> 0	<input type="checkbox"/> 0	<input type="checkbox"/> 0	<input type="checkbox"/> 0	<input type="checkbox"/> 0	<input type="checkbox"/> 0	<input type="checkbox"/> 0	<input type="checkbox"/> 0
<input type="checkbox"/> 1	<input type="checkbox"/> 1	<input type="checkbox"/> 1	<input type="checkbox"/> 1	<input type="checkbox"/> 1	<input type="checkbox"/> 1	<input type="checkbox"/> 1	<input type="checkbox"/> 1	<input type="checkbox"/> 1	<input type="checkbox"/> 1
<input type="checkbox"/> 2	<input type="checkbox"/> 2	<input type="checkbox"/> 2	<input type="checkbox"/> 2	<input type="checkbox"/> 2	<input type="checkbox"/> 2	<input type="checkbox"/> 2	<input type="checkbox"/> 2	<input type="checkbox"/> 2	<input type="checkbox"/> 2
<input type="checkbox"/> 3	<input type="checkbox"/> 3	<input type="checkbox"/> 3	<input type="checkbox"/> 3	<input type="checkbox"/> 3	<input type="checkbox"/> 3	<input type="checkbox"/> 3	<input type="checkbox"/> 3	<input type="checkbox"/> 3	<input type="checkbox"/> 3
<input type="checkbox"/> 4	<input type="checkbox"/> 4	<input type="checkbox"/> 4	<input type="checkbox"/> 4	<input type="checkbox"/> 4	<input type="checkbox"/> 4	<input type="checkbox"/> 4	<input type="checkbox"/> 4	<input type="checkbox"/> 4	<input type="checkbox"/> 4
<input type="checkbox"/> 5	<input type="checkbox"/> 5	<input type="checkbox"/> 5	<input type="checkbox"/> 5	<input type="checkbox"/> 5	<input type="checkbox"/> 5	<input type="checkbox"/> 5	<input type="checkbox"/> 5	<input type="checkbox"/> 5	<input type="checkbox"/> 5
<input type="checkbox"/> 6	<input type="checkbox"/> 6	<input type="checkbox"/> 6	<input type="checkbox"/> 6	<input type="checkbox"/> 6	<input type="checkbox"/> 6	<input type="checkbox"/> 6	<input type="checkbox"/> 6	<input type="checkbox"/> 6	<input type="checkbox"/> 6
<input type="checkbox"/> 7	<input type="checkbox"/> 7	<input type="checkbox"/> 7	<input type="checkbox"/> 7	<input type="checkbox"/> 7	<input type="checkbox"/> 7	<input type="checkbox"/> 7	<input type="checkbox"/> 7	<input type="checkbox"/> 7	<input type="checkbox"/> 7
<input type="checkbox"/> 8	<input type="checkbox"/> 8	<input type="checkbox"/> 8	<input type="checkbox"/> 8	<input type="checkbox"/> 8	<input type="checkbox"/> 8	<input type="checkbox"/> 8	<input type="checkbox"/> 8	<input type="checkbox"/> 8	<input type="checkbox"/> 8
<input type="checkbox"/> 9	<input type="checkbox"/> 9	<input type="checkbox"/> 9	<input type="checkbox"/> 9	<input type="checkbox"/> 9	<input type="checkbox"/> 9	<input type="checkbox"/> 9	<input type="checkbox"/> 9	<input type="checkbox"/> 9	<input type="checkbox"/> 9

← please encode
your student number below, and write
your first and last names below.

Firstname and lastname:

Answer sheet:

Answers must be given exclusively on this sheet: answers given on the other sheets will be ignored.

- QUESTION 1: ☐A ☐B ☐C ☐D ☐E
- QUESTION 2: ☐A ☐B ☐C ☐D
- QUESTION 3: ☐A ☐B ☐C ☐D
- QUESTION 4: ☐A ☐B
- QUESTION 5: ☐A ☐B ☐C ☐D ☐E
- QUESTION 6: ☐A ☐B ☐C ☐D
- QUESTION 7: ☐A ☐B ☐C ☐D ☐E ☐F
- QUESTION 8: ☐A ☐B ☐C ☐D ☐E
- QUESTION 9: ☐A ☐B ☐C ☐D ☐E
- QUESTION 10: ☐A ☐B
- QUESTION 11: ☐A ☐B ☐C ☐D ☐E ☐F ☐G
- QUESTION 12: ☐A ☐B ☐C ☐D ☐E ☐F
- QUESTION 13: ☐A ☐B ☐C ☐D
- QUESTION 14: ☐A ☐B ☐C ☐D
- QUESTION 15: ☐A ☐B ☐C ☐D

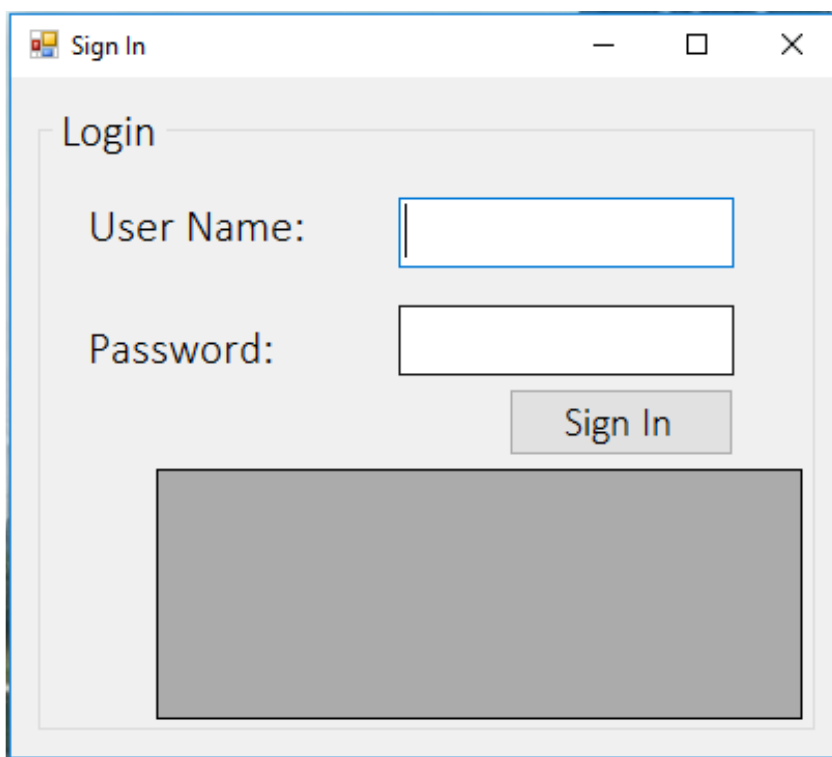
Appendix C

Questionnaire (Post-Test)

1. What are the two types of validating input?

- a. Whitelisting
- b. Blacklisting
- c. If statements
- d. Java API validation
- e. All of the above.

Answer: _____



2.

Use the picture to answer the following question:

User Input: Name' OR 1=1; DROP TABLE USERS;--

What would happen if the user enters the above input in the 'User Name' input field?

- a. User will be logged on to the system and the User table will be deleted.
- b. Nothing will happen.

- c. User would be required to enter a password first.
- d. The system will log the user on.

Answer: _____

3.

Which of the following SQL commands would be best to use in form shown the picture?

a.

```
SELECT UserID, Password FROM Users WHERE UserID = @UserID  
AND Password = @Password
```

b.

```
SELECT * FROM Users WHERE UserID = txtUserID AND Password = txtPassword
```

c.

```
SELECT Password, UserID FROM Users
```

d.

All of the above

Answer: _____

4. Which ONE of the following SQL statements would be most vulnerable to SQL injection?

a.

```
SELECT UserID, Password FROM Users WHERE UserID = @UserID  
AND Password = @Password
```

b.

```
SELECT * FROM Users WHERE UserID = txtUserID AND Password =  
txtPassword
```

c.

SELECT Password, UserID FROM Users

d.

All of the above.

Answer: _____

5. Parameterized SQL commands are a preferred way of coding data access?

a. Yes

b. No

Answer: _____

6. When using SQL in a web or windows form, which of the following are acceptable ways for dealing with placeholders in the SQL statements in a secure manner?

a. Stored Procedures

b. Parameterized SQL statements

c. Concatenated SQL strings

d. Use try-catch and concatenated SQL strings.

e. None of these answers are correct.

Answers: _____

7. Which of the following methods of coding is recommended as the most appropriate for data access?

- a. Parameterized SQL commands
- b. Stored Procedures.
- c. Use try-catch and concatenated strings
- d. Concatenated SQL strings

Answer: _____

8. Which of the following are valid approaches to authentication when connecting to a SQL server database?

- a. Login
- b. Integrated Authentication (Windows Authentication)
- c. Portion Authentication (Built Authentication)
- d. Mixed Authentication (SQL authentication)

Answers: _____

10. Why would an authentication method be regarded as the best method for authenticating users?

- a. Administrator and User login is simple.
- b. Requires no password because Windows takes care of security issues.
- c. Secure with a username and password.
- d. Secure with an encrypted username and password.
- e. Easiest method of them all.

Answer: _____

11. When sensitive data has been encrypted, only the authorized parties can analyze the data?

- a. True
- b. False.

Answer: _____

12. Which of the following should always be encrypted?

- a. Sensitive parts of the web.config file.
- b. SQL statements
- c. Sensitive information such as ID numbers and account numbers.
- d. A connection string
- e. Passwords

Answers: _____

13. Which are the preferred ways of encryption?

- a. Using a hashing algorithm.
- b. Windows Data Protection API (DPAPI)
- c. 64-bit Algorithm for the best encryption.
- d. Writing your own encryption using an algorithm with random numbers and alphabets.
- e. ASP.NET RegIIS

Answers: _____

14. Where does the **Principle of Least Privilege** apply?

- a. Login
- b. 64-bit Algorithm for best encryption.
- c. In the code behind for assigning user rights.
- d. Database (Admin and User rights of the system).

Answer: _____

15. Why is the **Principle of Least Privilege** seen as being important in software development?

- a. For security reasons.
- b. To give certain people rights to certain people and security reasons.
- c. For users of the system to be guided on every action they do on the system.
- d. For users to be able to complete their tasks in time.

Answer: _____

16. How could the **Principle of Least Privilege** be applied in an example of a system where they are employees involved?

- a. Give rights to the manager of a department and allow the manager to deal with logging in of employees.
- b. Giving employees rights according to the departments they belong to.
- c. Having a person looking at them whilst they sign in to the system.
- d. Create a password for users depending on a department.

Answer: _____

17. What are the two types of validating input?

- a. Whitelisting
- b. Blacklisting
- c. If Statements
- d. Java API Validation
- e. All of the above.

Answers: _____

18. Which of the following methods is the best for validating input?

- a. Whitelisting
- b. Blacklisting
- c. If statements
- d. Java API Validation
- e. None of the above.

Answer: _____

19. Why would a method for validating input be regarded as the best method?

- a. Tell the input field which values to allow and the rest of the values will not be allowed.
- b. Is safe and easy to use.
- c. Is mostly used by enterprise companies.
- d. Tell the input field which values not to allow and the rest will just be allowing in the system.

Answer: _____

20. What is the use of a connection string in an application?

- a. The connection string allows the application and the database to communicate with each other faster.
- b. Makes an application run faster when connecting to the database.
- c. Specifies information about a data source and the means of connecting to it.
- d. All of the above.

Answer: _____

21. How many times should a connection string appear in an application?

- a. 2 times.

- b. Only once.
- c. In every form so that it can communicate with the database.
- d. In the Data Access Layer classes that connect to the data base.

Answer: _____

22. Which of the following are true about a connection string?

- a. Connection strings should only be in the web.config file and should be encrypted.
- b. Connection strings should be referenced only once in a helper class in your Data Access Layer.
- c. Connection strings should not be encrypted.
- d. Connection strings should be a 64-bit long string.

Answers: _____

23. Which of the following is false about connection strings?

- a. If you have a wrong connection string format but correct database name, you can still connect to the database.
- b. The connection string is stored in the database.
- c. Connections strings have the correct database name and be in the correct format.
- d. Without the connection string, the database and the application will not communicate.

Answers: _____