

An Educational Intervention for Teaching Secure Coding Practices

Vuyolwethu Mdunyelwa, Lynn Futcher, Johan Van Niekerk

▶ **To cite this version:**

Vuyolwethu Mdunyelwa, Lynn Futcher, Johan Van Niekerk. An Educational Intervention for Teaching Secure Coding Practices. 12th IFIP World Conference on Information Security Education (WISE), Jun 2019, Lisbon, Portugal. pp.3-15, 10.1007/978-3-030-23451-5_1 . hal-02365736

HAL Id: hal-02365736

https://inria.hal.science/hal-02365736

Submitted on 15 Nov 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License

An Educational Intervention for Teaching Secure Coding Practices*

Vuyolwethu Mdunyelwa[1][0000−0001−9625−0925],
Lynn Futcher[1][0000−0003−0406−8718], and
Johan van Niekerk[1,2][0000−0003−1739−4563]

[1] Nelson Mandela University, Port Elizabeth, South Africa
{vuyolwethu.mdunyelwa, lynn.futcher, johan.vanniekerk}@mandela.ac.za
[2] Noroff University College, Kristiansand, Norway
johan.vanniekerk@noroff.no

Abstract. 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 paper is on technical controls with a specific focus on securing web applications. The secure coding practices used in this research are based on OWASP. An initial investigation found that there was a general lack of adherence to these secure coding practices by third year software development students doing their capstone project at a South African University. This research therefore focused on addressing this problem by developing an educational intervention to teach secure coding practices, specifically focusing on the data access layer of web applications developed in the .NET environment. Pre-tests and post-tests were conducted in order to determine the effectiveness of the intervention. Results indicated an increase in both knowledge and behaviour regarding the identified secure coding practices after exposure to the intervention.

Keywords: Educational Intervention · Secure Coding Practices · OWASP · Web Application Security.

1 Introduction

With the recent increase in cyber-related attacks, cybersecurity is becoming a key area of concern for many organisations. Web applications often handle very sensitive data, used for carrying out critical tasks such as banking, online shopping and online tax filing [9]. These applications are trusted by billions of users for performing such daily activities. However, 75% of all attacks on the internet are executed through the application layer of the OSI model [6], and more than 76% of web applications have vulnerabilities [2].
Handling risks related to the security of web applications is a major challenge for

* 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.

many organizations. Not surprisingly, Web applications have recently received attention from academia and industry to initiate some defence mechanisms to protect them from security threats [9]. Many of these Web applications have common vulnerabilities which can be easily corrected [18] through introducing secure coding practices.
The secure coding practices used in this research are based on OWASP. An initial investigation found that there was a general lack of adherence to these secure coding practices by third year software development students doing their capstone project at a South African University. This research therefore focused on addressing this problem by developing an educational intervention to teach secure coding practices, specifically focusing on the data access layer of web applications developed in the .NET environment.
The following section highlights the related literature, while Section 3 provides the research design. Section 4 presents the educational intervention including both the knowledge and behavioural components. This is followed by Section 5 which provides the results of the verification of the educational intervention before concluding in Section 6.

2 Related Literature

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. On average, over 340,000 web attacks were blocked from web applications per day in 2014 [2]. Although this improved to 229,000 in 2016 [2], it still remains a serious concern since most attacks are no longer on the networks, but more on the software applications that run on the application layer. 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 applications to avoid known vulnerabilities. Those web applications without known vulnerabilities probably adhere to some form of best practice for secure software development. This is true as some researchers suggest that applying such practices and methodologies can improve security in software application [1, 7].
There are various organisations and institutions responsible for developing standards and best practices. These include the National Institute of Standards and Technology (NIST), the International Organizations for Standardization (ISO) and the International Electro-Technical Commission (IEC), the Microsoft Developer Network (MSDN) and the Open Web Application Project (OWASP) which provides best practices for improving security in web applications.
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. OWASP is known by many organisations for its Top 10 Vulnerability List (Table 1) that it publishes and updates periodically [4, 6, 11]. This list focusses on identifying the most serious web ap-

plication security vulnerabilities for many organisations [16]. The Top 10 list changes according to which vulnerability is most dominant at any given time.

Table 1: OWASP Top 10 Vulnerability List 2017 [15].

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 know 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.

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 specific secure coding practices. Table 2 presents the nine secure coding practices (SP1 to SP9) for data access based on OWASP and used in this study. These relate to some of the vulnerabilities shown in Table 1. As an example, parameterised SQL commands (SP1), or the use of stored procedures (SP6), can block SQL injections. Therefore more than one control can reduce a vulnerability.

Table 2: Secure Coding Practices. Adapted from [16].

SP	**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 strings.
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.

The secure coding practices shown in Table 2 are referred to using the codes SP1 to SP9 throughout this paper. If one of them is not properly handled, it can be easy for an attacker to access and modify information that is in the database. For example, if the connection string is found in other parts of the application code 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 [1, 3]. Similarly, they cannot implement these security controls unless they are taught how they work [8]. Once software developers have been taught about secure coding practices, it is more likely that they will have the requisite knowledge [5]. However, there has to be some form of compliance instrument to monitor their adherence, since it is known that people with the requisite knowledge do not always behave accordingly. Therefore, an educational intervention that focuses on both knowledge and behaviour was developed and provided to software development students to improve the security of their web applications.

3 Research Design

This research was conducted in the School of Information and Communication Technology at a comprehensive institution in South Africa, offering both degrees and vocational qualifications. In this case, the sample was drawn from students registered for their third year in the National Diploma: Software Development. In South Africa, there are no locally recognised curricular guidelines for computing. Many universities therefore rely on the recommendations provided in global

computing curricular publications. The Association of Computing Machinery (ACM) Information Technology curricular guidelines have been used to model IT qualifications. The IT2008 and the more recent IT2017 curricular guidelines require students in computing and engineering disciplines to engage in a capstone project during their final year of study [13, 12]. Since the diploma is a three year qualification, students are required to complete a capstone project in their third year of study. These capstone projects take place over a full year of study. According to the ACM IT curricular guidelines, capstone projects should typically adhere to the following [13, 12]:

- 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.

Students registered for the diploma are introduced to programming and business application systems development. Therefore, most of their capstone projects focus on developing applications for solving real world problems using business applications. When students choose the capstone projects, many of them focus on web, mobile or gaming applications, while a few develop desktop 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.

This research focused on two aspects relating to secure coding practices, namely knowledge and behavioural compliance of the students and involved four main phases:

- **Phase 1** was the first phase for this research which started off by analysing students' behaviour relating to secure coding practices. This was done by conducting a code review on previously completed third year capstone projects, which were developed in the .NET environment. The results for this behavioural analysis indicated low levels of compliance to the identified secure coding practices.
- **Phase 2** addressed the knowledge assessment phase for this research, which assessed students' knowledge relating to secure coding practices. This was achieved using a questionnaire, which served as a pre-test for this study. Results from the pre-test indicated that students lacked knowledge relating to secure coding practices. Therefore, students lacked in both the knowledge and behavioural aspects.
- **Phase 3** comprised of an educational intervention for addressing both the knowledge and behavioural aspects, which students lacked in Phase 1 and 2. In terms of the knowledge aspect, students were provided with online lessons relating to secure coding practices to work through; in terms of the

behavioural aspect, students were given a checklist to check their application code against the listed secure coding practices.
- **Phase 4** involved the verification of the educational intervention for this research. The first part of this phase was the knowledge verification (Phase 4A), and the second part was the behavioural verification (Phase 4B).

The results for Phase 1 and 2 were published in the 2017 Human Aspects in Information Security and Assurance (HAISA) conference [14]. The focus of this paper is therefore on Phases 3 and 4. The following section describes the educational intervention (Phase 3), while Phase 4 (A and B) are discussed in Section 5.

4 Educational Intervention

The educational intervention was split into two parts, where the first part focused on the knowledge, and the second part focused on the behaviour of students relating to 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 [17]. Both the knowledge and behavioural components of this research were designed using the identified secure coding practices in Table 2.

4.1 Knowledge Component

The knowledge component for this research took the form of a blended learning course, called the Web Application Security Course, that students worked through to improve their knowledge regarding secure coding practices.

Design of the Knowledge Component The knowledge component for this research included online lessons that the researcher designed using the identified secure coding practices. For each of the secure coding practices, their importance and the security implications if they were ignored were explained. The online learning platform that was used to design the lessons was the *Moodle Learning Management System* that runs on the university's website. 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 [10]. Figure 1 provides an overview of the process followed by the students when completing the online lessons on Moodle.

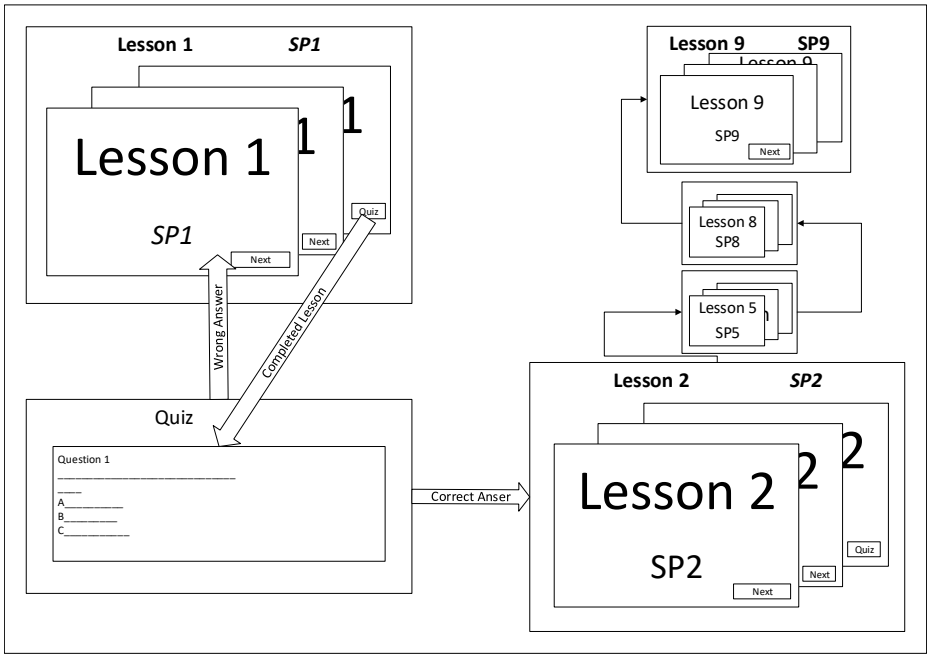


Fig. 1: Lesson Content Process Flow.

The 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 a single lesson. 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 generated quiz question before continuing to the next lesson. If the student 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 generated. Alternatively, if they selected the correct answer, they were allowed to continue to the next lesson.
A brief overview of each secure coding practice (SP1 to SP9) as listed in Table 2, within the knowledge component follows:

- 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 highlights 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.
- 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): For this secure coding practice, an analogy is used 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.

Administering the Knowledge Component The Web Application Security lessons were prepared by the researcher and distributed to the students on Moodle. 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 in the computer laboratories at the university as soon as it was made active and available to them. A total of 120 students completed the online lessons. The students had to work through all the lessons, since they were

required to take a quiz which served as a Post-test (Phase 4A) for which marks were recorded.

4.2 Behavioural Compliance Monitoring Instrument

Although it is possible for a student to have the requisite 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 firstly 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 as part of their third year capstone projects.

Design of Behavioural Compliance Instrument The behavioural compliance instrument took the form of a checklist as seen in Table 3. The code review checklist for this research was adapted from the secure coding practices in Table 2 and was provided to the students electronically via Moodle.

Table 3: Code Review Checklist.

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

Conducting the Behavioural Compliance Instrument 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 using the checklist provided in Table 3. Having worked through the knowledge component, as discussed in Section 4.1, the students should have

acquired the requisite 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 for their code and to ensure compliance to the secure coding practices.

5 Effectiveness of the Educational Intervention

Once students had completed the educational intervention, it was necessary to determine its effectiveness. 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, showing behavioural compliance.
The verification of the knowledge component was achieved through an online quiz distributed to the students through the Moodle site as discussed in Section 5.1. Verification of the behavioural compliance component took the form of a code review by the researcher on the students' capstone projects as discussed in Section 5.2.

5.1 Knowledge Verification

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. The 113 students who completed the post-test were only allowed to work through the post-test once. 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. A comparison of the knowledge pre- and post-test is shown in Table 4.

Table 4: Knowledge Assessment and Verification Results (Pre-Test vs Post-Test).

Phases	**SP1**	**SP2**	**SP3**	**SP4**	**SP5**	**SP6**	**SP7**	**SP8**	**SP9**
Phase 2	74%	36%	30%	58%	26%	39%	20%	3%	1%
Phase 4A	95%	95%	89%	91%	91%	77%	93%	88%	83%
Variance	**21%**	**59%**	**59%**	**33%**	**65%**	**38%**	**73%**	**85%**	**82%**

Table 4 shows the results for Phase 2, Knowledge Assessment (pre-test), and Phase 4A, Knowledge Verification (post-test). There was a substantial im-

provement in the students' knowledge as indicated in the second row, Phase 4A. Students' knowledge has improved in all of the secure coding practices (SP1 to SP9), as seen in the variances. SP2, SP3, SP5 and SP7 showed reasonable improvements, while SP8 and SP9 showed the highest improvements with variances above 80%. As mentioned previously, knowledge acquisition 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 applications behavioural compliance monitoring was required.

5.2 Behavioural Verification

The behavioural verification instrument used was the same checklist used in Phases 1 and 3 as shown in Table 3. The checklist was used by the researcher to conduct a code review on the third year 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 reviewed five web forms per project, which connected to the database and were related to the capstone projects' main functionality. 17 groups were present for the code review, and they were all reviewed successfully, in the presence of the students who belonged to the group being reviewed. Table 5 shows the results from the behavioural analysis for the students before and after exposure to the educational intervention.

Table 5: Behavioural Verification Results (Phase 1 and 4B).

Phases	**SP1**	**SP2**	**SP3**	**SP4**	**SP5**	**SP6**	**SP7**	**SP8**	**SP9**
Phase 1	86%	84%	77%	60%	N/A	38%	N/A	68%	31%
Phase 4B	96%	96%	100%	91%	N/A	96%	N/A	100%	100%
Variance	**10%**	**12%**	**23%**	**31%**	**N/A**	**58%**	**N/A**	**32%**	**69%**

As can be seen from the results in Table 5, there was an improvement in students' adherence to secure coding practices after the educational intervention, with most capstone project groups having adhered to all the secure coding practices. Although SP5 and SP7 were recommended by OWASP, they were not required by the capstone projects from which the sample for this research was drawn. All averages per secure coding practice were between 90% and 100%, with SP3, SP8, and SP9 showing 100% compliance. SP6 and SP9 showed the largest improvements of 58%(SP6) and 69%(SP9) respectively, while SP3, SP4 and SP8 showed good improvements of between 20% and 35%.

6 Conclusion

The results of this study indicate that students' adherence to secure coding practices can be positively impacted through a formal educational intervention. However, it is important that such an intervention addresses both the knowledge and behaviour of students since having the requisite knowledge does not ensure compliance. It is for this reason that a behavioural compliance monitoring instrument formed part of the study. This is a step towards educating students in secure application development which is essential in addressing the many security vulnerabilities existing in Web applications today.
Limitations of this study do exist. Firstly, this study addressed only the identified secure coding practices which were determined from OWASP. Secondly, the identified secure coding practices only focused on the data access layer of Web applications developed in the .NET environment. Future research could investigate similar interventions within various other application development contexts.

7 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).

References

1. Bishop, M., Dai, J., Dark, M., Ngambeki, I., Nico, P., Zhu, M.: Evaluating secure programming knowledge. In: IFIP World Conference on Information Security Education. pp. 51–62. Springer (2017)
2. Chandrasekar, K., Cleary, G., Cox, O., O Gorman, B.: Internet Security Threat Report. Tech. Rep. April, Symantec (2017), https://www.symantec.com/security-center/threat-report
3. Chi, H., Jones, E.L., Brown, J.: Teaching Secure Coding Practices to STEM Students. Proceedings of the 2013 on InfoSecCD '13 Information Security Curriculum Development Conference - InfoSecCD '13 pp. 42–48 (2013). https://doi.org/10.1145/2528908.2528911, http://dl.acm.org/citation.cfm?doid=2528908.2528911
4. Chung, S., Hansel, L., Bai, Y., Moore, E., Taylor, C., Crosby, M., Heller, R., Popovsky, V., Endicott-Popovsky, B.: What approaches work best for teaching secure coding practices? 2014 HUIC Education & STEM Conference (2014)
5. Conklin, A., White, G.: A graduate level assessment course: A model for safe vulnerability assessments. In: Proceedings of the 9th Colloquium for Information Systems Security Education (2005)
6. Customs Solutions Group: A CISO ' s Guide to Application Security. Tech. rep., Customs Solutions Group (2012), http://h30528.www3.hp.com/Security/CISOGuideToApplicationSecurity.pdf

7. Dark, M., Ngambeki, I., Bishop, M., Belcher, S.: Teach the Hands, Train the Mind ... A Secure Programming Clinic! Proceeding of the 19th Colloquium for Information System Security Education (2015)
8. Dark, M., Stuart, L., Ngambeki, I., Bishop, M.: Effect of the secure programming clinic on learners' secure programming practices. In: Journal of The Colloquium for Information System Security Education. vol. 4, pp. 18–18 (2016)
9. Deepa, G., Thilagam, P.S.: Securing web applications from injection and logic vulnerabilities: Approaches and challenges. Information and Software Technology **74**, 160–180 (2016), http://dx.doi.org/10.1016/j.infsof.2016.02.005
10. Florian, T.P., Zimmerman, J.P.: Understanding by Design , Moodle , and Blended Learning : A Secondary School Case Study. MERLOT Journal of Online Learning and Teaching **11**(1), 120–128 (2015)
11. Li, X., Xue, Y.: A survey on server-side approaches to securing web applications. ACM Computing Surveys (CSUR) **46**(4), 54 (2014)
12. Lunt, B., Sabin, M., Hala, A., Impagliazzo, J., Zhang, M.: Information Technology Curricula 2017. Tech. rep., Association for Computing Machinery (ACM) IEEE Computer Society (2017)
13. Lunt, B.M., Ekstrom, J.J., Lawson, E.: Curriculum guidelines for undergraduate degree programs in Information Technology. Tech. rep., Association for Computing Machinery (ACM) IEEE Computer Society (2008)
14. Mdunyelwa, V.S., Niekerk, J.F.V., Futcher, L.A.: Secure coding practices in the software development capstone projects. In: Eleventh International Symposium on Human Aspects of Information Security & Assurance, HAISA 2017, Adelaide, Australia, November 28-30, 2017, Proceedings. pp. 282–291 (2017), http://www.cscan.org/openaccess/?paperid=353
15. OWASP: OWASP Top 10 (2017), https://www.owasp.org/index.php/Top_10-2017_Top_10
16. OWASP: The OWASP Foundation (2017), https://www.owasp.org/index.php/Main_Page
17. Vroom, C., Von Solms, R.: Towards information security behavioural compliance. Computers and Security **23**(3), 191–198 (2004)
18. Zhu, J., Xie, J., Lipford, H.R., Chu, B.: Supporting secure programming in web applications through interactive static analysis. Journal of Advanced Research **5**(4), 449–462 (2014), http://dx.doi.org/10.1016/j.jare.2013.11.006