Web applications testing techniques: a systematic mapping study

Samer Hanna*

Department of Software Engineering,
Faculty of Information Technology,
Philadelphia University, Jordan
Email: shanna@philadelphia.edu.jo
*Corresponding author

Amro Al-Said Ahmad

School of Computing and Mathematics,
University of Keele, UK
Email: a.al-said.ahmad@keele.ac.uk

Abstract: Due to the importance of web application testing techniques for detecting faults and assessing quality attributes, many research papers were published in this field. For this reason, it became essential to analyse, classify and summarise the research in the field. To achieve this goal, this research conducted a systematic mapping study on 98 research papers in the field of web applications testing published between 2008 and 2021. The results showed that the most commonly used web applications testing techniques in literature are model-based testing and security testing. Besides, the most commonly used models in model-based testing are finite-state machines. The most targeted vulnerability in security testing is SQL injection. Test automation is the most targeted testing goal in both model-based and security testing. For other web applications testing techniques, the main goals of testing were test automation, test coverage, and assessing security quality attributes.

Keywords: web applications testing techniques; systematic mapping study; SMS; testing purposes; model-based testing; security testing; test automation; test coverage.

Reference to this paper should be made as follows: Hanna, S. and Al-Said Ahmad, A. (xxxx) 'Web applications testing techniques: a systematic mapping study', *Int. J. Web Engineering and Technology*, Vol. X, No. Y, pp.xxx–xxx.

Biographical notes: Samer Hanna is an Associate Professor and currently the Head of both the Software Engineering and the Web Engineering Departments at the Faculty of Information Technology at Philadelphia University in Jordan. He obtained his PhD in Computer Science (Software Engineering) in 2008 from Durham University in England. His current research interests include: assessing the security and dependability of web applications and mobile applications using different testing techniques, and enhancing software engineering curricula by comparing courses with the industrial needs.

Amro Al-Said Ahmad is a Lecturer in Computer Science at the School of Computing and Mathematics at Keele University, UK. He completed his PhD in Cloud Computing at the School of Computing and Mathematics of Keele

University, UK. His main research interest is focused on the technical scalability of cloud-based services and the delivery of such services from a technical perspective. His other research interests address the research areas of software engineering, software testing, software metrics, and agile methodology. In relation to web engineering, he is interested in the testing and evaluation of web/cloud systems and applications.

1 Introduction

Due to the importance of web applications in many domains, such as healthcare and banking, it is crucial to test these applications to assess their quality attributes, such as security, reliability, and performance. The field of web applications testing had thrived in the past years, and numerous research studies covering many techniques, methods, or approaches for testing web applications were published in the main journals and conferences related to software testing and software engineering in general. For this reason, it became important to analyse, categorise and classify published research papers to help researchers and practitioners to identify the main testing techniques used with web applications and the gaps in the research in this field. This allows researchers to focus on the areas that are not receiving the required attention in this field.

The main goal of this research is to analyse and classify the studies in the field of web applications testing to find out the main testing techniques used for test data generation for web applications and the main testing targets or goals for these techniques or approaches.

Systematic reviews and mapping studies aim to collect evidences on specific research questions (RQs) or subjects of interest (Kitchenham et al., 2015). To reach the goal of analysing web application testing techniques, this research conducted a systematic mapping study (SMS) that addresses the testing techniques, approaches or methods used with web applications and the goals or purposes of each of these testing techniques.

To conduct the SMS in this paper, 98 research papers published between 2008 and 2021 in the field of web applications testing were reviewed, analysed and classified based on the testing technique used and the testing goal in each paper.

The research papers in the field of web applications testing have proposed different testing techniques, approaches or methods for assessing a variety of quality attributes of web applications. Examples of the testing techniques used by researchers in this field are model-based testing, security testing, mutation testing, and performance testing.

The SMS results revealed that the most commonly used web applications testing techniques are model-based and security testing. For this reason, these testing techniques are given more attention in this paper by categorising the research papers related to the model-based testing techniques according to the model used to generate test data and categorising the research papers related to security testing according to the targeted vulnerability.

The result also showed that the most commonly used models in model-based testing are finite state machines (FSM) and the main investigated vulnerability in security testing is structured query language (SQL) injection. It was also concluded that test automation is the most commonly targeted testing goal for model-based testing and security testing.

The testing goals of all of the reviewed papers in the field of web applications testing are specified in Section 11.

Model-based web applications testing (MWAT) (Section 8) have special importance in web applications since these applications are complicated and built using many different programming languages and technologies, such as HTML, CSS, JavaScript, XML, PHP, ASP.NET, Java, and Python. Therefore, modelling web applications will make them more understandable by testers and will simplify the process of test data generation based on the resulted models. Different models or graphs, such as FSM and page flow graph (PFG), can be used to generate test cases for web applications.

In security-based web applications testing (SWAT) (Section 7), test case generation is based on the specific vulnerability targeted by the testing, such as SQL injection or cross-site scripting (XSS). In such testing, HTTP requests that include input values related to the targeted vulnerability are sent to the web application under test. The HTTP response from this application is analysed to determine whether such malicious inputs were detected by the application under test or not. In this case, test data include inputs that are related to the specific vulnerability to be detected. On the other hand, if we consider model-based testing as another example, test data in this case can be generated based on the paths specified by the model used in testing. Therefore, test data generation is based on the testing technique that is used to test a web application, which, in turn, is based on the targeted quality attribute specified by the requirements for this application.

This research provides a classification schema (Section 6) for the research papers in the field of web applications testing techniques to achieve the following main targets.

a To help researchers and practitioners in this field by providing an overview of the current trends in the field.

b To help researchers in this field to discover the areas that had been thoroughly studied by researchers and the areas that still require further investigation.

This will help researchers to focus on the areas of web applications testing that had not been fully explored and hence will lead to covering the different aspects or attributes of web applications by software testing.

The inclusion criteria for the papers in this SMS are as follows: for a paper to be included in this SMS, it must:

a Tackle a specific testing technique, approach or method for web applications testing.

b It must specify the goal or purpose of testing, such as assessing a certain quality attribute or covering all the inputs of a web application.

The main quality attributes found in literature are security, performance, and usability. Inputs to a web application can be login, personal, financial, or any other input depending on the purpose of the application under test.

The contributions of this research are:

- Building a classification schema for the research in the field of web applications testing that can be used to categorise every research in this field.
- Identifying the current state-of-the-art and trends in the field of web applications testing techniques and testing goals.

- Determining the mostly targeted testing goals for web applications testing techniques.

The remainder of this paper is organised as follows; Section 2 outlines the background of the field of web applications testing. Section 3 describes the related work to this paper. Section 4 describes the research method followed by this research. Section 5 classifies reviewed studies according to the publication year. Section 6 provides a classification schema for the research in the field of web applications testing techniques. Section 7 provides details about security-based web application testing techniques together with the targeted vulnerabilities, Section 8 presents details about model-based web application testing techniques together with the models used for test case generation, Section 9 explains the other testing techniques used in web applications testing. Section 10 provides a classification of the primary studies based on the testing technique or approach used, while Section 11 classifies the primary studies based on testing goals. Section 12 classifies the studies based on the evaluation method used in each study. Finally, Section 13 presents the conclusion and discusses future work.

2 Background

Because this study aims at analysing the research in the field of web applications testing and testing techniques, traditional software testing will be discussed in Subsection 2.1 and web applications testing will be discussed in Subsection 2.2. The aim is to describe these terms in detail and to explain the differences between testing traditional applications versus testing web applications.

2.1 Software testing

The software development life cycle includes the following main phases: requirements specification, architecture, design, coding, testing, and maintenance. Each of these phases has its own process and activities. Testing is considered as one of the most important phases of a software application's development life cycle since it consumes approximately 40% of the total time needed to build an application (Li et al., 2014).

Software testing can be used for detecting faults and assessing quality attributes, such as security, performance, and reliability. It is important to detect faults in a system as early as possible because the cost of correcting these faults will be much higher after the delivery of the system to the client. Besides, it is important to check that a system can accomplish all the required functionality and in a reliable, dependable and secure manner.

Software testing involves generating test cases that include both test data and the expected output. The software application is executed with test data to check that the actual output obtained when executing the application is equal to the expected output specified by each test case.

Software testing can be time-consuming and labour-intensive when performed manually; therefore, researchers and practitioners are doing their best to find approaches and tools that can be used to automate or at least semi-automate this process.

2.2 Web applications testing

Testing web applications is essential since we rely on these applications in many areas of our daily lives. Testing web applications can be used for ensuring that these applications fulfil the required functionality and are secure, well-performed, and dependable.

As web applications have peculiarities that did not exist before the inception of HTML in late 1989, testing these applications is different from software testing of traditional applications. One of these peculiarities is that more than one user can use a web application simultaneously and at any time, besides, any user around the world can insert any input into a web application. Traditional testing techniques must be modified to cope with such peculiarities of web applications.

A web application testing includes:

a Generating test data based on the different web applications testing techniques, such as model-based testing and security testing.

b Sending an HTTP request to the web application under test using the different test data, and comparing the HTTP response of this application with the expected response to each of the test data.

Web application testing techniques such as security testing (Section 7) and model-based testing (Section 8) can be used to assess different quality attributes of web applications. These techniques can have different goals, such as automating the testing process or covering all the paths of the web application under test, Section 11 describes all of the testing goals of the reviewed research in this SMS. Determining the required quality or other test goals depends on the web application's specified requirements.

Test data can be determined based on either the targeted quality attributes such as security or performance, or the test goal, or targets such as reducing test time or increasing test coverage. For example, suppose the targeted quality is security. In this case, the security-based testing technique must be performed to achieve this target. The test data, in this case, would be the inputs related to the top vulnerabilities of web applications, such as SQL injection and XSS. However, suppose the targeted quality is performance, in that case, the testing technique related to assessing the performance quality attribute must be performed, and the test data would be concluded based on increasing the load of users to determine the effect of this on performance. In short, web application's different qualities or test targets require different test data and testing techniques.

Web application testing techniques such as security and model-based testing can be used to assess different quality attributes of web applications. These techniques can have different goals, such as automating the testing process or covering all the paths of the web application under test. Determining the required quality or other test goals depends on the web application's specified requirements.

Because of the great importance of web applications, many research studies had been published in the field of testing these applications. The main objective of this research is to analyse and classify these studies in order to help researchers to find out the current trends in the field and the areas that require further investigation.

The research papers in the field are categorised into three main categories: model-based testing, security testing, and other types of testing. It must be emphasised here that these categories are not independent, for example, model-based testing can be

used for testing the security quality attribute, besides, many of the testing techniques in the 'other types of testing' category are also used for testing security and also these techniques can use models in the testing process.

In the case of model-based testing of web applications, research papers are classified according to the model used for test data generation, while the research papers in the field of web applications security testing are classified according to the targeted vulnerability.

More than one testing technique can be used in combination to assess a certain quality attribute of web applications. For example, model-based testing can be used with security testing, such as SQL injection testing, to model the SQL injection attack and describe how this attack can be performed. Another example is using mutation testing with security testing.

3 Related work

Because of the importance of web applications testing, it is crucial to conduct mapping studies and literature reviews for the research in this field in order to identify the current trends in the field and highlight the areas that require further investigation by researchers and practitioners.

Below are examples of the mostly related studies in this field of SMS or review for web application testing given that these studies are considered secondary research papers to this SMS.

The systematic review conducted by Garousi et al. (2013) reviewed and categorised the body of knowledge related to web applications testing techniques. The study analysed 79 papers in this field that were published between the 2000 and 2011. It explained in detail emerging trends in the field and the areas that require more attention from the research community. Doğan et al. (2014) conducted a systematic literature review of 95 studies in the area of web applications testing that were published between 2000 and 2013. Dadkhah et al. (2020) presented a systematic literature review of semantic web enabled testing, in which semantic-based technologies are used.

Li et al. (2014) presented a survey of web application testing advances during two decades. They specified each web application testing technique's goal, target, input/ output, and stopping criteria. They also discussed the strengths and weaknesses of these techniques. Van Deursen et al. (2015) reviewed the research in the field of crawling and testing of web applications during a five-year period. They identified several future research directions in this field.

Some studies conducted reviews and SMS on a specific type of web application testing, such as security testing; an example of such studies is Aydos et al. (2021).

Table 1 summarises the related secondary studies by specifying the type of each study, the number of reviewed papers, and the covered years.

The research in this paper is different in that it analyses the relations between web applications testing techniques in literature and the most commonly used testing purposes targeted in each of these techniques between 2008 and 2021. The research classifies the reviewed papers in the field of web applications testing, not only based on the used testing technique(s) in each paper but also based on the goal or target of testing.

To enhance the credibility of our study, the primary studies discussed in this study are compared with the related secondary studies. All of the primary studies were included based on our inclusion criteria or excluded based on our exclusion criteria.

Table 1 Secondary research papers in the field of web applications testing' literature reviews, mapping studies, and reviews

Study type	*Field*	*Number of reviewed papers*	*Covered years*	*Reference*
SMS	Web application testing	79	2000 to 2011	Garousi et al. (2013)
Systematic literature review (SLR)	Web applications testing	95	2000 to 2013	Doğan et al. (2014)
Survey	Web applications testing	Not specified	1993 to 2013	Li et al. (2014)
SMS	Security testing of web applications	80	2005 to 2020	Aydos et al. (2021)
SLR	Semantic web enabled software testing	52	2005 to 2019	Dadkhah et al. (2020)
Review	Crawling and testing of web applications	Not specified	Not specified	Van Deursen et al. (2015)
Review	Web applications testing tools and techniques	Not specified	Not specified	Lakshmi and Mallika (2017)

There are many research papers that conducted literature reviews and mapping studies on web services and cloud services which are very related to web applications, example of such research is Al-Said Ahmad et al. (2017) who conducted an SMS on the software testing techniques that are used with cloud-based services and applications. The research identified the common testing techniques and directions in the field of cloud testing methods.

4 Research method

This section describes the method that was used in this paper to conduct a SMS in the field of web applications testing techniques and testing goals or purposes. The method is based on the systematic mapping methodology discussed in Kitchenham et al. (2015). The main activities of this SMS are shown in Figure 1.

Figure 1 The mapping study process (see online version for colours)

The activities in Figure 1 will be discussed in more details in Subsections 4.1, 4.2 and 4.3.

4.1 Definition of RQs

The main goal of this research is to identify and classify the current web applications testing techniques used and how the testing techniques are related to the testing purposes. To achieve this goal, this research addresses the following RQs in each primary study.

RQ1 What is the testing technique, method or approach used to test web applications?

Web applications testing can be accomplished using different testing techniques, such as model-based testing, penetration testing, fuzz testing and mutation testing. This RQ specifies the testing technique used in a certain research paper. According to Li et al. (2014), the main testing techniques that are used to assess the functionality and quality of web applications are model- and graph-based testing, mutation testing, search-based testing, scanning and crawling, random testing, fuzz testing, concolic testing, and user session-based testing. Section 10 categorises the primary studies in this SMS according to the technique, method, or approach used in testing.

There can be different categories of the same testing technique; for example, for model-based testing, different models can be created for the web application under test, such as class diagram, FSM, data dependencies, and control dependencies. Security testing techniques can be crawler-based, scanner-based or penetration-based. This RQ aims to identify both the main testing technique used for test data generation and the category of this technique, for example, model-based testing using FSM.

RQ2 What is the goal or purpose of testing?

This RQ specifies the goal, target or purpose of the testing technique specified by RQ1, such as test coverage, test automation, assessing security or performance, and improving the fault detection rate.

The main testing goals or purposes considered by the papers in this mapping study are: test automation, test coverage, obtaining higher vulnerabilities exploitation rate, reducing false positive, reducing false negative, detecting security vulnerabilities, reducing test time or effort, reducing test suite or number of test cases, evaluation of sanitisation process or user input validation, assessing usability, assessing performance, test suite prioritisation, maximising test suite diversity, failure mitigation, examining database updates, testing the navigation behaviour, testing dynamic web applications, and detecting duplicate web pages. A research paper can target one or more of these testing goals; for example, a paper can target assessing performance quality and test automation. Section 11 explains testing goals in detail.

For some web applications testing techniques, such as security or performance testing, the goal of testing is clear. For example, in security testing or crawling and scanning-based testing, the targeted quality attribute is security. This goal or quality can be assessed by checking whether a web application under test accepts inputs related to injection vulnerabilities such as SQL injection, XSS and buffer overflow. However, for other web applications testing techniques, different testing goals can be targeted such as test automation and/or increasing the application coverage and/or reducing the test cost. Some testing techniques and hence research papers can have more than one goal, such as assessing the security quality attribute and increasing coverage. This research analyses

and categorises the research papers in the field of web applications testing according to the testing goals of the testing technique(s) adopted in each paper or study. Section 11 categorises the primary studies according to the testing purpose.

RQ3 What is the method of evaluation used by a research paper?

This RQ specifies the approach or method that was used by a research paper for the purpose of evaluation. Studies in the field of web applications testing can use different approaches for evaluating the proposed testing approach, such as case studies, experiments on real or experimental web applications, or prototypes and tools. Section 12 categorises the primary according to the evaluation method.

RQ4 What is the venue and year of the research paper?

This RQ determines the venue of the paper such as conference location, year, etc.

Answering the RQs above is important to specifying the recent techniques, approaches or trends adopted by researchers in the field of web applications testing techniques and goals. This can help researchers to find out the current trends in this field and direct their research to the less covered areas accordingly.

4.2 Search process

The search for the research papers in the field of web applications testing techniques and purposes was an automatic search that was performed using the following libraries (research databases):

- IEEE Xplore
- ScienceDirect
- SpringerLink
- Google Scholar
- ACM digital library
- CiteSeerX.

The targeted publication period was 2008 to 2021. An automatic search had been performed in the above libraries for both conference papers and journal articles. For each selected primary paper, a search was conducted in the references of that paper. This process is called snowballing. The search strings used in this research are:

- Web or web application or applications and testing, evaluation, validation, test case or data generation.
- AJAX or JavaScript and test or testing or test case or data generation.
- SQL injection or XSS and test or testing or test case or data generation.
- Web application or applications and user input validation or test case or data generation.

The first search string means that for a paper to be included in the mapping study the title of this paper must contain any of the words in the list (web or web application or applications) and also any of the words in the list (testing or test or evaluation or validation or test case or data generation). Similar discussion can be made for the other search strings.

4.3 Paper inclusion and exclusion criteria

For a research paper to be included in the mapping study, it must be published from the year 2008 to 2021. It must have at least five pages, it must specify a testing technique for web applications, clearly specify the testing purpose or target, and include a method for evaluating the proposed testing approach, such as an experiment or case study. Any paper that does not satisfy any of the above conditions was excluded from the mapping study.

4.4 Data extraction

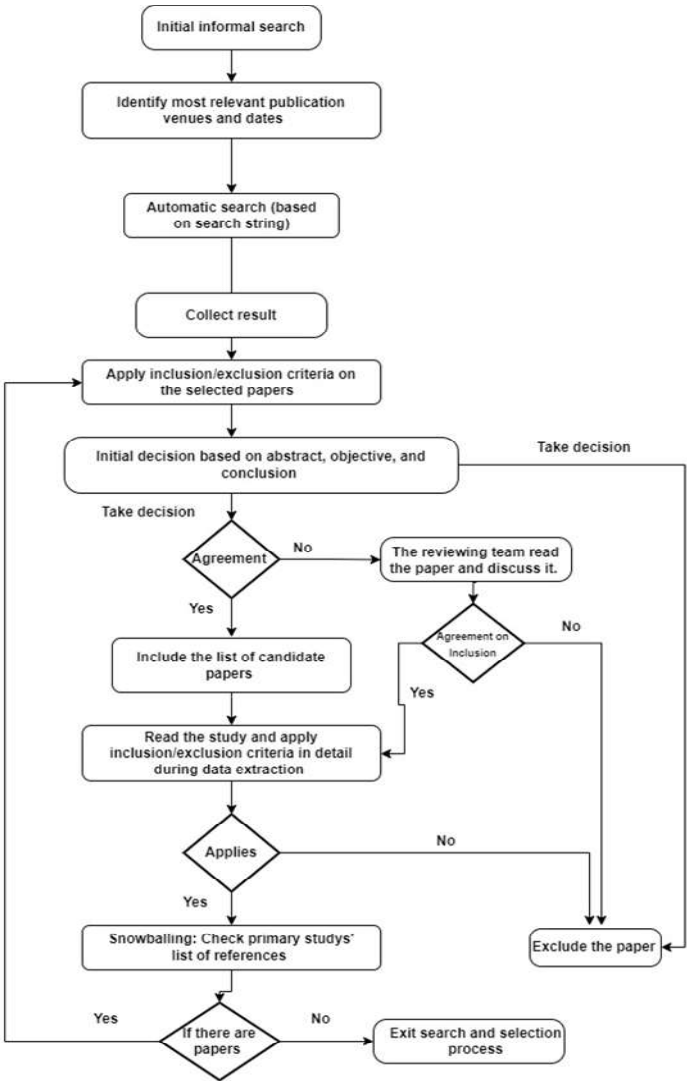
Data extraction aims to produce well-designed systematic mapping by clustering the primary papers in one area into mapping categories (Petersen et al., 2008). The data had been extracted to answer the RQs outlined in Subsection 4.1. During the data extraction stage, the full text of each paper was read, and the extracted data were stored in an independent spreadsheet. Both standard information and specific information extracted from each study are listed in Table 2.

Table 2 Data extraction values

Data item	*Description/value*	*Relevant RQ*
Study ID	Study ID	–
Paper title	Article title	–
Authors	Authors' names	–
Year	Year of publication	RQ4
Venue	Publication source and type (journal, conference)	RQ4
Testing methods	Testing methods, approaches, or techniques used in the studies	RQ1
Goals	The testing goal, purpose or testing requirement	RQ2
Evaluation methods	The method of evaluation implemented in the studies	RQ3

Figure 2 illustrates the methodology adopted in this study; first, we inducted an informal search to identify the most relevant libraries; based on this, we identified the years (2008–2021) for the targeted primary studies. Following that, we create the search string presented in Subsection 4.2, which allows us to collect primary studies in the targeted area. Next, the inclusion and exclusion criteria (see Subsection 4.3) have been applied to the collected study, a primary scan based on the papers abstract, objectives, and conclusion. Finally, each selected primary study's reference list has been examined to locate relevant papers.

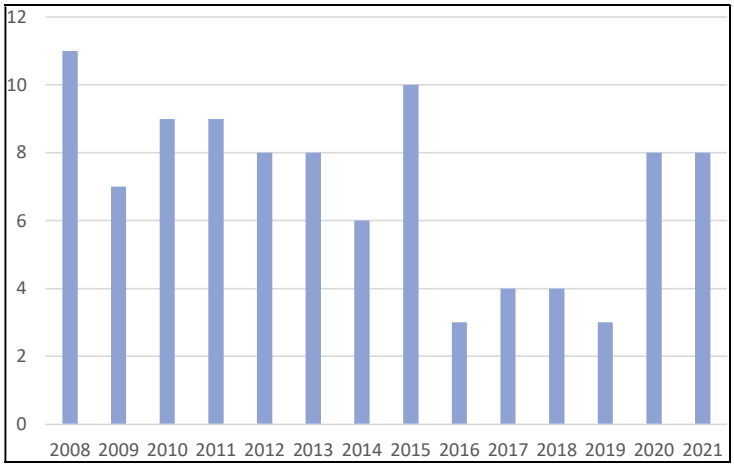
Figure 2 The methodology of searching for relevant studies in web applications testing



5 Distribution of studies according to the publication year

We identified 98 studies in the field of web applications testing published between 2008 and 2021. The search was conducted using the method described in Section 4 and fully described in Figure 2. Among the 98 studies, 50 studies were from conference proceedings, and 48 from journals. Figure 3 shows the distribution of primary studies by year of publication.

Figure 3 Distribution of primary studies by year of publication (see online version for colours)



It is noticed in Figure 3 that the years 2008 and 2015 are the years with the highest number of publications in the field of web application testing according to the reviewed papers in this SMS.

6 Classification of web application testing techniques

The extracted studies in the field of web application testing techniques were analysed and structured to answer the RQs in Subsection 4.1. In this section, we analyse the primary studies and the extracted data related to the RQs. After reviewing the research studies it was concluded that these studies could be classified according to the schema in Figure 4.

The main categories in the schema are security-based testing, model-based testing, and other types of testing. These specific categories were determined after analysing the research in the field and concluding that most of this research is related to either SWAT or MWAT.

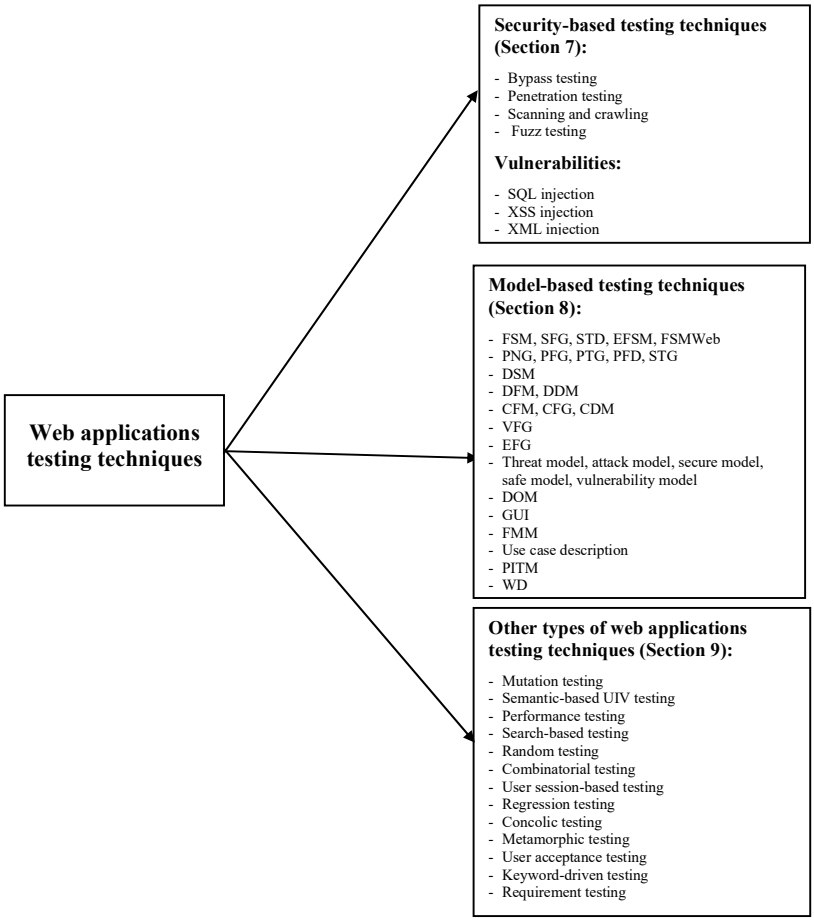
The schema in Figure 4 can help researchers in the field of web application testing to conclude the main testing techniques used with web applications and how these techniques can be categorised. It must be emphasised though that the schema is resulted from analysing the 98 primary studies only.

The security-based testing techniques in Figure 4 are classified according to the technique that is used for assessing the security of a web application and the vulnerability targeted by this technique. On the other hand, the model-based testing techniques are classified according to the model or graph that is used for test data generation.

The abbreviations in Figure 4 are as follows, FSMs, state flow graph (SFG), state transition diagram (STD), extended FSM (EFSM), FSM for web (FSMWeb), page navigation graph (PNG), PFG, page transition graph (PTG), page flow diagram (PFD), screen transition graph (STG), domain-specific model (DSM), data flow model (DFM), data dependency model (DDM), control flow model (CFM), control flow graph (CFG), control dependency model (CDM), validation flow graph (VFG), event flow graph (EFG), document object model (DOM), graphical user interface (GUI), event sequence

graph (ESG), failure mitigation model (FMM), platform independent test model (PITM), sequence diagrams (SD), and web diagram (WD).

Figure 4 Classification schema of web applications testing techniques



Section 7 will provide details about SWAT techniques together with the targeted vulnerabilities, Section 8 will present details about MWAT techniques together with the models used in test case generation, and Section 9 will explain the other testing techniques used in web applications testing. Section 10 will provide a classification of the primary studies based on the testing approach used, while Section 11 will classify the primary studies based on testing goals.

7 SWAT techniques and targeted vulnerabilities

Since most web applications do not take security threats into consideration (Li et al., 2010a), web application security testing is of great importance. In security-based testing

techniques, the aim is to determine whether a web application under test can handle invalid, malicious, perturbed or unsanitised inputs related to known vulnerabilities, such as SQL injection and XSS.

7.1 Testing techniques

The security-based testing techniques depicted in the concluded classification schema in Figure 4, are discussed below.

7.1.1 Bypass testing

This testing is based on sending invalid data to a web application under test, bypassing the client-side user input validations to evaluate this application's behaviour when the user input is invalid (Offutt et al., 2014).

7.1.2 Penetration testing

This testing is based on simulating an attack on a web application under test to detect vulnerabilities in this application (Halfond et al., 2011). Similar to bypass testing, invalid or malicious input is sent to the application, and the application's response is analysed to ascertain whether the attack was successful (Halfond et al., 2011).

7.1.3 Scanning and crawling

In some research papers, security testing is called scanning and crawling since scanning and crawling techniques are mainly used to assess or test the security of web applications (Li et al., 2014). In such testing techniques (similar to bypass testing and penetration testing), the web application under test is injected with invalid, malicious, perturbed or unsanitised input that can result in unauthorised modification of a database (Li et al., 2014). The differences between crawling and scanning are described in Li et al. (2010b). Crawling can also be used to achieve high coverage of the web application under test, such as in Dallmeier et al. (2013).

7.1.4 Fuzz testing

User input validation testing can also be called fuzz testing (Li et al., 2014). This testing is also based on sending invalid inputs as test data to the web application under test to determine whether this application includes the required validation to reject such inputs. If the invalid inputs belong to the malicious inputs related to the well-known web applications vulnerabilities, such as SQL injection, this type of testing will also belong to security testing.

7.2 Targeted vulnerabilities in security-based testing techniques

The vulnerabilities detected by the reviewed web application security testing techniques are briefly explained below.

7.2.1 Lack of user input validation vulnerability

Lack of user input validation vulnerability (Offutt et al., 2014) occurs when a web application accepts input from a user without checking whether this input is valid or not. Therefore, web applications must include client-side and server-side validations that can reject invalid, unsanitised, manipulated or perturbed user inputs to ensure security quality attributes.

Examples of user input validations are:

- Ensure that the user inserted name length does not exceed 20 characters.
- Ensure that the user inserted phone number is ten digits.
- Ensure that the user e-mail contains the @ symbol.
- Ensure that an input is not empty.
- Ensure that the user inserted age is between 18 and 120.

User input validation in web applications can be implemented using the methods outlined below.

a HTML attributes that can be used with input-related HTML elements, for example, 'input' and 'select', are used to receive input from a user. Examples of such attributes are:
   - '*Required*' is used to ensure that an input is not empty.
   - '*Maxlength*', is used to specify the maximum number of characters an input can have. This attribute can be used for the name validation in the above example.
   - '*Pattern*', which is used to ensure that input must follow a certain pattern or regular expression. For example, this attribute can be used for e-mail validation in the above example.

b Client-side JavaScript or other scripting languages:

   Scripting languages can manipulate the HTML DOM tree and extract the input inserted by a user in an HTML form. Subsequently, validation is applied to this input, such as ensuring it is within a certain range, as in the age example above or any other client-side user input validations.

c Server-side languages:

   The previous examples of user input validations can also be accomplished at the server side using languages such as PHP, ASP.NET, and JSP.

The absence of user input validation at the client side or server side is considered the main vulnerability in web applications since the top vulnerabilities, such as SQL injection and XSS, result from invalidated user inputs. Scholte et al. (2012) reported that the absence of user input validation causes most vulnerabilities in web applications.

7.2.3 SQL injection vulnerability

SQL injection is one of the most devastating vulnerabilities that affect a business (Clarke, 2009). This vulnerability is among the top 10 vulnerabilities in web applications specified by the open web applications security project (OWASP) (Anon., 2020). SQL injection

vulnerability occurs when a user sends a crafted malicious input to a web application. This input causes illegal execution of an SQL query and thereby gains access to a database and performs many unauthorised actions, such as stealing sensitive data.

7.2.4 XSS injection vulnerability

XSS injection vulnerability is also caused by the absence or the inappropriateness of user input validation of a web application. Malicious users or hackers can inject an input that contains a JavaScript or HTML form into a web application that does not include a proper input validation. When a victim uses this vulnerable web application, the malicious user can disclose their data, such as cookies or any other sensitive data. Web applications must include validation to ensure that user input does not include a script to solve these likely problems.

7.2.5 XML injection vulnerability

This vulnerability happens when an XML document is manipulated by injecting malicious content into the document and then injecting this XML document as an input into an application (Jan et al., 2019). Similar to SQL injection and XSS, this vulnerability is also caused by the absence of user input validation. The aim of injecting a manipulated XML document is to compromise the web application under test itself or other applications that process this XML message. This vulnerability can cause major problems, such as denial-of-service (DoS) and data breaches (Jan et al., 2019).

8 MWAT technique

This technique is based on creating a model for the web application under test and then generating test data based on this model. The models that are used for web applications testing found in the reviewed papers are discussed below.

8.1 FSMs, SFG, STD, EFSM, or FSMWeb

FSM (Ran et al., 2009) or SFG (Qi et al., 2017) or STD are used to model a web applications behaviour without the details related to the code or implementation of this application. Such models or diagrams show how users navigate pages of a web application (Ran et al., 2009).

EFSM (Song et al., 2011) is used to formalise the navigation model of a web application. A web navigation model together with a web browser's interaction can be formally described using EFSM (Song and Miao, 2009).

FSMWeb models web inputs, navigation between web application pages, and behavioural characteristics (Boukhris et al., 2017).

FSM or the other related models help in exploring the web application under test by finding out all the states, pages, user interface elements, etc. that this application includes. Test cases can be generated afterwards to force the application to go from one state to another for the purpose of covering all the states of the applications. Qi et al. (2017) used STG to dynamically explore user interface elements and then generate test cases accordingly.

8.2 PNG, PFG, PTG, PFD, or STG

Navigation of a web application can be defined as the sequence of pages that a user visits to accomplish the required functionality (Sabharwal et al., 2013). Navigation models of web applications can be described as EFSM. Such models aim to specify both navigations among the pages of the modelled web application and the interaction of these pages with the browser (Song and Miao, 2009). In PFG (PNG, PTG or PFD) (Tkachuk and Rajan, 2011), the nodes of the graph represent a page of the modelled application, and the edges describe the navigation between the source and destination pages of the application (Polpong and Kansomkeat, 2015). Syntax models that are created based on PFG can also be used in web applications testing (Polpong and Kansomkeat, 2015). For Example, in STG, a screen of the modelled web application is represented as a node in the graph, in contrast, screen transition caused by a user's interaction, such as clicking a button, is described as an edge of the graph between the source and destination screen (Zhang and Tanno, 2015).

Detecting all the pages or screens of the web application under test and the navigation structure between these pages, can help software testers to make sure that all the pages or screens of the application had been tested by at least one test case (depending on the coverage criteria). Besides, test data can be used based on the inputs that can force the application under test to go for a certain page.

8.3 Domain-specific model

Domain-specific languages are based on EFSM (Törsel, 2013). DSM can be used to automatically generate test cases for web applications testing. The advantages of DSM notation are that it can be tailored depending on the requirement, making it more understandable than generic modelling notation (Törsel, 2013).

It should be noted that all the previous models that are used in MWAT are based on FSM. Therefore, generating test cases for web applications based on these models is accomplished based on exploring the different states of the web application under test to make sure that testing had covered all of the states of the application besides the transition between these states.

8.4 DFM or DDM

DFM or DDM describes how data are transferred from a source page in a web application to a target or destination page (Tung et al., 2010). Data dependencies for web applications occur when a variable is defined in a statement of a certain page in a web application and then used in a statement on another page of this application (Tung et al., 2010). Knowing the structure of the web application under test helps in automatically generating the test case for this application.

8.5 CFM, CFG or CDM

CFM, CFG or CDM is extracted from the source code of a web application. It is based on control statements that cause an application to navigate from a source page in this application to a target or destination page (Tung et al., 2010). Test cases are based on

inputs that can cause a control statement to produce a true or false output in order to test the different pages of the web application under test.

8.6 Validation flow graph

VFG (Liu and Tan, 2008) is a variant of CFG that gives a higher-level view of the UIV feature in a web application. Test coverage criteria can be concluded based on the input validation features determined by the model.

8.7 Event flow graph

EFG or event functional graph (Habibi and Mirian-Hosseinabadi, 2015) consists of nodes representing events and edges showing the order of occurrence of the two attached events.

8.9 Threat model, attack model, secure model, safe model, and vulnerability model

These models are all security related models that is the reason they were put in the same category. Each of these models will be explained below:

- *The threat model* describes how attacks can be performed against a web application under test. It describes how a malicious user may perform attacks to violate a security goal (Xu et al., 2012). All the attack paths for a web application under test can be generated from the threat model of this application (Xu et al., 2012).
- *Attack model* (Tian et al., 2012) used an attack mode to model SQL injection attacks using an attack tree to describe the attack behaviour. In this research, the attack models of SQL injection are used to describe the types and regularity of this attack.
- *Secure model* (Buchler et al., 2012) used a formal model for the specification of the web application under test. The model is considered secure in that it does not violate the specified security goals.

  Mallouli et al. (2008) used a formal language called nomad to specify or model security rules in a context that includes time constraints.
- *Safe model* can describe the structure of the response expected by a web application under test when the input inside the request is safe to input (Avancini and Ceccato, 2012). An attack occurs when an input causes the violation of the structure of the safe model.
- *Vulnerability test case generation model* (Lei et al., 2013) proposed a formalised SQL injection vulnerability test case generation model that is used to test different types of SQL injection. Tian et al. (2012) proposed a model for generating penetration test case inputs to detect SQL injection vulnerability.

In all of the previous security related models, test case are created based on inputs that resembles those used by malicious users and that can be concluded based on these models. The goal is to check whether the web application under test can handle such inputs or not.

8.10 Document object model

DOM is an application programming interface (API) for valid HTML documents (W3C, 2004). DOM defines the logical structure of an HTML document. DOM can access and manipulate the elements inside HTML documents of web applications (W3C, 2004).

Due to the interaction between DOM and JavaScript in a web application, DOM can generate test cases for JavaScript-based web applications (Mirshokraie et al., 2016). Test coverage is based on covering all the paths in a DOM tree related to an HTML page of a web application under test.

8.11 GUI test models

Testing web applications depending on a GUI test model is based on the fact that many web applications are built using a common user interface pattern (Nabuco and Paiva, 2014). Accordingly, tests of web applications with the same user pattern can be reused. Furthermore, test data can be generated from the modelled business processes (Heinecke et al., 2011).

8.12 Event sequence graph

This graph describes the path in a web application where input data is to be inserted. Therefore, it can be used for modelling and testing the interaction between a user and a web application under test (Krüger and Linschulte, 2012). Both valid and invalid input data constraints can be modelled using ESG. The test sequence is combined with input data in ESG. Krüger and Linschulte (2012) provide an example for creating ESG for a web application login page.

8.13 Failure mitigation model

FMM can be used to determine an application's test paths that can be affected by a failure, such as network outages (Boukhris et al., 2017); these paths are used to create mitigation test paths.

8.14 Use case description

A use case description describes user interactions and a web application under test (Zhang and Tanno, 2015). Both normal and exceptional scenarios can be described using use cases (Zhang and Tanno, 2015).

Testing a web application based on a use case model includes making sure that each user of the application can accomplish all the required functionality specified by the use cases of this application.

8.15 Platform independent test model

Liu et al. (2010) proposed using model-driven architecture (MDA) for automating the process of test data generation for web applications. In this approach, the platform

independent model (PIM) is converted to the PITM, and the latter model is converted to test cases.

8.16 Sequence diagrams

SD can be used to describe the behaviour of a web application under test by specifying the state changes in response to each operation in this application (Suhag and Bhatia, 2014). SD can describe the navigation between the web pages in the web application under test; hence, test cases can be generated in a similar approach described with PNG.

8.17 Web diagram

WD describes the functional requirements of a web application under test (Suhag and Bhatia, 2014). It describes the elements of a web application, such as server pages, HTML pages, and links (Suhag and Bhatia, 2014). When using such model in testing, test coverage is based on every required function by test cases.

9 Other types of web applications testing

In addition to security-based and model-based testing techniques discussed in Sections 7 and 8 above, the testing techniques below were used by the researchers in the field of web applications testing for test data generation for web applications.

- *Mutation testing (Habibi and Mirian-Hosseinabadi, 2015):* This testing technique is based on deliberately making minor changes in the code of the web application under test, where the resulted application is called mutant, to determine whether test cases can detect such changes or not. The aim is to make sure that the test cases can be used to detect faults that usually exist in web applications.

  There are different approaches for mutation testing and one of these approaches is a model-based approach where the mutating operators are based on state machine model such as intentionally adding or removing an edge to this model (Habibi and Mirian-Hosseinabadi, 2015). This is an example that shows that the testing techniques categories in Figure 4 are not mutually exclusive, where testing techniques in the other types of testing category of this figure can also be model-based testing.

- *Semantic-based UIV testing (Li et al., 2010b):* UIV testing is security-related testing which aims to determine whether the web application under test has a required user validation that can be used to detect user inputs related to security vulnerabilities such as SQL injection. Semantic-based UIV testing is used to check that an application has the required validation for detecting semantically invalid or perturbed input, such as inserting '*xyz*' as a user e-mail or a birth year of '*5000*' or any semantically invalid value.

  Semantic-based UIV testing of a web application is to insert semantically invalid inputs by a tester and then determine if this application can handle such input gracefully or not.

- *Performance testing (Ahmad et al., 2018):* In this type of web applications testing, the load of users is increased to an application under test to determine whether this will affect the application's performance. If a web application has poor performance, users will stop using it; therefore, this type of testing is crucial for web applications. Ahmad et al. (2018) defined performance testing as evaluating the response time and the scalability of an application under test when this application is under a certain workload.

  Software testers can simulate using a web application under test by many users to assess the effect in response time as the number of users increase.

- *Search-based testing (Marchetto and Tonella, 2009):* The main target of this testing is to cover the majority of the branches in the web application under test and thereby improve the test coverage. Marchetto and Tonella (2009) proposed an algorithm for search-based testing of a web application based on exploring interaction sequences in this application with the purpose determining the most promising interactions based on test case diversity.

- *Random testing (Artzi et al., 2011):* Random invalid inputs are passed to the web application under test to ascertain whether this application can handle such inputs. This is a basic testing where test inputs are chosen randomly from the input space of the different inputs of the web application under test.

- *Combinatorial testing (Nie and Leung, 2011):* This testing technique depends on generating test data based on the interaction between input parameters. This type of testing can detect faults caused by the interactions of parameters in the web application under test. It is called combinatorial testing test input are based on parameters combinations. For example, a web application under test may be affected by many parameters such as the used web browser or the operating system, number of users, etc. Combinatorial testing checks if any combination of such parameters can cause failure in the web application under test.

- *User session-based testing (Quan and Lu, 2010):* This type of web applications testing is based on gathering user session data or usage data to help in generating test cases for the web application under test. User-session data can be used in other types of testing such as performance testing as done in Quan and Lu (2010).

- *Regression testing (Tarhini et al., 2008):* This testing focuses on selecting a subset of the test case used to test a web application to cover the changes made to this application. When we modify a web application by adding or removing a certain functionality, software tester must determine the parts of the application that were affected by the modifications to test only these parts and not the whole application. This is called regression testing.

- *Concolic testing (Sen, 2007):* This testing is based on automating the process of test input generation based on combining the concrete and symbolic (or concolic) execution of the code under test. The aim is to test as many branches as possible of the web application under test (Li et al., 2014).

- *Metamorphic testing (Aruna and Prasad, 2014):* This testing is based on obtaining useful information from a test case, even if this test case has not been successful in

detecting a fault. This testing technique is used to verify an application output without a complete testing (Aruna and Prasad, 2014). Although a test case did not succeed in detecting a fault, it can still have important information for a software tester.

- *User acceptance testing (Otaduy and Díaz, 2017):* This testing is based on using a web application by the actual users of this application to ensure that the application meets these user's requirements or needs. It is important to test a web application by software tester but it is even more important that this application being tested by the actual users of this application to make sure that it match their expectations.
- *Keyword-driven testing (Gupta and Bajpai, 2014):* This type of testing includes creating test components and then assembling these components into test scripts. There are two types of keywords: base keywords and user keywords. Gupta and Bajpai (2014) described the different type of keywords and how these keywords can be combined to generate test data for a web application under test.
- *Requirement testing:* Sengupta and Dasgupta (2015) presented a framework for performing formal specification and testing on the web applications' functional and interface requirements. This type of testing is used to assess the functional as well as interface requirements of a web application under test. Models such as FSM can be used for the specification of the tested application's requirements. So this type of testing can also be used with model-based testing explained in Section 8.

The research in the field of web applications testing and test case generation for web applications in this mapping study had used all of the above-mentioned testing techniques.

10 Categorising web applications testing techniques based on the approach applied in testing (RQ1)

The most relevant papers in the field of web applications testing were searched for in the libraries specified in Section 4 using the search strings specified in the same section. Subsequently, the inclusion and exclusion criteria were applied to the selected papers.

The data extracted from each paper were

1 the applied web applications testing technique, method or approach

2 the testing purpose

3 the method of evaluation

4 the year of publication as explained in section 4.

After reviewing all the primary research papers in this study, it was decided to classify web applications testing techniques into three categories: model-based testing, security testing, and other types of testing (see Section 6). The reason for this categorisation is that after reviewing the 98 primary research papers in this study, it was concluded that most of the selected research in this field was conducted in the model-based testing and security testing fields. Subsection 10.1 categorises the research papers in the field of SWAT based on the targeted vulnerability, and Subsection 10.2 categorises the research

in the field of MWAT based on the model used for test data generation. Finally, Subsection 10.3 classifies other types of testing used with web applications depending on the investigated studies.

10.1 Classifying research papers in the field of SWAT according to the targeted vulnerability

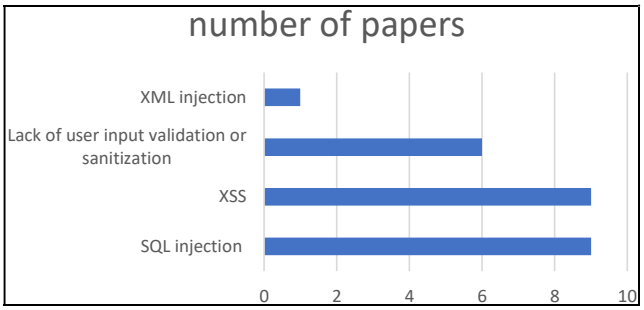
In SWAT faults are detected by injecting malicious or unsanitised inputs in user forms (Li et al., 2014). SWAT techniques attempt to inject such malicious inputs into a web application under test to determine whether these inputs are accepted or rejected by this application. The web applications security-based testing techniques found in the reviewed research papers are SQL injection testing (Lei et al., 2013), XSS testing (Bozic et al., 2015b), XML injection testing (Jan et al., 2019), bypass testing (Offutt et al., 2014), and penetration testing (Tian et al., 2012). Table 3 classifies the reviewed research papers according to the targeted vulnerability in each paper.

Table 3 Classifying research papers in the field of swat according to the targeted vulnerability

Vulnerability	*Research papers*
SQL injection	Shahriar and Zulkernine (2008), Dao and Shibayama (2009, 2010), Huang et al. (2011), Tian et al. (2012), Lei et al. (2013), Akrout et al. (2014), Bozic and Wotawa (2020) and Muzaki et al. (2020)
XSS	McAllister et al. (2008), Huang et al. (2011), Buchler et al. (2012), Avancini and Ceccato (2012), Bozic et al. (2015a, 2015b), Bozic and Wotawa (2020), Muzaki et al. (2020) and Leithner et al. (2021)
Lack of user input validation or sanitisation	Balzarotti et al. (2008), Liu and Tan (2008), Li et al. (2010b), Avancini and Ceccato (2012), Offutt et al. (2014) and Hanna and Munro (2018)
XML injection	Jan et al. (2019)

Figure 5 summarises the results of Table 3 by specifying the number of papers that targeted each of the security vulnerabilities in this study.

Figure 5 Targeted vulnerabilities for the research in the field of SWAT (see online version for colours)



It must be emphasised here that a research paper can include methods to test more than one vulnerability; therefore, the studies in Table 3 are not mutually exclusive. However, it can be concluded from Figure 5 that based on the reviewed studies, the research in the

field of security testing of web applications mainly targets the XSS, SQL injection and lack of user input validation vulnerabilities.

10.2 Classifying web applications model-based testing techniques according to the model used in testing

One of the most commonly used web applications testing techniques is model-based testing. This research categorises the research in this field based on the models used for test data generation and also based on the testing goals. The main aim of this categorisation is to help researchers and practitioners to understand the current trends in this field.

The primary studies in the field of MWAT in this SMS used different models or graphs for test data generation, such as the FSM model or PFG. The studies in this category created a model for the web application under test and then generated test cases based on this model. Table 4 summarises the models or graphs used in each of the primary research papers in the category of model-based testing.

This research study classified the research in the field of MWAT based on the model used in each paper to identify the models used the most by the researchers in this field for test data generation. Another focus of this study was to identify the aims of each research paper in this field. The aim of a paper can be test automation, test coverage, or assessing a specific quality attribute, such as security.

Figure 6 Models and graphs used in MWAT with the number of papers that used each of them (see online version for colours)

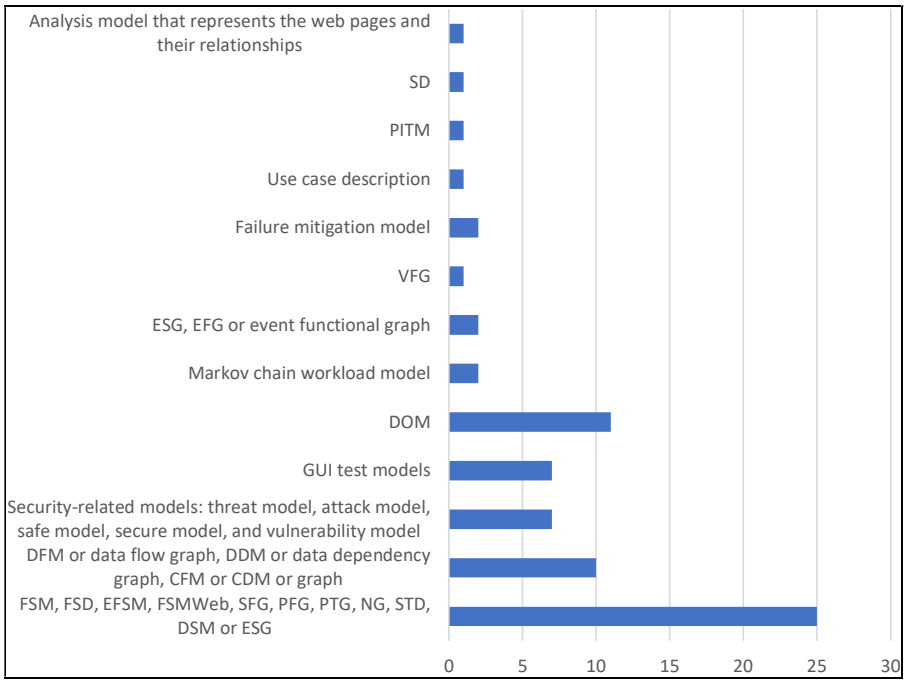


Table 4 Classifying web applications model-based testing techniques based on the model used in testing

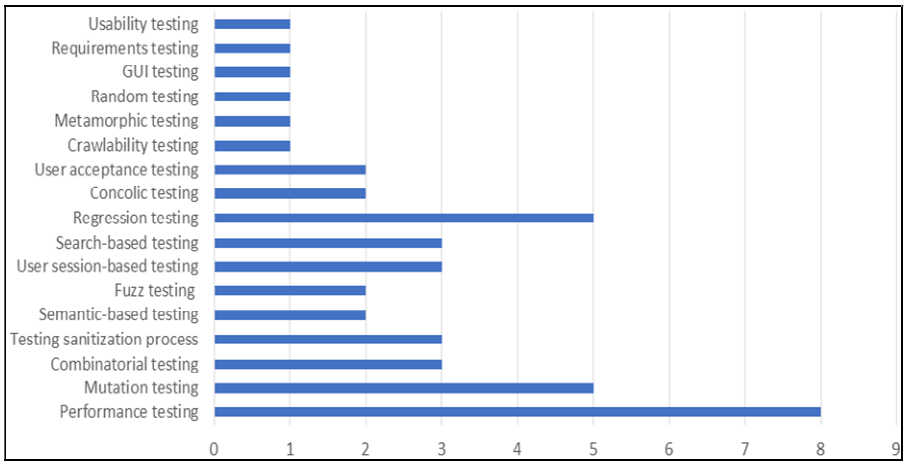
The model or graph used in model-based testing	*Research papers*
FSM, FSD, EFSM, FSMWeb, SFG, PFG, PTG, navigation graph or model, STD or screen transition graph, DSM or ESG	Kuk and Kim (2008), Marchetto et al. (2008), Marchetto and Tonella (2009), Song and Miao (2009), Ran et al. (2009), Andrews et al. (2010), Song et al. (2011), Tkachuk and Rajan (2011), Törsel (2011), Chen et al. (2012), Tanida et al. (2013), Bansal and Sabharwal (2013), Törsel (2013), Arora and Sinha (2013), Törsel (2013), Suhag and Bhatia (2014), Nabuco and Paiva (2014), Polpong and Kansomkeat (2015), Zhang and Tanno (2015), Dixit et al. (2015), Habibi and Mirian-Hosseinabadi (2015), Sengupta and Dasgupta (2015), Qi et al. (2017), Boukhris et al. (2017) and Gao et al. (2022)
DFM or data flow graph, DDM or data dependency graph, CFM or CDM or graph	Tarhini et al. (2008), Ran et al. (2009), Offutt and Wu (2010), Tung et al. (2010), Törsel (2011), Sun et al. (2011), Bansal and Sabharwal (2013), Panthi and Mohapatra (2017), Akpinar et al. (2020) and Elgendy et al. (2020a)
Security-related models: threat model, attack model, safe model, secure model, and vulnerability model	Mallouli et al. (2008), Xu et al. (2012), Tian et al. (2012), Buchler et al. (2012), Avancini and Ceccato (2012), Lei et al. (2013) and Bozic et al. (2015b)
GUI test models	Li et al. (2010a), Heinecke et al. (2011), Boumiza and Azzouz (2012), Dallmeier et al. (2013), Nabuco and Paiva (2014), Mirshokraie et al. (2016) and Hallé et al. (2016)
DOM	Arora and Sinha (2013), Stocco et al. (2014), Fard et al. (2015), Mirshokraie et al. (2015, 2016), Stocco et al. (2016), Akpinar et al. (2020), Imtiaz and Iqbal (2021), Sherin et al. (2021), Corazza et al. (2021) and Gao et al. (2022)
Markov chain workload model	Habibi and Mirian-Hosseinabadi (2015) and Ahmad et al. (2018)
ESG, EFG or event functional graph	Krüger and Linschulte (2012) and Habibi and Mirian-Hosseinabadi (2015)
VFG	Liu and Tan (2008)
Failure mitigation model	Boukhris et al. (2017) and Andrews et al. (2019)
Use case description	Zhang and Tanno (2015)
PITM	Liu et al. (2010)
SD	Suhag and Bhatia (2014)
Analysis model that represents the web pages and their relationships	Kuk and Kim (2008)

In Table 4, the models and graphs used for web applications model-based testing are arranged in descending order from the most commonly used models to the models used the least. One research paper may use more than one model for testing web applications; accordingly, this paper will appear in different sections in Table 4.

It should be noted that one research paper can use more than one model for WAT. For example, a research paper can use both FSM and PFG in test data generation, which means that this paper will appear in different categories of the models used in testing. This applies to all the other research papers in this review where a paper can use different models for test data generation. This means that the models' categories are not mutually exclusive.

Figure 6 summarises the results in Table 4 by specifying the number of papers that used each of the models of test data generation for web applications among the reviewed papers. It can be concluded from Figure 6 that the most commonly used models in MWAT are FSM, FSD, EFSM, FSMWeb, SFG, PFG, PTG, navigation graph (NG) or model, STD or STG, DSM or ESG.

Figure 7 Number of papers in the 'other testing techniques' category (see online version for colours)



10.3 Other testing techniques used for web applications testing

This section will discuss the other types of web applications testing techniques found in the literature. Table 5 presents these testing techniques together with the research papers in this SMS that used each of these techniques. Testing techniques appear in descending order from those used the most to those used the least.

Figure 7 summarises the result of Table 5 by specifying the number of reviewed papers in each type of testing for web applications.

Figure 7 clearly shows that the most commonly used other web applications testing techniques based on the SMS are performance testing, mutation testing, and regression testing. In the performance testing of web applications, the properties of an application are checked when it is used concurrently by multiple users (Rodríguez et al., 2013). In mutation testing of web applications, minor changes are deliberately made in the code of the web application under test to ascertain whether test cases can detect such changes or not. Regression testing focuses on selecting a subset of the test case used to test a web application to cover the changes made to this application.

Table 5 Other web applications testing techniques

Testing technique	*Research papers*
Performance testing	Jiang and Jiang (2009), Gao et al. (2010), Kao et al. (2013), Rodríguez et al. (2013), Ali and Badr (2015), Ahmad et al. (2018), Eid et al. (2020) and Porres et al. (2020)
Mutation testing	Shahriar and Zulkernine (2008, 2009), Xu et al. (2012), Habibi and Mirian-Hosseinabadi (2015) and Sherin et al. (2021)
Combinatorial testing (Nie and Leung, 2011)	Bozic et al. (2015b), Qi et al. (2017) and Bozic and Wotawa (2020)
Testing sanitisation process	Balzarotti et al. (2008), Li et al. (2010b) and Hanna and Munro (2018)
Semantic-based testing (Dadkhah et al., 2020)	Li et al. (2010b) and Hanna and Munro (2018)
Fuzz testing	Li et al. (2014, 2010b) and Offutt et al. (2014)
User session-based testing	Sampath et al. (2008), Quan and Lu (2010) and Sampath and Bryce (2012)
Search-based testing	Marchetto and Tonella (2009), Bolis et al. (2012) and Elgendy et al. (2020b)
Regression testing	Tarhini et al. (2008), Kwon et al. (2018), Andrews et al. (2019), Eid et al. (2020) and Abadeh (2021)
Concolic testing	Wassermann et al. (2008) and Fard et al. (2015)
User acceptance testing	Yu et al. (2009) and Otaduy and Díaz (2017)
Crawlability testing	Marchetto et al. (2011)
Metamorphic testing	Aruna and Prasad (2014)
Random testing	Artzi et al. (2011)
GUI testing	Habibi and Mirian-Hosseinabadi (2015
Requirements testing	Sengupta and Dasgupta (2015)
Usability testing	Marien et al. (2019)

11 Classifying research papers according to the goal of testing (RQ2)

This section will classify the studies in the field of web applications testing according to the goal or target of testing. Subsection 11.1 specifies testing purposes for the research in the field of SWAT, Subsection 11.2 repeat the same process for the MWAT, and finally Subsection 11.3 specifies testing purposes for other types of web applications testing.

11.1 Classifying studies in the field of web applications security testing techniques according to the testing goal

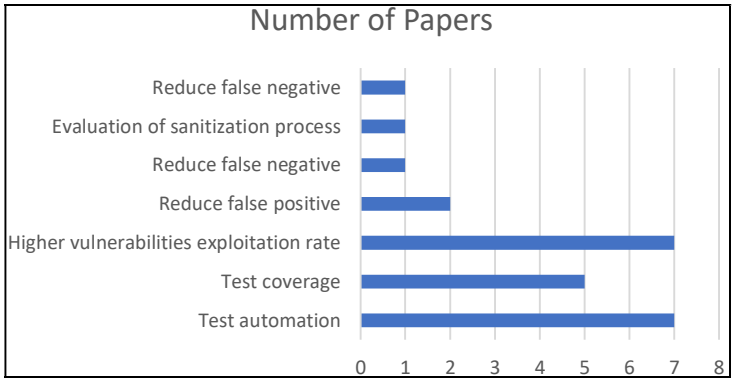
Table 6 displays the testing purposes for the security-based testing techniques in the papers reviewed in this study.

Figure 8 summarises the result of Table 6 by specifying the main testing purposes of the research papers in the field of SWAT.

Table 6 Testing purposes for SWAT techniques

Main testing purpose	*Research papers*
Test automation	Shahriar and Zulkernine (2008), Dao and Shibayama (2009, 2010), Buchler et al. (2012), Akrout et al. (2014), Offutt et al. (2014) and Jan et al. (2019)
Test coverage	Bozic et al. (2015b), McAllister et al. (2008), Dao and Shibayama (2009, 2010) and Liu and Tan (2008)
Higher vulnerabilities exploitation rate	Balzarotti et al. (2008), Dao and Shibayama (2009), Tian et al. (2012), Lei et al. (2013), Bozic et al. (2015a), Muzaki et al. (2020) and Leithner et al. (2021)
Reduce false positive	Huang et al. (2011) and Akrout et al. (2014)
Reduce false negative	Tian et al. (2012)
Evaluation of sanitisation process	Balzarotti et al. (2008)

Figure 8 Testing purposes for the SWAT research (see online version for colours)



It can be concluded from Figure 8 that most of the research in the field of security-based testing of web applications, based on the reviewed papers in this SMS, targets test automation and focuses on higher vulnerabilities exploitation rates.

11.2 Classifying studies in the field of web application model-based testing according to the testing goal

The reviewed papers in the field of MWAT were also classified according to the goal of testing. Table 7 shows the main goals of the model-based testing techniques and the number of research papers that targeted each of these goals. The testing goals appear in descending order, with the purposes targeted the most appearing first.

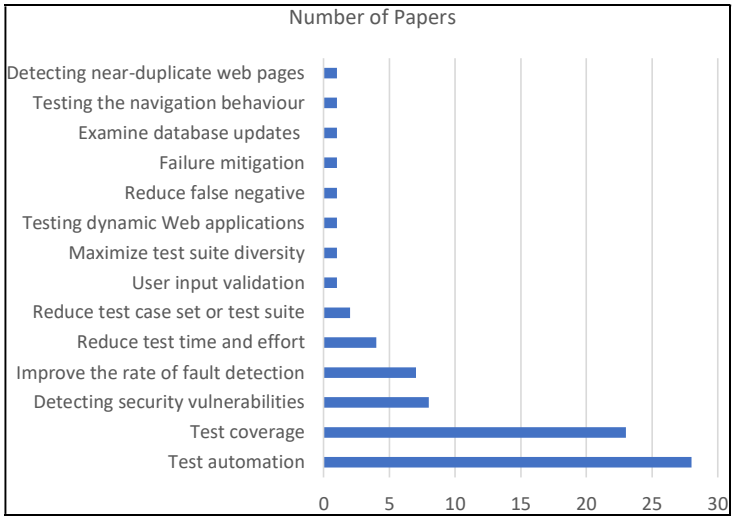
Figure 9 summarises the results in Table 7.

It can be concluded from Figure 9 that, similar to security-based testing, the most commonly targeted testing purposes in the field of model-based testing of web applications is test automation. It must be emphasised here that one research paper can have more than one goal. For example, a research may target test automation and test coverage at the same time; therefore, the research categories in this case are not mutually exclusive.

Table 7 Classification of testing goals for MWAT techniques

Main testing purpose	*Research papers*
Test automation	Kuk and Kim (2008), Ran et al. (2009), Liu et al. (2010), Li et al. (2010a), Tung et al. (2010), Törsel (2011), Heinecke et al. (2011), Tkachuk and Rajan (2011), Sun et al. (2011), Chen et al. (2012), Xu et al. (2012), Buchler et al. (2012), Tanida et al. (2013), Dallmeier et al. (2013), Törsel (2013), Suhag and Bhatia (2014), Stocco et al. (2014), Fard et al. (2015), Mirshokraie et al. (2015), Zhang and Tanno (2015), Stocco et al. (2016), Hallé et al. (2016), Qi et al. (2017), Akpinar et al. (2020), Elgendy et al. (2020a), Imtiaz and Iqbal (2021) and Gao et al. (2022)
Test coverage	Mallouli et al. (2008), Liu and Tan (2008), Ran et al. (2009), Song and Miao (2009), Li et al. (2010a), Song et al. (2011), Arora and Sinha (2013), Bansal and Sabharwal (2013), Tanida et al. (2013), Dallmeier et al. (2013), Lei et al. (2013), Bozic et al. (2015b), Polpong and Kansomkeat (2015), Fard et al. (2015), Mirshokraie et al. (2015), Habibi and Mirian-Hosseinabadi (2015), Stocco et al. (2016), Qi et al. (2017), Andrews et al. (2019), Akpinar et al. (2020), Sherin et al. (2021), Elgendy et al. (2020a) and Gao et al. (2022)
Detecting security vulnerabilities	Mallouli et al. (2008), Song et al. (2011), Xu et al. (2012), Tian et al. (2012), Buchler et al. (2012), Avancini and Ceccato (2012), Lei et al. (2013) and Bozic et al. (2015a)
Improve the rate of fault detection	Marchetto and Tonella (2009), Offutt and Wu (2010), Tian et al. (2012), Arora and Sinha (2013), Mirshokraie et al. (2016), Qi et al. (2017) and Sherin et al. (2021)
Reduce test time and effort	Marchetto et al. (2008), Krüger and Linschulte (2012), Boumiza and Azzouz (2012) and Dixit et al. (2015)
Reduce test case set or test suite	Tarhini et al. (2008) and Tung et al. (2010)
User input validation	Liu and Tan (2008)
Maximise test suite diversity	Marchetto and Tonella (2009)
Testing dynamic web applications	Panthi and Mohapatra (2017)
Reduce false negative	Tian et al. (2012)
Failure mitigation	Boukhris et al. (2017)
Examine database updates	Ran et al. (2009)
Testing the navigation behaviour	Bansal and Sabharwal (2013)
Detecting near-duplicate web pages	Corazza et al. (2021)

Figure 9 Goals of the research papers in the field of MWAT with the number of papers that targeted each of these goals (see online version for colours)



11.3 Classifying other types of studies in the field of web applications testing according to the testing goal

Table 8 specifies the testing purposes or goals for the other web applications testing techniques besides SWAT and MWAT in Subsections 10.1 and 10.2, respectively. The number of research papers targeted each of the testing goals for each testing technique in this category is also specified.

It can be concluded from Table 8 that based on the reviewed studies:

- The main testing goal or purpose is test automation for the web applications performance testing, user session-based testing, and concolic testing.
- For the web applications fuzz testing, mutation testing, and combinatorial testing the main testing goal is detecting security vulnerabilities.
- For the web applications search-based testing, crawlability testing, user acceptance testing, and requirement-based testing, the main testing goal is test coverage.
- For the web applications regression testing, reducing test case set is the main testing purpose based on the reviewed papers.
- For the sanitisation testing and semantic-based testing, detecting semantic-based user input validation vulnerabilities is the main testing purpose.

Table 8 Testing goals for other web applications testing techniques

Testing technique	*Main testing goal*	*Research papers*	#
Performance testing	Test automation	Jiang and Jiang (2009), Rodríguez et al. (2013), Ali and Badr (2015), Gao et al. (2010), Eid et al. (2020) and Porres et al. (2020)	6
	Load assessment	Jiang and Jiang (2009), Rodríguez et al. (2013), Kao et al. (2013) and Ali and Badr (2015)	4
	Test efficiency or accuracy	Jiang and Jiang (2009) and Kao et al. (2013)	2
	Reduce test cost	Jiang and Jiang (2009)	1
	Reactivity assessment	Gao et al. (2010)	1
	Identifying worst path (i.e., a sequence of user interactions)	Ahmad et al. (2018)	1
Fuzz testing	Detecting security vulnerabilities	McAllister et al. (2008), Avancini and Ceccato (2012), Offutt et al. (2014) and Bozic et al. (2015a)	5
	Reduce test time and effort	Krüger and Linschulte (2012)	1
	Higher vulnerabilities exploitation rate	Bozic et al. (2015a)	1
	Test automation	Offutt et al. (2014)	1
	Test coverage	McAllister et al. (2008)	1
Mutation testing	Detecting security vulnerabilities (SQL injection and XSS)	Shahriar and Zulkernine (2008, 2009) and Xu et al. (2012)	3
	Test automation	Shahriar and Zulkernine (2008) and Xu et al. (2012)	2
	Test coverage	Habibi and Mirian-Hosseinabadi (2015) and Sherin et al. (2021)	2
	Generating adequate test data	Shahriar and Zulkernine (2009)	1
Combinatorial testing	Detecting security vulnerabilities	Bozic et al. (2015a, 2015b), Qi et al. (2017) and Bozic and Wotawa (2020)	4
	Test coverage	Bozic et al. (2015b) and Qi et al. (2017)	2
	Increase fault detection	Bozic et al. (2015b) and Qi et al. (2017)	2
	Test automation	Qi et al. (2017)	1
Sanitisation testing	Detecting semantic-based UIV vulnerabilities	Li et al. (2010b) and Hanna and Munro (2018)	2
	Higher security vulnerabilities exploitation rate	Balzarotti et al. (2008)	1

Table 8 Testing goals for other web applications testing techniques (continued)

Testing technique	*Main testing goal*	*Research papers*	#
Semantic-based testing	Detecting semantic-based UIV vulnerabilities	Li et al. (2010b) and Hanna and Munro (2018)	2
User session-based testing	Test automation	Quan and Lu (2010)	1
	Performance assessment	Quan and Lu (2010)	1
	Improve the rate of fault detection	Sampath et al. (2008) and Sampath and Bryce (2012)	2
	Test suite reduction	Sampath and Bryce (2012)	1
	Test suite prioritisation	Sampath and Bryce (2012)	1
	Regression testing	Sampath et al. (2008)	1
Search-based testing	Test coverage	Bolis et al. (2012) and Elgendy et al. (2020b)	2
	Test automation	Bolis et al. (2012)	1
	Increase fault detection	Marchetto and Tonella (2009)	1
	Maximise test suit diversity	Marchetto and Tonella (2009)	1
Regression testing	Reduce test case set	Tarhini et al. (2008)	1
	Faster feedback on failures	Kwon et al. (2018)	1
	Test coverage	Andrews et al. (2019), Eid et al. (2020) and Abadeh (2021)	3
Concolic testing	Test automation	Wassermann et al. (2008) and Fard et al. (2015)	2
	Test coverage	Fard et al. (2015)	1
Crawlability testing	Test coverage	Marchetto et al. (2011)	1
	Test automation	Marchetto et al. (2011)	1
Metamorphic testing	Test automation	Aruna and Prasad (2014)	1
	Increase fault detection	Aruna and Prasad (2014)	1
User acceptance testing	Test coverage	Yu et al. (2009) and Otaduy and Díaz (2017)	2
Random testing	Test coverage	Artzi et al. (2011)	1
	Test automation	Artzi et al. (2011)	1
Requirements testing	Test coverage	Sengupta and Dasgupta (2015)	1
Usability testing	Usability assessment	Marien et al. (2019)	1

12 Classifying research studies according to the method used for evaluation (RQ3)

This section describes the evaluation methods used by each primary study in this SMS. Table 9 specifies the evaluation methods together with the research papers that used each of these methods.

Table 9 The evaluation methods used in the primary studies

Main evaluation method	*Research studies*
Case study	Marchetto et al. (2008), Mallouli et al. (2008), Tarhini et al. (2008), Liu and Tan (2008), Marchetto and Tonella (2009), Song and Miao (2009), Ran et al. (2009), Li et al. (2010a, 2010b), Andrews et al. (2010), Huang et al. (2011), Song et al. (2011), Krüger and Linschulte (2012), Avancini and Ceccato (2012), Törsel (2013), Bansal and Sabharwal (2013), Tanida et al. (2013), Sabharwal et al. (2013), Kao et al. (2013), Aruna and Prasad (2014), Offutt et al. (2014), Suhag and Bhatia (2014), Habibi and Mirian-Hosseinabadi (2015), Polpong and Kansomkeat (2015), Sengupta and Dasgupta (2015), Zhang and Tanno (2015), Stocco et al. (2016), Panthi and Mohapatra (2017), Otaduy and Díaz (2017), Boukhris et al. (2017), Hanna and Munro (2018), Ahmad et al. (2018), Andrews et al. (2019), Akpinar et al. (2020), Elgendy et al. (2020a), Sherin et al. (2021) and Corazza et al. (2021)
Experiments	McAllister et al. (2008), Liu and Tan (2008), Dao and Shibayama (2009), Jiang and Jiang (2009), Sampath et al. (2008), Yu et al. (2009), Ran et al. (2009), Dao and Shibayama (2010), Offutt and Wu (2010), Tung et al. (2010), Gao et al. (2010), Artzi et al. (2011), Halfond et al. (2011), Tkachuk and Rajan (2011), Sun et al. (2011), Marchetto et al. (2011), Heinecke et al. (2011), Bolis et al. (2012), Sampath and Bryce (2012), Xu et al. (2012), Lei et al. (2013), Ali and Badr (2015), Bozic et al. (2015a, 2015b), Mirshokraie et al. (2015), Hallé et al. (2016), Mirshokraie et al. (2016), Qi et al. (2017), Kwon et al. (2018), Jan et al. (2019), Muzaki et al. (2020), Porres et al. (2020), Abadeh (2021), Imtiaz and Iqbal (2021), Sherin et al. (2021) and Gao et al. (2022)
Prototype or tool	Shahriar and Zulkernine (2008), Marchetto et al. (2008), Kuk and Kim (2008), Balzarotti et al. (2008), Wassermann et al. (2008), Dao and Shibayama (2009), Ran et al. (2009), Quan and Lu (2010), Sun et al. (2011), Törsel (2011), Artzi et al. (2011), Bolis et al. (2012), Boumiza and Azzouz (2012), Buchler et al. (2012), Chen et al. (2012), Dallmeier et al. (2013), Arora and Sinha (2013), Rodríguez et al. (2013), Törsel (2013), Tanida et al. (2013), Sabharwal et al. (2013), Gupta and Bajpai (2014), Nabuco and Paiva (2014), Stocco et al. (2014), Akrout et al. (2014), Dixit et al. (2015), Fard et al. (2015), Mirshokraie et al. (2015), Hallé et al. (2016), Stocco et al. (2016), Panthi and Mohapatra (2017), Marien et al. (2019), Eid et al. (2020), Long et al. (2020), Elgendy et al. (2020b) and Leithner et al. (2021)

Some studies used more than one evaluation method such as building a tool and then using this tool in many experiments or case studies.

Also, for the purpose of evaluation, some studies used an experimental web application while other studies used real world industrial web applications. These studies are: Aruna and Prasad (2014), Dao and Shibayama (2010), Dixit et al. (2015), Habibi and Mirian-Hosseinabadi (2015), Hallé et al. (2016), Hanna and Munro (2018), Jan et al. (2019), Krüger and Linschulte (2012), Kwon et al. (2018), Lei et al. (2013), Mallouli et al. (2008), Marchetto et al. (2011), Marien et al. (2019), McAllister et al. (2008), Mirshokraie et al. (2016), Offutt et al. (2014), Qi et al. (2017), Rodríguez et al. (2013) and Wassermann et al. (2008).

To reduce the gap between the research in the field of web applications testing and industry it is preferable to use real applications in the research in this field. This can also encourage industry to use the testing techniques proposed by research studies for detecting faults and assessing quality.

13 Conclusions and future work

The importance of web applications seems to be increasing with each passing day, and researchers need to identify approaches, methods or techniques for testing these applications to assess their dependability.

Due to the importance of web applications, many research studies had been conducted in the field of web applications testing to be able to detect faults in these applications and assess their different quality attributes, such as security and performance.

After conducting a comprehensive literature review in the area of web applications testing, it was found that most of the research in this field is about model-based testing and security testing. This paper conducted an SMS on the research in the field of web applications testing and testing purposes or goals. Ninety eight research papers published between 2008 and 2021 were categorised based on the testing technique and testing purposes.

The results revealed that most of the research in this field is related to model-based testing and security testing of web applications. The testing purpose targeted the most is test automation in both types of testing. The results also revealed that the most commonly used model in research in the field of MWAT is FSM. SQL injection testing is the most commonly used testing type for the SWAT technique. Test automation was the mostly targeted goal for both MWAT and SWAT. This paper also specified all the other testing techniques used in the literature for testing web applications, such as performance testing, fuzz testing, and semantic-based testing. The testing purposes for each of these testing techniques were also specified.

The other web applications testing techniques listed in are performance testing, fuzz testing, mutation testing, combinatorial testing, testing sanitisation process or UIV, semantic-based testing, user session-based testing, search-based testing, regression testing, concolic testing, crawlability testing, metamorphic testing, user acceptance testing, and random testing.

Similar to model-based testing and security testing, for some of the other web applications test techniques such as performance testing, test automation is the main or most targeted testing purpose, while for other testing techniques, such as mutation testing, fuzz testing and combinatorial testing, the main testing purpose is detecting security vulnerabilities. The testing purposes for each of the other web applications testing techniques appear in descending order in Table 8.

Web applications testing techniques are related. For example, mutation testing, combinatorial testing, fuzz testing and semantic-based testing can be used to detect security vulnerabilities such as SQL injection and XSS, which are considered part of security testing. In addition, performance testing can be used in load testing, and user session-based testing can be used with performance testing.

Future research will add more RQs to the SMS. Since researchers use model-based testing and security testing the most, an independent review of each of these techniques

will be conducted. Future research will also consider conducting a SMS for specific web applications testing techniques, especially the techniques that are not adequately covered in literature, such as web applications acceptance testing. Moreover, future research will conduct an SMS for mobile applications due to their great importance nowadays.

References

Abadeh, M.N. (2021) 'Genetic-based web regression testing: an ontology-based multi-objective evolutionary framework to auto-regression testing of web applications', *Service Oriented Computing and Applications*, Vol. 15, No. 1, pp.55–74.

Ahmad, T., Truscan, D. and Porres, I. (2018) 'Identifying worst-case user scenarios for performance testing of web applications using Markov-chain workload models', *Future Generation Computer Systems*, Vol. 87, pp.910–920.

Akpinar, P., Aktas, M.S., Keles, A.B., Balaman, Y., Guler, Z.O. and Kalipsiz, O. (2020) 'Web application testing with model based testing method: case study', *International Conference on Electrical, Communication, and Computer Engineering (ICECCE)*, Istanbul, IEEE, pp.1–6.

Akrout, R., Alata, E., Kaaniche, M. and Nicomette, V. (2014) 'An automated black box approach for web vulnerability identification and attack scenario generation', *Journal of the Brazilian Computer Society*, Vol. 20, No. 1, pp.1–16.

Ali, A. and Badr, N. (2015) 'Performance testing as a service for web applications', *IEEE Seventh International Conference on Intelligent Computing and Information Systems (ICICIS)*, Cairo, Egypt, pp.356–361.

Al-Said Ahmad, A., Brereton, P. and Andras, P. (2017) 'A systematic mapping study of empirical studies on software cloud testing methods', *IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, Prague, Czech Republic, IEEE, pp.555–562.

Andrews, A., Alhaddad, A. and Boukhris, S. (2019) 'Black-box model-based regression testing of fail-safe behavior in web applications', *Journal of Systems and Software*, Vol. 149, No. 2019, pp.318–339.

Andrews, A.A., Offutt, J., Dyreson, C., Mallery, C.J., Jerath, K. and Alexander, R. (2010) 'Scalability issues with using FSMWeb to test web applications', *Information and Software Technology*, Vol. 52, No. 1, pp.52–66.

Anon. (2020) *OWASP Top Ten* [online] https://owasp.org/www-project-top-ten/ (accessed 26 June 2022).

Arora, A. and Sinha, M. (2013) 'Applying variable chromosome length genetic algorithm for testing dynamism of web application', *International Conference on Recent Trends in Information Technology (ICRTIT)*, IEEE, USA, pp.539–545.

Artzi, S., Dolby, J., Jensen, S.H., Møller, A. and Tip, F. (2011) 'A framework for automated testing of JavaScript web applications', *33rd International Conference on Software Engineering*, IEEE, Honolulu, USA, pp.571–580.

Aruna, C. and Prasad, R.S.R. (2014) 'Testing approach for dynamic web applications based on automated test strategies', *48th Annual Convention of Computer Society of India*, Springer, India, Vol. 2, pp.399–410.

Avancini, A. and Ceccato, M. (2012) 'Grammar based oracle for security testing of web applications', *Proceedings of the 7th International Workshop on Automation of Software Test*, IEEE, Zurich, Switzerland, pp.15–21.

Aydos, M., Aldan, Ç., Coşkun, E. and Soydan, A. (2021) 'Security testing of web applications: a systematic mapping of the literature', *Journal of King Saud University-Computer and Information Sciences*, ISSN: 1319-1578.

Balzarotti, D., Cova, M., Felmetsger, V., Jovanovic, N., Kirda, E., Kruegel, C. and Vigna, G. (2008) 'Saner: composing static and dynamic analysis to validate sanitization in web applications', *2008 IEEE Symposium on Security and Privacy (SP 2008)*, California, USA, pp.387–401.

Bansal, P. and Sabharwal, S. (2013) 'A model based approach to test case generation for testing the navigation behavior of dynamic web applications', *Sixth International Conference on Contemporary Computing*, IEEE, India, pp.213–2018.

Bolis, F., Gargantini, A., Guarnieri, M. and Magri, E. (2012) 'Evolutionary testing of PHP web applications with WETT', *International Symposium on Search Based Software Engineering*, Springer, Berlin, Heidelberg, Trento, Italy, pp.285–291.

Boukhris, S., Andrews, A., Alhaddad, A. and Dewri, R. (2017) 'A case study of black box fail-safe testing in web applications', *Journal of Systems and Software*, Vol. 131, No. 2018, pp.146–167.

Boumiza, D.S. and Azzouz, A.B. (2012) 'Design and development of a user interface to customize web testing scenarios', *International Conference on Education and E-Learning Innovations*, IEEE, Tunisia, pp.1–5.

Bozic, J., Garn, B., Kapsalis, I., Simos, D., Winkler, S. and Wotawa, F. (2015a) 'Attack pattern-based combinatorial testing with constraints for web security testing', *International Conference on Software Quality, Reliability and Security*, IEEE, Vancouver, Canada, pp.207–212.

Bozic, J., Garn, B., Simos, D. and Wotawa, F. (2015b) 'Evaluation of the IPO-family algorithms for test case generation in web security testing', *IEEE Eighth International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, IEEE, Vienna, Austria, pp.1–10.

Bozic, J.L.Y. and Wotawa, F. (2020) 'Ontology-driven security testing of web applications', *International Conference on Artificial Intelligence Testing (AITest)*, IEEE, UK, pp.115–122.

Buchler, M., Oudinet, J. and Pretschner, A. (2012) 'Semi-automatic security testing of web applications from a secure model', in *2012 IEEE Sixth International Conference on Software Security and Reliability*, Maryland, USA, 20–22 June, pp.253–262.

Chen, S., Miao, H. and Song, B. (2012) 'AGT4W: automatic generating tests for web applications', *Frontiers in Computer Education*, Vol. 133, pp.885–892.

Clarke, J. (2009) *SQL Injection Attacks and Defense*, Elsevier, ISBN-10: 1597499633.

Corazza, A., Di Martino, S., Peron, A. and Starace, L.L.L. (2021) 'Web application testing: using tree kernels to detect near-duplicate states in automated model inference', *Proceedings of the 15th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, IEEE/ACM, Bari, Italy, pp.1–6.

Dadkhah, M., Araban, S. and Paydar, S. (2020) 'A systematic literature review on semantic web enabled software testing', *Journal of Systems and Software*, Vol. 162, No. 2020, p.110485.

Dallmeier, V., Burger, M., Orth, T. and Zeller, A. (2013) 'WebMate generating test cases for Web 2.0', *International Conference on Software Quality*, Springer, Berlin, Heidelberg, pp.55–69.

Dao, T.B. and Shibayama, E. (2010) 'Coverage criteria for automatic security testing of web applications', *Gandhinagar, International Conference on Information Systems Security*, Springer, Berlin, Heidelberg, India, pp.111–124.

Dao, T-B. and Shibayama, E. (2009) 'Idea: automatic security testing for web applications', *International Symposium on Engineering Secure Software and Systems*, Springer-Verlag, Belgium, pp.180–184.

Dixit, R., Lutteroth, C. and Weber, G. (2015) 'FormTester: effective integration of model-based and manually specified test cases', *37th International Conference on Software Engineering*, IEEE, Firenze, Italy, Vol. 2, pp.745–748.

Doğan, S., Betin-Can, A. and Garousi, V. (2014) 'Web application testing: a systematic literature review', *Journal of Systems and Software*, Vol. 91, No. 2014, pp.174–201.

Eid, S., Makady, S. and Ismail, M. (2020) 'Detecting software performance problems using source code analysis techniques', *Egyptian Informatics Journal*, Vol. 21, No. 4, pp.219–229.

Elgendy, I.T., Girgis, M.R. and Seiwsy, A. (2020a) 'An automated tool for data flow testing of ASP .NET web applications', *Applied Mathematics & Information Sciences*, Vol. 14, No. 4, pp.679–691.

Elgendy, I.T., Girgis, M.R. and Sewisy, A.A. (2020b) 'A GA-based approach to automatic test data generation for ASP .NET web applications', *IAENG International Journal of Computer Science*, Vol. 47, No. 3, pp.557–564.

Fard, A.M., Mesbah, A. and Wohlstadter, E. (2015) 'Generating fixtures for JavaScript unit testing', *30th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, IEEE, USA, pp.190–200.

Gao, P.S.F., Chen, T., Zeng, Y. and Su, T. (2022) 'Model-based automated testing of JavaScript web applications via longer test sequences', *Frontiers of Computer Science*, Vol. 16, No. 3, pp.1–14.

Gao, T., Ge, Y., Wu, G. and Ni, J. (2010) 'A reactivity-based framework of automated performance testing for web applications', *Ninth International Symposium on Distributed Computing and Applications to Business, Engineering and Science*, Washington, USA, pp.593–597.

Garousi, V., Mesbah, A., Betin-Can, A. and Mirshokraie, S. (2013) 'A systematic mapping study of web application testing', *Information and Software Technology*, Vol. 55, No. 8, pp.1374–1396.

Gupta, R. and Bajpai, N. (2014) 'A keyword-driven tool for testing web applications (KeyDriver)', *IEEE Potentials*, Vol. 33, No. 5, pp.35–42.

Habibi, E. and Mirian-Hosseinabadi, S.H. (2015) 'Event-driven web application testing based on model-based mutation testing', *Information and Software Technology*, Vol. 67, No. 2015, pp.159–179.

Halfond, W.G., Choudhary, S.R. and Orso, A. (2011) 'Improving penetration testing through static and dynamic analysis', *Software Testing, Verification and Reliability*, Vol. 21, No. 3, pp.195–214.

Hallé, S., Bergeron, N., Guérin, F., Le Breton, G. and Beroual, O. (2016) 'Declarative layout constraints for testing web applications', *Journal of Logical and Algebraic Methods in Programming*, Vol. 85, No. 5, pp.737–758.

Hanna, S. and Munro, M. (2018) 'Test case generation for semantic-based user input validation of web applications', *International Journal of Web Engineering and Technology*, Vol. 13, No. 3, pp.225–254.

Heinecke, A., Griebe, T., Gruhn, V. and Flemig, H. (2011) 'Business process-based testing of web applications', *International Conference on Business Process Management*, Springer, Berlin, Heidelberg, pp.603–614.

Huang, Y-Y., Chen, K. and Chiang, S-L. (2011) 'Finding security vulnerabilities in Java web applications with test generation and dynamic taint analysis', *2nd International Congress on Computer Applications and Computational Science*, Springer-Verlag, pp.133–138.

Imtiaz, J. and Iqbal, M.Z. (2021) 'An automated model-based approach to repair test suites of evolving web applications', *Journal of Systems and Software*, Vol. 171, No. 2021, p.110841.

Jan, S., Panichella, A., Arcuri, A. and Briand, L. (2019) 'Automatic generation of tests to exploit XML injection vulnerabilities in web applications', *IEEE Transactions on Software Engineering*, Vol. 45, No. 4, pp.335–362.

Jiang, G. and Jiang, S. (2009) 'A quick testing model of Web performance based on testing flow and its application', *Sixth Web Information Systems and Applications Conference*, IEEE, China, pp.57–61.

Kao, C.H., Lin, C.C. and Chen, J.N. (2013) 'Performance testing framework for REST-based web applications', *Najing, 13th International Conference on Quality Software*, IEEE, China, pp.349–354.

Kitchenham, B.A., Budgen, D. and Brereton, P. (2015) *Evidence-Based Software Engineering and Systematic Reviews*, CRC Press, New York, USA.

Krüger, B. and Linschulte, M. (2012) 'Cost reduction through combining test sequences with input data', *IEEE Sixth International Conference on Software Security and Reliability Companion*, IEEE, Gaithersburg, MD, USA, pp.207–216.

Kuk, S.H. and Kim, H.S. (2008) 'Automatic generation of testing environments for web applications', *International Conference on Computer Science and Software Engineering*, IEEE, China, pp.694–697.

Kwon, J.H., Ko, I.Y. and Rothermel, G. (2018) 'Prioritizing browser environments for web application test execution', *40th International Conference on Software Engineering*, Gothenburg, Sweden, pp.468–479.

Lakshmi, D.R. and Mallika, S.S. (2017) 'A review on web application testing and its current research directions', *International Journal of Electrical and Computer Engineering*, Vol. 7, No. 4, pp.2132–2141.

Lei, L., Jing, X., Minglei, L. and Jufeng, Y. (2013) 'A dynamic SQL injection vulnerability test case generation model based on the multiple phases detection approach', *IEEE 37th Annual Computer Software and Applications Conference*, IEEE, Japan, pp.256–261.

Leithner, M., Garn, B. and Simos, D.E. (2021) 'HYDRA: feedback-driven black-box exploitation of injection vulnerabilities', *Information and Software Technology*, Vol. 140, No. 2021, p.106703.

Li, L., Miao, H. and Chen, S. (2010a) 'Test generation for web applications using model-checking', *11th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, Washington, USA, pp.237–242.

Li, N., Xie, T., Jin, M. and Liu, C. (2010b) 'Perturbation-based user-input-validation testing of web applications', *Journal of Systems and Software*, Vol. 83, No. 11, pp.2263–2274.

Li, Y.F., Das, P.K. and Dowe, D.L. (2014) 'Two decades of web application testing – a survey of recent advances', *Information Systems*, Vol. 43, No. C, pp.20–54.

Liu, H. and Tan, H.B.K. (2008) 'Testing input validation in web applications through automated model recovery', *The Journal of Systems and Software*, Vol. 81, No. 2, pp.222–233.

Liu, Y., Li, Y. and Wang, P. (2010) 'Design and implementation of automatic generation of test cases based on model driven architecture', *Second International Conference on Information Technology and Computer Science*, IEEE, Bali Island, Indonesia, pp.344–347.

Long, Z., Wu, G., Chen, X., Chen, W. and Wei, J. (2020) 'WebRR: self-replay enhanced robust record/replay for web application testing', *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ACM, USA, pp.1498–1508.

Mallouli, W., Lallali, M., Morales, G. and Cavalli, A.R. (2008) 'Modeling and testing secure web-based systems: application to an industrial case study', *IEEE International Conference on Signal Image Technology and Internet Based Systems*, IEEE, Bali, Indonesia, pp.128–136.

Marchetto, A. and Tonella, P. (2009) 'Search-based testing of Ajax web applications', *1st International Symposium on Search Based Software Engineering*, IEEE, Windsor, UK, pp.3–12.

Marchetto, A., Ricca, F. and Tonella, P. (2008) 'A case study-based comparison of web testing techniques applied to AJAX web applications', *International Journal on Software Tools for Technology Transfer*, Vol. 10, No. 6, pp.477–492.

Marchetto, A., Tiella, R., Tonella, P., Alshahwan, N. and Harman, M. (2011) 'Crawlability metrics for automated web testing', *International Journal on Software Tools for Technology Transfer*, Vol. 13, No. 2, pp.131–149.

Marien, S., Legrand, D., Ramdoyal, R., Nsenga, J., Ospina, G., Ramon, V. and Spinewine, A. (2019) 'A user-centered design and usability testing of a web-based medication reconciliation application integrated in an eHealth network', *International Journal of Medical Informatics*, Vol. 126, pp.138–146.

McAllister, S., Kirda, E. and Kruegel, C. (2008) 'Leveraging user interactions for in-depth testing of web applications', *International Workshop on Recent Advances in Intrusion Detection*, Springer-Verlag, MA, USA, pp.191–210.

Mirshokraie, S., Mesbah, A. and Pattabiraman, K. (2015) 'JSEFT automated JavaScript unit test generation', *8th International Conference on Software Testing, Verification and Validation*, IEEE, Graz, Austria, pp.1–10.

Mirshokraie, S., Mesbah, A. and Pattabiraman, K. (2016) 'Atrina: inferring unit oracles from GUI test cases', *IEEE International Conference on Software Testing, Verification and Validation (ICST)*, IEEE, Chicago, USA, pp.330–340.

Muzaki, R.A., Briliyant, O.C., Hasditama, M.A. and Ritchi, H. (2020) 'Improving security of web-based application using ModSecurity and reverse proxy in web application firewall', *2020 International Workshop on Big Data and Information Security (IWBIS)*, IEEE, Depok, Indonesia, pp.85–90.

Nabuco, M. and Paiva, A.C. (2014) 'Model-based test case generation for web applications', *International Conference on Computational Science and Its Applications*, Springer, Portugal, pp.248–262.

Nie, C. and Leung, H. (2011) 'A survey of combinatorial testing', *ACM Computing Surveys (CSUR)*, Vol. 43, No. 2, pp.1–29.

Offutt, J. and Wu, Y. (2010) 'Modeling presentation layers of web applications for testing', *Software & Systems Modeling*, Vol. 9, No. 2, pp.257–280.

Offutt, J., Papadimitriou, V. and Praphamontripong, U. (2014) 'A case study on bypass testing of web applications', *Empirical Software Engineering*, Vol. 19, No. 1, pp.69–104.

Otaduy, I. and Díaz, O. (2017) 'User acceptance testing for agile-developed web-based applications: empowering customers through Wikis and mind maps', *Journal of Systems and Software*, Vol. 133, No. 2017, pp.212–229.

Panthi, V. and Mohapatra, D.P. (2017) 'An approach for dynamic web application testing using MBT', *International Journal of System Assurance Engineering and Management*, Vol. 8, No. 2, pp.1704–1716.

Petersen, K., Feldt, R., Mujtaba, S. and Mattsson, M. (2008) 'Systematic mapping studies in software engineering', *EASE'08: Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering*, Italy, pp.1–10.

Polpong, J. and Kansomkeat, S. (2015) 'Syntax-based test case generation for web application', *International Conference on Computer, Communications, and Control Technology (I4CT)*, Malaysia, pp.389–393.

Porres, I.T., Rexha, H., Lafond, S. and Truscan, D. (2020) 'Automatic exploratory performance testing using a discriminator neural network', *2020 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, Porto, Portugal, IEEE, pp.105–113.

Qi, X.F., Wang, Z.Y., Mao, J.Q. and Wang, P. (2017) 'Automated testing of web applications using combinatorial strategies', *Journal of Computer Science and Technology*, Vol. 32, No. 1, pp.199–210.

Quan, X. and Lu, L. (2010) 'Session-based performance test case generation for web applications', *IEEE*, Hong Kong, pp.1–7.

Ran, L., Dyreson, C., Andrews, A., Bryce, R. and Mallery, C. (2009) 'Building test cases and oracles to automate the testing of web database applications', *Information and Software Technology*, Vol. 51, No. 2, pp.460–477.

Rodríguez, F.T., Reina, M., Baptista, F., Usaola, M.P. and Lamancha, B.P. (2013) 'Automated generation of performance test cases from functional tests for web applications', *International Conference on Evaluation of Novel Approaches to Software Engineering*, Springer, Berlin, Heidelberg, pp.164–173.

Sabharwal, S., Bansal, P. and Aggarwal, M. (2013) 'Modeling the navigation behavior of dynamic web applications', *International Journal of Computer Applications*, Vol. 65, No. 13, pp.20–27.

Sampath, S. and Bryce, R.C. (2012) 'Improving the effectiveness of test suite reduction for user-session-based testing of web applications', *Information and Software Technology*, Vol. 54, No. 7, pp.724–738.

Sampath, S., Bryce, R.C., Viswanath, G., Kandimalla, V. and Koru, A.G. (2008) 'Prioritizing user-session-based test cases for web applications testing', *International Conference on Software Testing, Verification, and Validation*, Norway, pp.141–150.

Scholte, T., Balzarotti, D. and Kirda, E. (2012) 'Have things changed now? An empirical study on input validation vulnerabilities in web applications', *Computers & Security*, Vol. 31, No. 3, pp.344–356.

Sen, K. (2007) 'Concolic testing', *Proceedings of the Twenty-Second IEEE/ACM International Conference on Automated Software Engineering*, IEEE/ACM, Atlanta, Georgia USA, pp.571–572.

Sengupta, S. and Dasgupta, R. (2015) 'A VDM-based approach for specifying and testing requirements of web-applications', *Procedia Computer Science*, Vol. 46, pp.774–783.

Shahriar, H. and Zulkernine, M. (2008) 'MUSIC: mutation-based SQL injection vulnerability checking', *The Eighth International Conference on Quality Software (QSIC'08)*, IEEE, USA, pp.77–86.

Shahriar, H. and Zulkernine, M. (2009) 'Mutec: mutation-based testing of cross site scripting', *ICSE Workshop on Software Engineering for Secure Systems*, IEEE, Vancouver, Canada, pp.47–53.

Sherin, S., Iqbal, M.Z., Khan, M.U. and Jilani, A.A. (2021) 'Comparing coverage criteria for dynamic web application: an empirical evaluation', *Computer Standards & Interfaces*, Vol. 73, No. 2021, p.103467.

Song, B. and Miao, H. (2009) 'Modeling web applications and generating tests: a combination and interactions-guided approach', *Third IEEE International Symposium on Theoretical Aspects of Software Engineering*, IEEE, Tianjin, China, pp.174–181.

Song, B., Gong, S. and Chen, S. (2011) 'Model composition and generating tests for web applications', *Seventh International Conference on Computational Intelligence and Security*, IEEE, China, pp.568–572.

Stocco, A., Leotta, M., Ricca, F. and Tonella, P. (2014) 'PESTO: a tool for migrating DOM-based to visual web tests', *14th International Working Conference on Source Code Analysis and Manipulation*, IEEE, Victoria, Canada, pp.65–70.

Stocco, A., Leotta, M., Ricca, F. and Tonella, P. (2016) 'APOGEN: automatic page object generator for web testing', *Software Qual J.*, Vol. 25, No. 3, pp.1007–1039.

Suhag, V. and Bhatia, R. (2014) 'Model based test cases generation for web applications', *International Journal of Computer Applications*, Vol. 92, No. 3, pp.23–31.

Sun, L., Li, J. and Liu, S. (2011) 'Automatic test case generation for web applications testing. Communication systems and information technology', *Lecture Notes in Electrical Engineering*, Vol. 100, pp.657–666.

Tanida, H., Prasad, M.R., Rajan, S.P. and Fujita, M. (2013) 'Automated system testing of dynamic web applications', *International Conference on Software and Data Technologies*, Springer, Berlin, Heidelberg, pp.181–196.

Tarhini, A., Ismail, Z. and Mansour, N. (2008) 'Regression testing web applications', *International Conference on Advanced Computer Theory and Engineering*, IEEE, Thailand, pp.902–906.

Tian, W., Yang, J-F., Xu, J. and Si, G-N. (2012) 'Attack model based penetration test for SQL injection vulnerability', *IEEE 36th Annual Computer Software and Applications Conference Workshops*, Turkey, pp.589–594.

Tkachuk, O. and Rajan, S. (2011) 'Automated driver generation for analysis of web applications', *International Conference on Fundamental Approaches to Software Engineering*, Springer, Berlin, Heidelberg, pp.326–340.

Törsel, A.M. (2013) 'A testing tool for web applications using a domain-specific modelling language and the NuSMV model checker', *IEEE Sixth International Conference on Software Testing, Verification and Validation*, Luxembourg, IEEE, pp.383–390.

Törsel, A-M. (2011) 'Automated test case generation for web applications from a domain specific model', *35th IEEE Annual Computer Software and Applications Conference Workshops*, IEEE, Munich, pp.137–142.

Tung, Y-H., Tseng, S-S., Lee, T-J. and Weng, J-F. (2010) 'A novel approach to automatic test case generation for web applications', *Zhangjiajie, 10th International Conference on Quality Software*, IEEE, China, pp.399–404.

Van Deursen, A., Mesbah, A. and Nederlof, A. (2015) 'Crawl-based analysis of web applications: prospects and challenges', *Science of Computer Programming*, Vol. 97, Part 1, pp.173–180.

W3C (2004) *Document Object Model (DOM) Level 3 Core Specification*, W3C.

Wassermann, G., Yu, D., Chander, A., Dhurjati, D., Inamura, H. and Su, Z. (2008) 'Dynamic test input generation for web applications', *International Symposium on Software Testing and Analysis*, ACM, Seattle, WA, USA, pp.249–260.

Xu, D. Tu, M., Sanford, M., Thomas, L., Woodraska, D. and Xu, W. (2012) 'Automated security test generation with formal threat models', *IEEE Transactions on Dependable and Secure Computing*, Vol. 9, No. 4, pp.526–540.

Yu, L., Zhao, W., Di, X., Kong, C., Zhao, W., Wang, Q. and Zhu, J. (2009) 'Towards call for testing: an application to user acceptance testing of web applications', *33rd Annual IEEE International Computer Software and Applications Conference*, IEEE, Seattle, Washington, USA, pp.166–171.

Zhang, X. and Tanno, H. (2015) 'Requirements document based test scenario generation for web application scenario testing', *IEEE Eighth International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, Graz, Austria, pp.1–3.