XIAMEN UNIVERSITY MALAYSIA

Course Code : SWE301

Course Name : Software Verification and Validation

Lecturer : Dr Subashini Raghavan

Academic Session : 2024/09

Assessment Title : Assignment 1

Submission Due Date : 21 Oct 2024

Prepared by :

Student ID	Student Name
SWE2209518	Huang Tianjing

Date Received :

Feedback from Lecturer:

Mark:

XIAMEN UNIVERSITY MALAYSIA

Own Work Declaration

I/We hereby understand my/our work would be checked for plagiarism or other misconduct, and the softcopy would be saved for future comparison(s).

I/We hereby confirm that all the references or sources of citations have been correctly listed or presented and I/we clearly understand the serious consequence caused by any intentional or unintentional misconduct.

This work is not made on any work of other students (past or present), and it has not been submitted to any other courses or institutions before.

Signature:

黄天竞

Date:

21/10/2024

Contents

Enhancing Web Application Verification and Validation: Identifying Challenges and Implementing AI-Driven Solutions

Abstract

As web applications grow more complex, they encounter numerous verification and validation (V&V) challenges. This paper examines key limitations and challenges related to requirement changes, cross-platform compatibility, and security testing in web applications. Existing techniques to address these issues, such as agile methodologies, automated cross-browser testing tools like Selenium, and security frameworks like OWASP ZAP, are evaluated. Furthermore, this paper introduces advanced solutions, including machine learning for automated test case generation and Generative Adversarial Networks (GANs) for simulating novel security attacks. Together, these methods aim to enhance the V&V process, ensuring more secure and reliable web applications in a dynamic development environment.

***Keywords**: Web application, Verification and Validation, Cross-platform compatibility, Security testing, Agile methodologies, Automated testing, Machine learning, Generative Adversarial Networks (GANs).*

1 Introduction

This assignment focuses on the challenges of verification and validation, as well as the corresponding measures, in Web Application Systems. Following is a brief definition of web applications: Web Application Systems are applications delivered over a network, typically the internet, that run on remote servers and provide an interactive interface to users through web browsers. Unlike traditional desktop applications, web applications do not require software installation on user devices; users can access them simply through a browser. This ease of deployment and maintenance makes them suitable for a wide range of use cases, but it also introduces several related challenges.

2 Research questions

2.1 What are the verification and validation limitations or challenges for web application systems?

2.1.1 Complexity of requirement changes

With the rapid development of internet technology, the volatility of client needs and available technology continues to rise. In internet-based systems, technology, development skills, business models, and competing systems change so rapidly that the domain is often not only poorly understood but also constantly evolvingLowe (2003).

At the early stages of a project, requirements are often unclear or unstable, and stakeholders may have differing understandings of the final goals and functions of the system. This uncertainty makes it difficult to create accurate test cases and validation standards during the verification and validation phases, increasing the risk of undetected issues.

In large projects, changes in requirements can affect the work of multiple teams. For instance, when front-end and back-end teams are implementing a feature, changes in requirements may lead to collaboration issues between them. A lack of effective communication and collaboration may result in incomplete test coverage and inconsistent feature implementation, further complicating the verification and validation process.

After each change in requirements, regression testing is necessary to ensure that new code does not introduce new defects or disrupt existing functionality. However, as the number and complexity of requirements increase, the workload for regression testing can become significantly greater, especially in the absence of automated testing support, making this burden even more pronounced.

2.1.2 Cross-platform compatibility

Different browsers (such as Chrome, Firefox, Safari, and Edge) may exhibit variations in rendering pages, executing JavaScript, and handling CSS styles. These differences can cause the same web application to present distinct appearances and behaviors across various browsers. Some browsers may not support certain HTML5 or CSS3 features, resulting in specific functionalities becoming unusable and negatively affecting user experience. Therefore, comprehensive testing across all major browsers is crucial, although this significantly increases the testing workload.
Modern users access web applications on a variety of devices, including smartphones, tablets, laptops, and desktop computers. Each device has different screen sizes, resolutions, and hardware capabilities, which can affect the application's layout and responsiveness. Xanthopoulos and Xinogalos (2013)Designing responsive web applications requires consideration of the characteristics of different devices to ensure that the application displays well on all of them. Testing and optimizing for each device individually not only consumes a lot of time but can also lead to unforeseen issues.
Web applications need to run on multiple operating systems, such as Windows, macOS, Linux, iOS, and Android. The network settings, security policies, and file handling methods of different operating systems can lead to varied functional performance of the application. For example, certain operating systems may restrict access to specific APIs or limit file system operations, which can impact some features of the application.

2.1.3 Complexity of security testing

The attack surface of web applications is often quite extensive, comprising multiple components such as the front-end user interface, back-end servers, databases, and APIs. Each of these components has its specific potential vulnerabilities that attackers may exploit. For instance, attackers can perform SQL injection attacks through input fields, execute XSS attacks via web forms, or leverage CSRF vulnerabilities through forged requests.Aydos, Aldan, Coşkun, and Soydan (2022) This necessitates that security testing covers all potential attack paths, which increases the complexity of the testing process.
Cybersecurity threats are continually evolving, with new attack methods and tools emerging frequently. Attackers are constantly searching for new ways to breach security measures, which means that security testing for web applications must be continually updated to address these emerging threats. This requires security testing teams not only to monitor current security vulnerabilities and attack patterns but also to anticipate potential new threats to maintain effective protection.

2.2 What existing verification and validation techniques can we use to tackle the challenges found in answering RQ1?

2.2.1 Agile methodologies

Adopting agile methodologies, such as Scrum or Kanban, provides teams with a flexible framework that enables them to quickly adapt to changing requirements and environments. Unlike traditional waterfall models, agile methodologies encourage frequent interaction and feedback with stakeholders at every stage of the projectClutterbuck, Rowlands, and Seamons (2009).

In agile development, projects are divided into multiple short iterations, commonly referred to as ”sprints,” each with clear goals and deliverables. Agile management tools such as JIRA, Trello, and Azure DevOps can be utilized to track tasks, manage workflows, and visualize progress. This approach allows the team to demonstrate work results at the end of each iteration, enabling timely adjustments to requirements and priorities, ensuring that the final product better meets user expectations and market changes.

At the start of each sprint, the team holds a planning meeting to determine sprint goals and priority tasks. Stakeholders should participate in discussions to ensure a shared understanding of the requirements. The team conducts brief daily stand-up meetings where members share what they are working on, the challenges they face, and the support they need.

At the end of each sprint, the team holds a review meeting with stakeholders to showcase completed features. Stakeholders can provide feedback, helping the team adjust the direction of future iterations. Following the review, the team conducts a retrospective meeting to discuss the successes and challenges of the current iteration. Ge, Paige, Polack, Chivers, and Brooke (2006)Team members collaboratively reflect on ways to improve processes for the next iteration.

2.2.2 Automated cross-browser testing tools

Using automated testing tools, such as Selenium, can significantly simplify the process of testing web applications across different browsers and devices. These tools simulate various operating environments (such as different browser versions, device types, and operating systems) to achieve consistent and efficient testing, thereby eliminating the burdensome manual testing work and improving test coverage and efficiency.

Based on the key functionalities and user flows of the application, automated test scripts should be written to cover important scenarios across major browsers (such as Chrome, Firefox, Safari, and Edge) and various devices (like smartphones, tablets, and desktop computers). Utilizing Selenium’s API and capabilities makes it straightforward to implement this.

The automated tests should be integrated into the Continuous Integration/Continuous Deployment (CI/CD) pipelineChoudhary, Versee, and Orso (n.d.). This means that whenever code is committed to the version control system, the automated tests will be triggered to run. By regularly executing these tests within the CI/CD process, teams can quickly identify and resolve cross-platform compatibility issues, ensuring that compatibility standards are met with every release.

As web applications and browsers continue to evolve, automated test scripts need to be maintained and updated regularly to accommodate new features and changes. Development teams should periodically review the test coverage to ensure that all new functionalities and alterations are appropriately tested.

2.2.3 Security testing framework

Implementing a comprehensive security testing framework, such as OWASP ZAP or Burp Suite, can effectively help identify vulnerabilities within web applications. These tools are designed to automatically detect common security issues, such as SQL injection, XSS, and CSRF, providing a more thorough assessment of the application’s security posture.

To implement this, security testing should be integrated into the development lifecycle, adopting a DevSecOps approach. This means that automated scans should be conducted regularly, alongside manual penetration testing, to ensure that security vulnerabilities are consistently identified and addressedPeroli, De Meo, Viganò, and Guardini (2018). By incorporating security testing into the Continuous Integration/Continuous Deployment (CI/CD) pipeline, teams can receive timely feedback regarding any vulnerabilities introduced by new code, enabling them to address security concerns proactively before deployment.

2.3 What are some recommendations for the remaining challenges not tackled by the existing techniques found in answering RQ2?

2.3.1 Machine learning for automated testing

Using machine learning to enhance test case generation is an effective strategy to tackle the challenges posed by Automated Cross-Browser Testing. Machine learning can analyze historical defect data to identify potential defect patterns and risk areas. By conducting an in-depth analysis of past project defect records, machine learning models can understand which functionalities are more prone to issues, thereby generating test cases targeted at these high-risk areas. For example, the model can identify specific modules that frequently failed in previous versions, guiding the testing team to apply stricter testing to these modules.

Machine learning can also analyze user behavior data to gain insights into user operations in actual usage scenarios. By capturing user behavior characteristics and usage habits, machine learning models can generate more targeted test cases, ensuring that testing scenarios align closely with real usage situationsChen and Chau (2004). This data-driven approach not only enhances the relevance of testing but also improves the effectiveness of test cases, better reflecting the true needs of users.

Additionally, the adaptive capabilities of machine learning allow for the continuous optimization of generated test cases over time. The model can adjust its generation strategies based on newly collected data, ensuring that test cases consistently meet current requirements and adapt to changing environments. This flexibility is particularly important in fast-paced development cycles, as it can reduce unnecessary repetitive work resulting from changing requirements.

2.3.2 Generative adversarial networks for security testing

GANs (Generative Adversarial Networks) are a form of deep learning technique that can be used to generate highly realistic data or patterns. In the context of security testing, GANs can simulate advanced and sophisticated attack patterns that go beyond typical known vulnerabilities. Unlike predefined test cases that focus on known vulnerabilities, GANs can be trained to create entirely new types of attack scenarios. This allows for the discovery of hidden or non-obvious vulnerabilities that static rule-based systems or signature-based detection tools might miss. For instance, GANs can learn to evade security measures by generating novel inputs designed to exploit potential weaknesses in code logic or system behavior.

Cyber threats constantly evolve as attackers innovate new methods to bypass defenses. GANs can be continuously trained on updated data, such as real-world attack patterns and new threat reports, allowing them to simulate emerging and zero-day attack techniques. By doing so, they provide a dynamic and evolving security testing framework, which traditional static testing tools lackChowdhary, Jha, and Zhao (2023). This adaptability makes them particularly valuable in environments where security needs to keep up with fast-evolving threats.

Standard security tools often struggle to detect multi-step attacks that may involve a sequence of actions over time, such as an attacker gaining low-level access and gradually escalating their privileges. GANs can be trained to simulate complex, multi-stage attacks, mimicking how an attacker might infiltrate and compromise systems in stages. This allows security teams to better anticipate and mitigate risks posed by sophisticated adversaries who execute advanced persistent threats (APTs).

3 Conclusion

In this paper, I focus on the major challenges in the verification and validation (V&V) of web application systems, particularly focusing on issues related to changing requirements, cross-platform compatibility, and security testing. The inherent complexity and evolving nature of web technologies make it difficult to ensure accurate validation and effective verification across various components of the system.

To address these issues, I reviewed existing V&V techniques. Agile methodologies provide teams with flexibility to adapt to frequent requirement changes through iterative feedback loops. Automated cross-browser testing tools like Selenium help ensure consistent user experiences across multiple platforms, improving the verification process. Security testing frameworks, such as OWASP ZAP and Burp Suite, focus on identifying known vulnerabilities, strengthening the validation of system security.

Nevertheless, some challenges remain unresolved, particularly those tied to the fast-paced evolution of requirements and the complexity of modern cyber threats. Machine learning offers a promising approach to enhance V&V by automating test case generation based on historical defect patterns and user behavior data. This enables more accurate validation by focusing on high-risk areas and aligning testing scenarios with real-world usage. Similarly, Generative Adversarial Networks (GANs) can significantly improve security validation by simulating advanced, novel attack patterns beyond the scope of traditional security tools. GANs dynamically create new attack scenarios, enabling the detection of hidden vulnerabilities, which enhances both the verification of system robustness and the validation of security measures.

In conclusion, while traditional V&V techniques provide a solid foundation for web application testing, the integration of advanced AI-driven methods like machine learning and GANs offers enhanced adaptability, accuracy, and comprehensiveness. These technologies strengthen the overall V&V process by addressing the complexities of evolving requirements and security challenges, ensuring higher-quality and more secure web applications in today's dynamic development landscape.

References

Aydos, M., Aldan, Ç., Coşkun, E., & Soydan, A. (2022). Security testing of web applications: A systematic mapping of the literature. *Journal of King Saud University-Computer and Information Sciences*, *34*(9), 6775–6792.

Chen, H., & Chau, M. (2004). Web mining: Machine learning for web applications. *Annual review of information science and technology*, *38*(1), 289–329.

Choudhary, S. R., Versee, H., & Orso, A. (n.d.). Webdiff: Automated identification of cross-browser issues in web applications. In *2010 ieee international conference on software maintenance* (pp. 1–10).

Chowdhary, A., Jha, K., & Zhao, M. (2023). Generative adversarial network (gan)-based autonomous penetration testing for web applications. *Sensors*, *23*(18), 8014.

Clutterbuck, P., Rowlands, T., & Seamons, O. (2009). A case study of sme web application development effectiveness via agile methods. *Electronic Journal of Information Systems Evaluation*, *12*(1), pp13–26.

Ge, X., Paige, R. F., Polack, F. A., Chivers, H., & Brooke, P. J. (2006). Agile development of secure web applications. In *Proceedings of the 6th international conference on web engineering* (pp. 305–312).

Lowe, D. (2003). Web system requirements: an overview. *Requirements Engineering*, *8*, 102–113.

Peroli, M., De Meo, F., Viganò, L., & Guardini, D. (2018). Mobster: A model-based security testing framework for web applications. *Software Testing, Verification and Reliability*, *28*(8), e1685.

Xanthopoulos, S., & Xinogalos, S. (2013). A comparative analysis of cross-platform development approaches for mobile applications. In *Proceedings of the 6th balkan conference in informatics* (pp. 213–220).

SWE2209518.docx

by Tianjing Huang

Submission date: 21-Oct-2024 07:25PM (UTC+0800)

Submission ID: 2492256661

File name: SWE2209518.docx (52.73K)

Word count: 2327

Character count: 14400

XIAMEN UNIVERSITY MALAYSIA

Enhancing Web Application Verification and Validation: Identifying Challenges and Implementing AI-Driven Solutions

Abstract

As web applications grow more complex, they encounter numerous verification and validation (V&V) challenges. This paper examines key limitations and challenges related to requirement changes, cross-platform compatibility, and security testing in web applications. Existing techniques to address these issues, such as agile methodologies, automated cross-browser testing tools like Selenium, and security frameworks like OWASP ZAP, are evaluated. Furthermore, this paper introduces advanced solutions, including machine learning for automated test case generation and Generative Adversarial Networks (GANs) for simulating novel security attacks. Together, these methods aim to enhance the V&V process, ensuring more secure and reliable web applications in a dynamic development environment.

***Keywords**: Web application, Verification and Validation, Cross-platform compatibility, Security testing, Agile methodologies, Automated testing, Machine learning, Generative Adversarial Networks (GANs).*

1 Introduction

This assignment focuses on the challenges of verification and validation, as well as the corresponding measures, in Web Application Systems. Following is a brief definition of web applications: Web Application Systems are applications delivered over a network, typically the internet, that run on remote servers and provide an interactive interface to users through web browsers. Unlike traditional desktop applications, web applications do not require software installation on user devices; users can access them simply through a browser. This ease of deployment and maintenance makes them suitable for a wide range of use cases, but it also introduces several related challenges.

2 Research questions

2.1 What are the verification and validation limitations or challenges for web application systems?

2.1.1 Complexity of requirement changes

With the rapid development of internet technology, the volatility of client needs and available technology continues to rise. In internet-based systems, technology, development skills, business models, and competing systems change so rapidly that the domain is often not only poorly understood but also constantly evolvingLowe (2003).

At the early stages of a project, requirements are often unclear or unstable, and stakeholders may have differing understandings of the final goals and functions of the system. This uncertainty makes it difficult to create accurate test cases and validation standards during the verification and validation phases, increasing the risk of undetected issues.

In large projects, changes in requirements can affect the work of multiple teams. For instance, when front-end and back-end teams are implementing a feature, changes in requirements may lead to collaboration issues between them. A lack of effective communication and collaboration may result in incomplete test coverage and inconsistent feature implementation, further complicating the verification and validation process.

After each change in requirements, regression testing is necessary to ensure that new code does not introduce new defects or disrupt existing functionality. However, as the number and complexity of requirements increase, the workload for regression testing can become significantly greater, especially in the absence of automated testing support, making this burden even more pronounced.

2.1.2 Cross-platform compatibility

Different browsers (such as Chrome, Firefox, Safari, and Edge) may exhibit variations in rendering pages, executing JavaScript, and handling CSS styles. These differences can cause the same web application to present distinct appearances and behaviors across various browsers. Some browsers may not support certain HTML5 or CSS3 features, resulting in specific functionalities becoming unusable and negatively affecting user experience. Therefore, comprehensive testing across all major browsers is crucial, although this significantly increases the testing workload.
Modern users access web applications on a variety of devices, including smartphones, tablets, laptops, and desktop computers. Each device has different screen sizes, resolutions, and hardware capabilities, which can affect the application's layout and responsiveness. Xanthopoulos and Xinogalos (2013)Designing responsive web applications requires consideration of the characteristics of different devices to ensure that the application displays well on all of them. Testing and optimizing for each device individually not only consumes a lot of time but can also lead to unforeseen issues.
Web applications need to run on multiple operating systems, such as Windows, macOS, Linux, iOS, and Android. The network settings, security policies, and file handling methods of different operating systems can lead to varied functional performance of the application. For example, certain operating systems may restrict access to specific APIs or limit file system operations, which can impact some features of the application.

2.1.3 Complexity of security testing

The attack surface of web applications is often quite extensive, comprising multiple components such as the front-end user interface, back-end servers, databases, and APIs. Each of these components has its specific potential vulnerabilities that attackers may exploit. For instance, attackers can perform SQL injection attacks through input fields, execute XSS attacks via web forms, or leverage CSRF vulnerabilities through forged requests.Aydos, Aldan, Cos,kun, and Soydan (2022) This necessitates that security testing covers all potential attack paths, which increases the complexity of the testing process.
Cybersecurity threats are continually evolving, with new attack methods and tools emerging frequently. Attackers are constantly searching for new ways to breach security measures, which means that security testing for web applications must be continually updated to address these emerging threats. This requires security testing teams not only to monitor current security vulnerabilities and attack patterns but also to anticipate potential new threats to maintain effective protection.

2.2 What existing verification and validation techniques can we use to tackle the challenges found in answering RQ1?

2.2.1 Agile methodologies

Adopting agile methodologies, such as Scrum or Kanban, provides teams with a flexible framework that enables them to quickly adapt to changing requirements and environments. Unlike traditional waterfall models, agile methodologies encourage frequent interaction and feedback with stakeholders at every stage of the projectClutterbuck, Rowlands, and Seamons (2009). In agile development, projects are divided into multiple short iterations, commonly referred to as ”sprints,” each with clear goals and deliverables. Agile management tools such as JIRA, Trello, and Azure DevOps can be utilized to track tasks, manage workflows, and visualize progress. This approach allows the team to demonstrate work results at the end of each iteration, enabling timely adjustments to requirements and priorities, ensuring that the final product better meets user expectations and market changes.

At the start of each sprint, the team holds a planning meeting to determine sprint goals and priority tasks. Stakeholders should participate in discussions to ensure a shared understanding of the requirements. The team conducts brief daily stand-up meetings where members share what they are working on, the challenges they face, and the support they need.

At the end of each sprint, the team holds a review meeting with stakeholders to showcase completed features. Stakeholders can provide feedback, helping the team adjust the direction of future iterations. Following the review, the team conducts a retrospective meeting to discuss the successes and challenges of the current iteration. Ge, Paige, Polack, Chivers, and Brooke (2006)Team members collaboratively reflect on ways to improve processes for the next iteration.

2.2.2 Automated cross-browser testing tools

Using automated testing tools, such as Selenium, can significantly simplify the process of testing web applications across different browsers and devices. These tools simulate various operating environments (such as different browser versions, device types, and operating systems) to achieve consistent and efficient testing, thereby eliminating the burdensome manual testing work and improving test coverage and efficiency.

Based on the key functionalities and user flows of the application, automated test scripts should be written to cover important scenarios across major browsers (such as Chrome, Firefox, Safari, and Edge) and various devices (like smartphones, tablets, and desktop computers). Utilizing Selenium’s API and capabilities makes it straightforward to implement this.

The automated tests should be integrated into the Continuous Integration/Continuous Deployment (CI/CD) pipelineChoudhary, Versee, and Orso (n.d.). This means that whenever code is committed to the version control system, the automated tests will be triggered to run. By regularly executing these tests within the CI/CD process, teams can quickly identify and resolve cross-platform compatibility issues, ensuring that compatibility standards are met with every release.

As web applications and browsers continue to evolve, automated test scripts need to be maintained and updated regularly to accommodate new features and changes. Development teams should periodically review the test coverage to ensure that all new functionalities and alterations are appropriately tested.

2.2.3 Security testing framework

Implementing a comprehensive security testing framework, such as OWASP ZAP or Burp Suite, can effectively help identify vulnerabilities within web applications. These tools are designed to automatically detect common security issues, such as SQL injection, XSS, and CSRF, providing a more thorough assessment of the application’s security posture.

To implement this, security testing should be integrated into the development lifecycle, adopting a DevSecOps approach. This means that automated scans should be conducted regularly, alongside manual penetration testing, to ensure that security vulnerabilities are consistently identified and addressedPeroli, De Meo, Viganò, and Guardini (2018). By incorporating security testing into the Continuous Integration/Continuous Deployment (CI/CD) pipeline, teams can receive timely feedback regarding any vulnerabilities introduced by new code, enabling them to address security concerns proactively before deployment.

2.3 What are some recommendations for the remaining challenges not tackled by the existing techniques found in answering RQ2?

2.3.1 Machine learning for automated testing

Using machine learning to enhance test case generation is an effective strategy to tackle the challenges posed by Automated Cross-Browser Testing. Machine learning can analyze historical defect data to identify potential defect patterns and risk areas. By conducting an in-depth analysis of past project defect records, machine learning models can understand which functionalities are more prone to issues, thereby generating test cases targeted at these high-risk areas. For example, the model can identify specific modules that frequently failed in previous versions, guiding the testing team to apply stricter testing to these modules.

Machine learning can also analyze user behavior data to gain insights into user operations in actual usage scenarios. By capturing user behavior characteristics and usage habits, machine learning models can generate more targeted test cases, ensuring that testing scenarios align closely with real usage situationsChen and Chau (2004). This data-driven approach not only enhances the relevance of testing but also improves the effectiveness of test cases, better reflecting the true needs of users.

Additionally, the adaptive capabilities of machine learning allow for the continuous optimization of generated test cases over time. The model can adjust its generation strategies based on newly collected data, ensuring that test cases consistently meet current requirements and adapt to changing environments. This flexibility is particularly important in fast-paced development cycles, as it can reduce unnecessary repetitive work resulting from changing requirements.

2.3.2 Generative adversarial networks for security testing

GANs (Generative Adversarial Networks) are a form of deep learning technique that can be used to generate highly realistic data or patterns. In the context of security testing, GANs can simulate advanced and sophisticated attack patterns that go beyond typical known vulnerabilities. Unlike predefined test cases that focus on known vulnerabilities, GANs can be trained to create entirely new types of attack scenarios. This allows for the discovery of hidden or non-obvious vulnerabilities that static rule-based systems or signature-based detection tools might miss. For instance, GANs can learn to evade security measures by generating novel inputs designed to exploit potential weaknesses in code logic or system behavior.

Cyber threats constantly evolve as attackers innovate new methods to bypass defenses. GANs can be continuously trained on updated data, such as real-world attack patterns and new threat reports, allowing them to simulate emerging and zero-day attack techniques. By doing so, they provide a dynamic and evolving security testing framework, which traditional static testing tools lackChowdhary, Jha, and Zhao (2023). This adaptability makes them particularly valuable in environments where security needs to keep up with fast-evolving threats.

Standard security tools often struggle to detect multi-step attacks that may involve a sequence of actions over time, such as an attacker gaining low-level access and gradually escalating their privileges. GANs can be trained to simulate complex, multi-stage attacks, mimicking how an attacker might infiltrate and compromise systems in stages. This allows security teams to better anticipate and mitigate risks posed by sophisticated adversaries who execute advanced persistent threats (APTs).

3 Conclusion

In this paper, I focus on the major challenges in the verification and validation (V&V) of web application systems, particularly focusing on issues related to changing requirements, cross-platform compatibility, and security testing. The inherent complexity and evolving nature of web technologies make it difficult to ensure accurate validation and effective verification across various components of the system.

To address these issues, I reviewed existing V&V techniques. Agile methodologies provide teams with flexibility to adapt to frequent requirement changes through iterative feedback loops. Automated cross-browser testing tools like Selenium help ensure consistent user experiences across multiple platforms, improving the verification process. Security testing frameworks, such as OWASP ZAP and Burp Suite, focus on identifying known vulnerabilities, strengthening the validation of system security.

Nevertheless, some challenges remain unresolved, particularly those tied to the fast-paced evolution of requirements and the complexity of modern cyber threats. Machine learning offers a promising approach to enhance V&V by automating test case generation based on historical defect patterns and user behavior data. This enables more accurate validation by focusing on high-risk areas and aligning testing scenarios with real-world usage. Similarly, Generative Adversarial Networks (GANs) can significantly improve security validation by simulating advanced, novel attack patterns beyond the scope of traditional security tools. GANs dynamically create new attack scenarios, enabling the detection of hidden vulnerabilities, which enhances both the verification of system robustness and the validation of security measures.

In conclusion, while traditional V&V techniques provide a solid foundation for web application testing, the integration of advanced AI-driven methods like machine learning and GANs offers enhanced adaptability, accuracy, and comprehensiveness. These technologies strengthen the overall V&V process by addressing the complexities of evolving requirements and security challenges, ensuring higher-quality and more secure web applications in today's dynamic development landscape.

SWE2209518.docx

ORIGINALITY REPORT

7 %	5%	2%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	mdpi-res.com Internet Source	2%
2	David Lowe. "Web system requirements: an overview", Requirements Engineering, 2003 Publication	1 %
3	sandbox.hackerrank.com Internet Source	1 %
4	fastercapital.com Internet Source	1 %
5	www.moe.gov.cn Internet Source	1 %
6	aiforsocialgood.ca Internet Source	<1 %
7	repository.uel.ac.uk Internet Source	<1 %
8	Yusop, Norazlin Binte. "Issue Resolution and Scope Clarification in Web Systems Development: A Qualitative Study", University of Technology Sydney (Australia), 2024 Publication	<1 %

9	technewsdaily.com Internet Source	<1 %

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		

APPENDIX A

MARKING RUBRICS

Component Title	**Assignment**					
Criteria	**Score and Descriptors**				**Weight (%)**	**Marks (Score x Weight)**
	Excellent (5)	**Good (4)**	**Average (3)**	**Need Improvement (1-2)**		
General	The topic was very well-researched. The factual points clearly supported the answer to their research question. Statements were supported by quality research based on their articles. Clearly demonstrated in-depth knowledge of this topic.	The topic was good and acceptable. The points are acceptable and supported the answer to their research question. Statements were supported by acceptable quality research based on their articles. Clearly demonstrated knowledge of this topic.	The topic was average. The points are averagely acceptable that the students' answer to their research question. Statements were supported by average quality research based on their articles. Averagely demonstrated knowledge of this topic.	The topic was not researched in depth. The research evidences were not used help answer the research question or was applied inaccurately. The knowledge of this topic seemed very surface-level, indicating a lack of thoughtfulness and understanding.	**4**	
Integration of Knowledge	The paper demonstrates that the author fully understands and has applied concepts learned in the course. Concepts are integrated into the writer's own insights. The writer provides concluding remarks that show analysis and synthesis of ideas.	The paper demonstrates that the author, for the most part, understands and has applied concepts learned in the course. Some of the conclusions, however, are not supported in the body of the paper.	The paper demonstrates that the author, to a certain extent, understands and has applied concepts learned in the course.	The paper does not demonstrate that the author has fully understood and applied concepts learned in the course	**4**	
Depth of discussion	In-depth discussion & elaboration in all sections of the paper.	In-depth discussion & elaboration in most sections of the paper.	Discussion & elaboration in most sections of the paper is average.	Brief discussion in only a few sections.	4	
Organization	Paper is well-organized. Ideas are arranged logically, flow smoothly, with a strong progression of thought from paragraph to paragraph connecting to the central purpose or position statement. Includes all required components.	Paper is organized for the most part. Ideas are arranged logically and usually linked clearly from paragraph to paragraph connecting to the central purpose or position statement. Includes required components for the most part.	Paper is organized in general, although occasionally ideas from paragraph to paragraph may not make sense to the central purpose or position statement and/or be clear as a whole and/or may be lacking a required component.	Paper lacks logical organization and impedes readers' comprehension of ideas. Central purpose or position statement is rarely evident from paragraph to paragraph, and/or is missing required components.	4	
References & Format	Adequate citation or references and are effectively used not accordingly to APA style. Paper conforms to the journal paper format.	Citation or references serve a purpose and are generally well-used accordingly to APA style. Paper almost fully complies with the journal paper format but there is some inconsistency.	Cites some data obtained from other sources. Citation style is either inconsistent or incorrect. Contained a few references but no citation accordingly to APA style. Paper does not fully comply with the writing format.	Paper does not comply with the journal paper format. No reference and citation. The references and citation not accordingly to APA style.	4	
					TOTAL	**/100**

Note to students: Print out and attach this appendix together with the submission of coursework.