Journal of Informatics and Web Engineering

Vol. 3 No. 3 (October 2024) eISSN: 2821-370X

MobiTest – A Software for Mobile-Based Testing

Ayesha Anees Zaveri[1*], Ramsha Mashood[2], Nabiha Faisal[3], Misbah Parveen[4], Naveera Sami[5], Mobeen Nazar[6], Saba Imtiaz[7]

[1, 2,3,4,6,7] University of Kuala Lumpur, Kuala Lumpur, Malaysia
[5] NED University of Engineering & Technology, Karachi, Pakistan
corresponding author: (zaveri.ayesha@s.unikl.edu.my, ORCiD: 0009-0007-7230-1696)

Abstract - MobiTest is an application that serves as a valuable tool in the fast-growing field of software testing. Efficiency is crucial in this industry, where testers, quality assurance teams, and others must meticulously test each application, avoiding the need to repeat the entire cycle to identify bugs. This application is a breeze thanks to its intuitive features and educational content. Thanks to continuous integration, testers can easily keep up with the fast-paced development cycle and start automating tasks as soon as the user interface development is completed. This saves valuable time and ensures a smoother and more efficient process. During the development of this application, a need arose for manual testing, which unfortunately resulted in the inefficient use of resources. MobiTest was designed to overcome these limitations by providing the ability to generate generic test scripts for any application as needed. It can efficiently and adaptively handle intricate tasks according to predefined parameters. This application thoroughly examines every possible detail, allowing the hacker to exploit the system.

Keywords— MobiTest, Codeless, Automation, E-commerce, Leverage, Quality Assurance Teams, Integration, UI development, Generic, Test Scripts, Hacker.

Received: 02 April 2024; Accepted: 30 May 2024; Published: 16 October 2024

1. INTRODUCTION

The product's primary focus is on automating mobile device testing. The goal is to build and deliver successful computerization arrangements, reduce the effort required in code and content, save resources, improve criticism and consistency, and achieve faster results. This will, in turn, directly contribute to an increase in efficiency and prompt a boost in profits, among other things. In addition, once the contents have been created, those same contents are used to carry out the experiments, increasing overall test inclusion and productivity [1].

When the development team can advance swiftly without the fear of breaking current features, the actual value of portable computerization testing is recognized. Appium, Robotium, Selendroid, and Calabash can be used to test mobile applications. Only the very first step of the portable mechanization testing process consists of tests on universal automation equipment with the aid of the devices described in this article. When we have mobile mechanization tests, we need to run them on Android emulators, iOS test systems, or actual devices. Depending on the test, we may use one of these three options [2].

Testing is a necessary step for anyone developing applications. Out of all the readily available options, Appium is almost always the best choice. Not only does it have a Selendroid mode, but it is also much more flexible and can be used for a broader range of things. It's a free tool that can be used to test local, hybrid, and adaptable web apps. It is compatible with almost all platforms, including iOS, Android, Windows, and Firefox OS. Regardless, for us to move forward with Appium, we need to do trials using compost. MobiTest is designed for mobile devices, and the analyst is not required to create test cases or select the type of testing he will conduct (by choosing the ID, locators, etc.). The software that comes with this present has the potential to automate any web-based company's webpage in a single day. Applications that are proficient, rich in quality, and free from defects and blunders are assured by MobiTest, which saves a considerable amount of effort, assets, time, and expense on the endeavor. The program ensures dependable and uncomplicated testing and performance improvement [3]. Because e-commerce apps are so popular with consumers, the MobiTest tool focuses on ensuring they can be used. As a result, mobile testing has developed into an activity that must be addressed. A better way is to check each part or feature of the application to make sure it meets all the requirements and business goals before it can be given to the customer.

Due to the wide variety of mobile devices, platforms, operating systems, and connectivity options, testing mobile applications is a complex and time-consuming operation [3]. One of the tools available on the market for mobile testing applications is Appium. There are many other tools accessible. Nevertheless, it does not include the idea of automated test scripts.

The MobiTest tool, on the other hand, was developed to be compatible with a wide variety of mobile phones and is adaptable to fit into any testing scenario. In addition, since the need for quality elements among customers continues to rise, an effective and automated strategy must be able to interpret the practical results and evaluate the actual outcomes concerning the projected outcomes [4]. Figure 1 depicts how automation testing usually works [1,2,3,4].

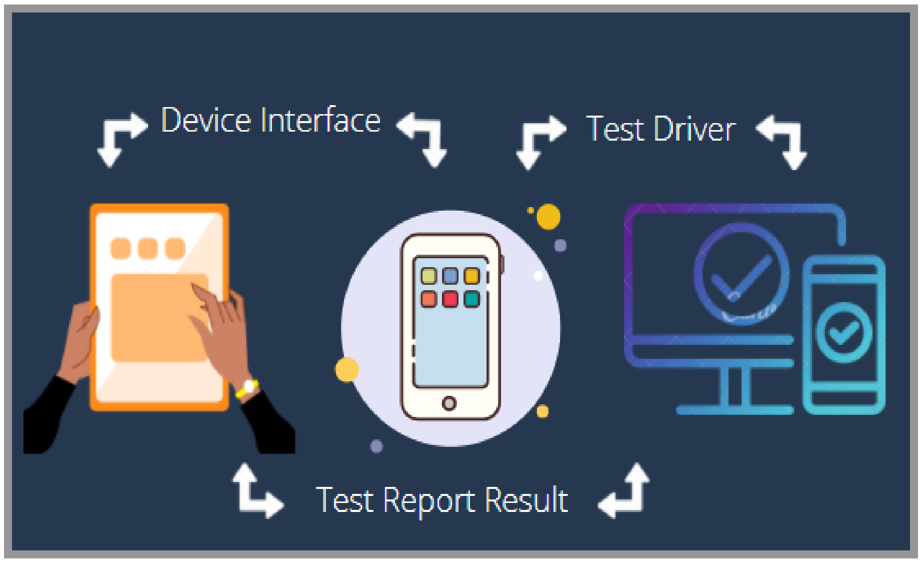


Figure 1. Automation Testing

Codeless Automation Testing is the approach that has been introduced into MobiTest. This strategy can help test engineers avoid writing test scripts before testing any application or component, especially when they are having trouble. To cut down on the time and money spent on testing, all the necessary test scripts have been built on the tool's back end by carefully analyzing each possible scenario of the e-commerce application. With the help of this framework, testing can be carried out efficiently, and a log can easily be kept tracking which tests have been successful and which have been unsuccessful. MobiTest is a codeless automation solution that is now designed for use with mobile e-commerce applications. MobiTest's primary mission is to free software quality engineers from the monotony of manually writing test scripts so they can focus on testing apps. Test automation, when carried out effectively, not only delivers ease and quality but also eliminates the need for manual labor, thereby saving time, resources, and money.

MobiTest is a clever piece of software since it only gets in the way of the other things a tester needs to accomplish while testing in the background. This is one of the reasons why it is so effective. Testers can test apps in one go without agonizing over the scenarios of test scripts, and they can test applications from any location in the world. This provides a moderate amount of relaxation for testers. With the application of this testing strategy, the product's quality can be raised to meet all the quality criteria. MobiTest would provide the most sophisticated and comprehensive automated test design solution. This solution would be based on automation technology that would enable the next generation of testing, which includes transforming, streamlining, and automating even the most complex system-level testing environments. MobiTest would provide this solution to address the market's urgent need.

2. LITERATURE REVIEW

This paper takes a testing approach that is centered on the user. The application's architecture, user base, and usage scenario are crucial when testing an application. Considering these aspects will guarantee that test cases cover all pertinent regions. Automation of testing is necessary for most mobile applications for two main reasons: agility and compatibility. Agile projects do regular testing, such as every night, to identify defects as soon as possible. Compatibility testing on an app ensures it functions appropriately on all available devices and operating system versions. Therefore, testers test scripts on a variety of different devices. This necessitates the use of a private device cloud as well as an automation framework for mobile testing. Swisscom IT Services took this course of action, which allowed us to address the significant quality problems we had identified for mobile apps. These problems included pre-usage failures (such as failed installation and app crashes during startup) and a lack of basic regression testing (upgrades buggier than predecessor) [5].

Mobile applications face a different set of issues than desktop applications and web applications do. For instance, mobile applications must process user input and continuously shift situations to function correctly. In addition, compared to contemporary personal computers and laptops, smartphones and other mobile devices still have restricted features and capabilities. In addition, there is a wide variety of mobile operating systems, and even the same operating system undergoes frequent and rapid iterations of improvement over relatively short periods [4, 5].

When it comes to mobile computing, an application is said to be mobile if it can be executed on an electronic device that can be moved (like an mp3 reader, digital camera, or mobile phone, for example). In [6], the research synthesizes a way to express the distinctiveness and conceptual distinction of mobile computing using four limitations. These include restricted resources, security and vulnerability, performance and reliability uncertainty, and a finite energy source [6, 7, 8].

Mobile applications are present everywhere. While some apps provide entertainment, others make it possible to conduct business on the go. There is a growing level of interaction between apps and complicated IT landscapes. For instance, a banking application running on a mobile device fulfills the function of a front end by invoking services hosted on a back-end server belonging to the bank. These services may then contact different servers. Testing on mobile devices becomes both essential and challenging [9].

In general, two types of mobile applications can be distinguished from one another: Native apps, also called "mobile apps," are software programs that have been made to work only on the operating system and machine firmware of a particular device. Most of the time, these programs must be changed to work on different mobile platforms. When an application is started, some or all its software components may be downloaded from the Internet. This type of application is known as a "web app" or "browser app." It is typically accessible from any mobile device connecting to the Internet. Native applications inherit the peculiarities of mobile applications, such as mobility, autonomy, and connectivity, which directly impact testing [10, 11]. In contrast, mobile applications also inherit the challenges associated with context-aware applications. Native applications also inherit the peculiarities of mobile applications.

Testing carried out on mobile platforms can be broken down into several subcategories, including testing carried out on native mobile applications, testing carried out on mobile devices, and testing carried out on mobile Web applications. We refer to "testing activities for native and web applications on mobile devices using well-defined software test methods and tools to ensure quality in functions, behaviors, performance, and quality of service, as well as features, such as mobility, usability, interoperation ability, connectivity, security, and privacy." When we talk about "mobile app testing," we mean "testing activities for native and Web applications on mobile devices" [12, 13]. As mobile applications and mobile consumers are rising swiftly, it is a concern to researchers and testing.

Professionals will devise viable testing techniques to guarantee the unwavering quality of these mobile applications. A proper mobile quality charter would serve developers as a parameter for mobile quality confirmation. Due to the minor development life cycle of mobile applications, the developed apps tend to be faulty as little effort is put into ensuring the app's quality. Thus, rigorous testing is required to ensure its quality, and that is too quick. Various mobile application testing techniques, such as manual and automated, are used. However, automated approaches are very important due to their multiple advantages [14, 15, 16].

Mobile app testing is a process of testing applications developed for handheld devices. It checks the application for functionality, usability, and performance issues. Mobile app testing is different from testing desktop applications, as apart from regular functional and UI requirements, we must also consider factors like device hardware, screen size, platform, connectivity issues, and many more [17, 18]. Automated testing of Android apps is essential for app users, developers, and market maintainer communities. Given Android's widespread adoption and its development model's specificities, the literature has proposed various testing approaches to satisfy functional and non-functional requirements [19]. Automatic testing of mobile applications was developed for the Google Android platform, and a technique for rapid crash testing and regression testing of Android applications was presented. The method is based on a crawler that automatically builds a model of the application GUI and obtains test cases that can be automatically executed. The process is supported by a tool for crawling the application and generating the test cases [20]. The existing systems working are shown in Figure 2.

Figure 2. Existing Systems

3. AIM AND OBJECTIVE OF MOBITEST

Our research is focused on developing and implementing automation solutions that streamline processes, minimize manual coding and resource requirements, provide rapid feedback, maintain consistency, and deliver expedited results. These efforts are aimed at enhancing productivity and ultimately achieving maximum profitability. MobiTest's primary focus is to create and apply automation solutions that improve efficiency, minimize manual coding and resource requirements, expedite feedback, maintain uniformity, and produce faster outcomes. We are committed to meeting the market's urgent demand by providing a cutting-edge and comprehensive solution for automated test design, with the foundation of our solution lying in state-of-the-art automation technology.

At MobiTest, our primary focus is to create and apply automation solutions that streamline processes and minimize the reliance on coding and resources. This approach enhances feedback speed, maintains consistency, and achieves faster results, ultimately boosting productivity and maximizing profits. We specialize in e-commerce applications, recognizing their wide user impact and potential to transform businesses' operations.

4. TESTING SETUP

Testing mobile applications has always been a complex and demanding endeavor. While there are software solutions that can automate testing, the absence of codeless automation testing is a significant gap. This tool, when available, is precious for saving time and resources while testing mobile applications. Its absence can lead to a substantial increase in manual effort and potential quality compromises. After a thorough investigation into the difficulties faced by testers in software testing, we have uncovered several limitations:

The process of creating test cases manually for each component can be a time-consuming and arduous task. Without a suitable platform, this can compromise the quality of the work and lead to potential errors. This challenge is one that many of us can relate to, and it underscores the need for more efficient testing methods.

The application often requires additional human resources to test its different components thoroughly. This can lead to miscommunication within the team, which in turn can significantly impact the testing process. Testing the entire application in one attempt can be daunting for a single individual, highlighting the need for better team coordination and communication.

The lack of collaboration between the development and testing teams leads to a significant communication gap. This issue not only prolongs the time required to resolve and test bugs and errors but also causes delays in delivering the final product to the client. The importance of a more collaborative approach to addressing this issue cannot be overstated.

Testers encounter many tasks that need to be accomplished throughout the testing process. Thus, the testing of mobile applications has become a crucial task. Given the difficulties at hand, it is essential to adopt a more streamlined approach to comprehensively assess all components and functionalities of an application. This guarantees it fulfills all requirements and aligns with the desired business outcomes before client deployment. Testers have noticed that the time needed for testing can occasionally become longer due to heightened effort, resulting in stress and performance problems. This application ensures the delivery of efficient, high-quality, and error-free applications. This dramatically reduces the need for manual labor, saves precious time, and cuts down on project costs. This application ensures a smooth and effective testing process.

5. RESEARCH METHODOLOGY

The goal of MobiTest is to develop and implement efficient automation solutions that reduce the need for manual coding and resources, provide faster feedback, ensure consistency, and deliver quicker results. These efforts directly contribute to increased productivity and ultimately lead to maximizing profits. MobiTest aims to develop and implement automation solutions that streamline processes, reduce the need for manual coding and resources, provide faster feedback, ensure consistency, and deliver quicker results. These efforts directly contribute to increased productivity and ultimately maximize profits. In response to this urgent demand in the market, MobiTest aims to offer a highly advanced and all-encompassing solution for automated test design. Our solution is built on cutting-edge automation technology.

The main objective of Mobitest is to develop and implement efficient automation solutions that reduce the need for extensive coding and resources. This approach aims to provide faster feedback, ensure consistency, and deliver quicker results. Ultimately, these efforts contribute to increased productivity and the maximization of profits. MobiTest focuses primarily on e-commerce applications due to widespread usage [18, 19, 20].

Testing mobile applications has always been a challenging task to perform. While software solutions are available for automated testing, they do not offer codeless automation testing. This crucial tool can save time and resources when testing mobile applications. After conducting extensive research on the current challenges encountered by testers during software testing, the following limitations have been identified:

1. Writing test cases manually for each component is a time-consuming task that can negatively impact the quality of the work due to the need for a proper platform.
2. The application requires additional human resources to test its various components, leading to potential team miscommunication. This is because it becomes challenging for a single individual to test the entire application in one attempt.

The lack of collaboration between the development and testing teams results in a significant communication gap. This gap not only prolongs the process of resolving and testing bugs and errors but also leads to delays in delivering the final product to the client.

Testers are faced with numerous tasks to complete during the testing process. Therefore, the process of testing mobile applications has become an essential undertaking. The impact of these challenges necessitates a more efficient strategy to thoroughly test every aspect and feature of an application to ensure it meets all expectations and business objectives before being deployed to the client.

Testers have reported that the time required for testing can sometimes increase due to increased effort, leading to stress and performance issues. This application guarantees the provision of efficient, high-quality, and bug-free applications. This significantly reduces the need for manual effort, saves valuable time, and minimizes project expenses. This application guarantees seamless and efficient testing. Efficient resource utilization is achieved [20, 21].

MobiTest is an application designed to test e-commerce mobile applications. It's not just another testing tool but a comprehensive solution. With MobiTest, Software Quality Engineers can test any e-commerce platform without requiring manual test script writing. This is possible by including pre-written test cases for each application's main component in the back end. This approach reduces resource consumption, improves throughput, and empowers testers to test multiple applications simultaneously, a unique feature not commonly found in other tools like Appium. The introduction of MobiTest has led to accelerated release cycles, reduced testing timelines, and decreased expenses. The ability to incrementally develop automated test cases on a sprint-by-sprint basis offers significant adaptability [22].

6. ANALYSIS AND DESIGN

Analysis was made to capture the requirements and implement the designs to complete the project successfully. System analysis is a technique to solve problems. Emphasizing the system's needs, identifying them, and how to fulfill them. System design helps us plan a new business system by satisfying specific requirements, replacing the old one, and telling us how to accomplish the system's objective [22, 23, 24].

6.1 Requirement Capture

Requirements were captured by discovering them from target users and researching to qualify the project's scope. Then, we performed testing to ensure work was carried out successfully, mitigating the risks and the failure to meet the requirements. Finally, we listed all the needs, conditions, and testing areas, as shown in Table 1.

Table 1. Requirement Capture

No.	Requirement	Description
1	Existing testing applications performance and features.	Analyzing and checking the features and interpretations of the existing tools, devices, etc.
2	Software to execute test cases of all possible scenarios.	Writing generic test scripts to test any mobile e-commerce applications.
3	Testing of applications simultaneously to save time.	Testing takes a lot of time; therefore, to reduce stress, the application should simultaneously support testing as many applications as possible.
4	A user-friendly interface for easy user interaction.	A user-friendly interface for easy user interaction.

Figure 3 displays the comprehensive structure of the MobiTest system. Here are the steps:

1. It is simple for the user to create an account. Once they have submitted their information, they will be automatically signed in. It will immediately register as the user on our portal if it is not already registered.
2. Supplying the information that is required for the testing.
3. To do testing, it is essential to select test cases.
4. Following the completion of the test cases, the findings are shown to the operator.
5. After inserting the application key, the administrator executes the test cases and shares the findings [22,23].

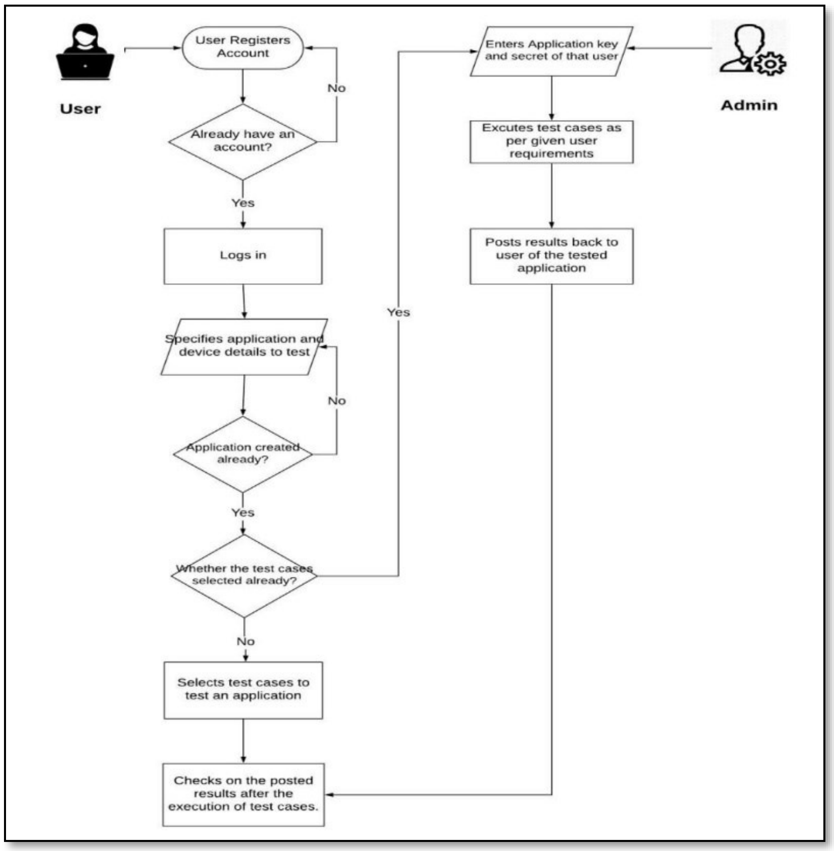

User
User Registers Account
Already have an account?
No
Yes
Logs in
Specifies application and device details to test
No
Application created already?
Yes
Whether the test cases selected already?
Yes
No
Selects test cases to test an application
Checks on the posted results after the execution of test cases.
Enters Application key and secret of that user
Admin
Excutes test cases as per given user requirements
Posts results back to user of the tested application

Figure 3. Workflow Diagram

6.2 Survey Conducted

To get more information, before taking a final step into choosing MobiTest as our project, we conducted a survey, as shown in Figure 4, and the results of the study are shown in Figures 5 to 12.

i. What type of testing does your company or organization prefer to test any application?

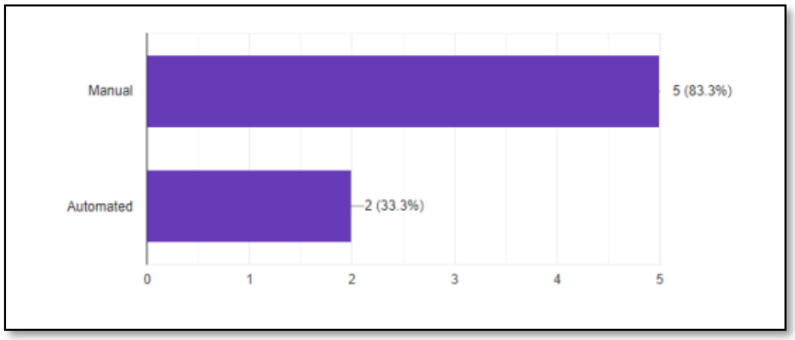


Figure 4. Type of Testing Application

ii. Do you test mobile applications?

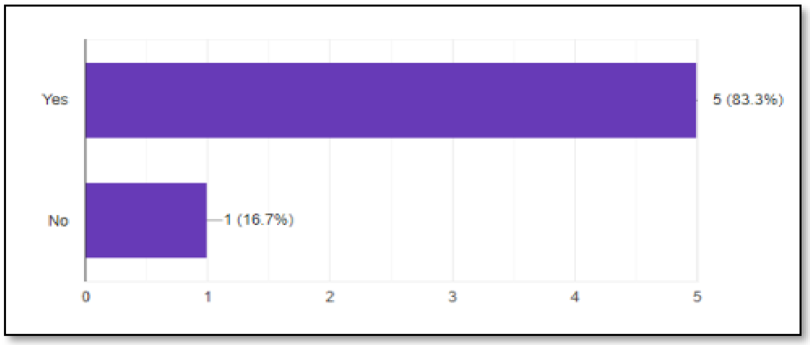


Figure 5. Testing Applications

iii. What automation tools are being used in your software house for mobile testing?

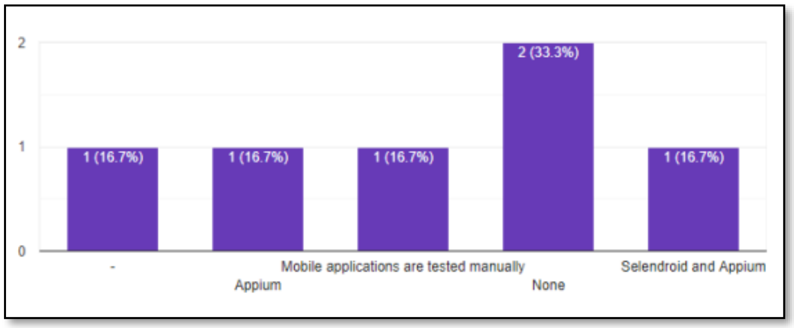


Figure 6. Automation Tools

iv. Would you prefer codeless automation testing over the tools you are using now?

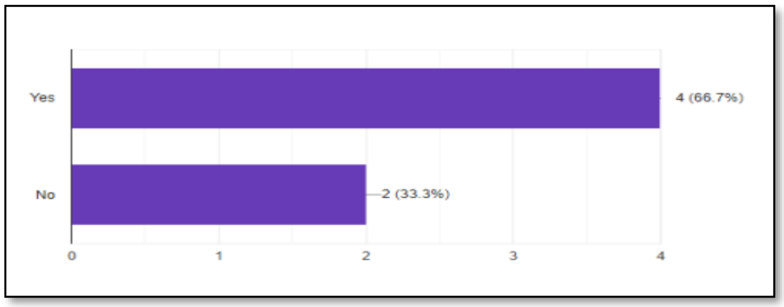


Figure 7. Codeless Automation Testing

v. ***Do you write test plans?***

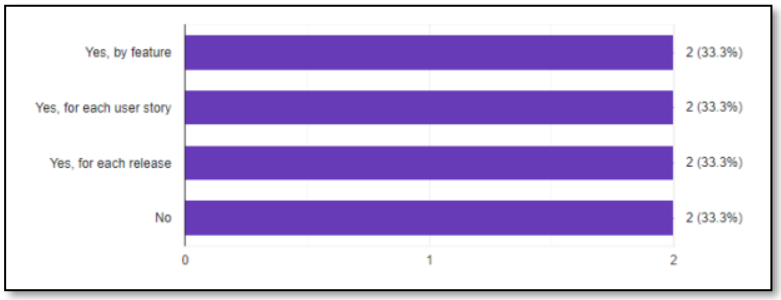


Figure 8. Writing Test Plans

vi. ***What percentage of tests are automated?***

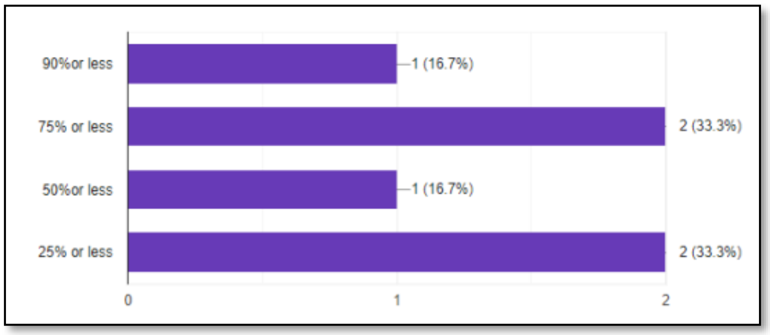


Figure 9. Automation Testing

vii. ***What percentage of time do you serve for mobile application testing?***

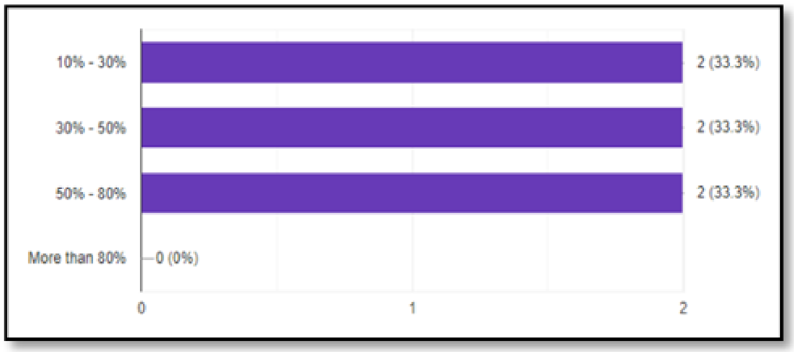


Figure 10. Percentage of Time

viii. ***What difficulties or challenges does your QA team face while testing?***

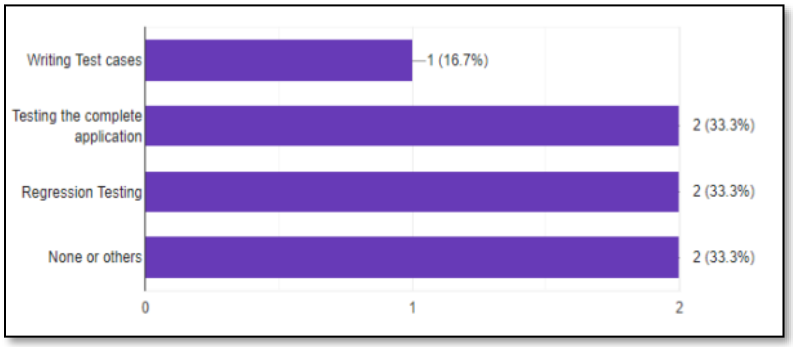


Figure 11. Time for Mobile Aapplication Testing

ix. Any suggestions for MobiTest would be highly appreciated.

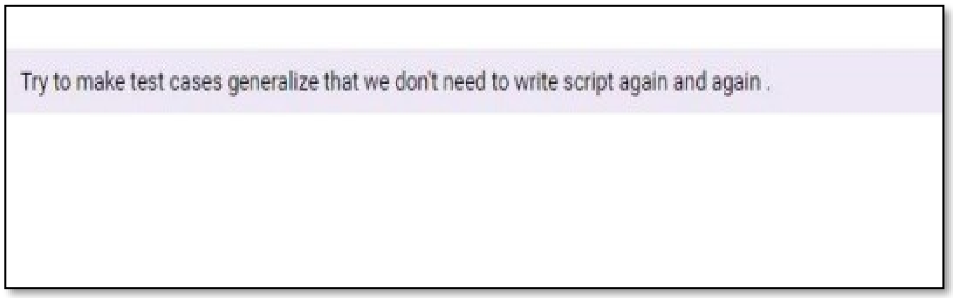

Try to make test cases generalize that we don't need to write script again and again .

Figure 12. Challenges for the QA Team

7. RESULTS

MobiTest uses the black-box and white-box testing methods. These tests provide valuable insights into the application's usability for users. As part of the testing phase, we conducted surveys with the end users to comprehensively understand their experiences.

Assessing the usability of this application is crucial to understanding users' perspectives on its effectiveness. After completing the testing of MobiTest, we were delighted to receive overwhelmingly positive feedback from our target users, Software Quality Engineers from various organizations. They found this application user-friendly and a competitive alternative to other testing tools. The survey we conducted during our analysis phase was instrumental in successfully implementing the application, meeting the exact requirements of our target users, and assuring its quality.

We evaluated to assess the usability of our application. Actual targeted users were asked to use the application while we observed their actions and noted any problems they encountered. Firstly, we provided a concise explanation of how MobiTest functions to help users understand how to use it during testing [23, 25].

Here are the MobiTest interfaces that we tested with our users (see Figures 13 to 18). In conclusion, you may review the paper's main points, elaborate on the importance of the work, or suggest applications and extensions for future work.

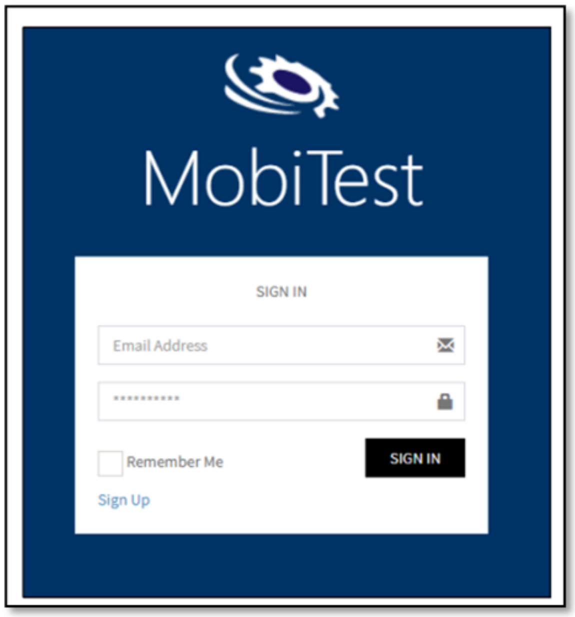


Figure 13. Login Interface

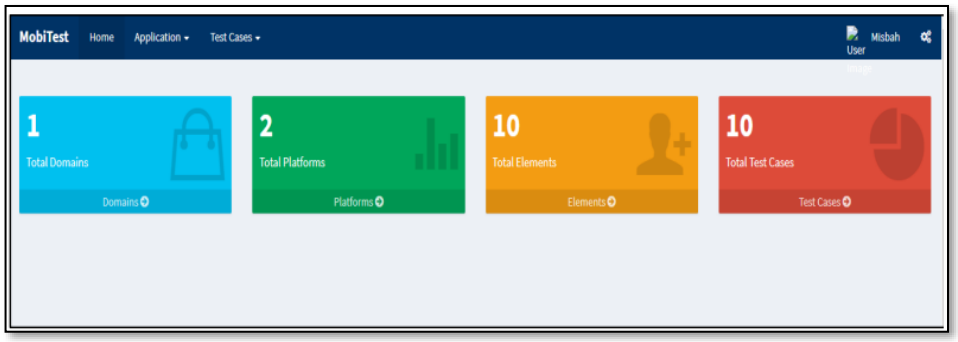


Figure 14. Domains, Platforms and Other Interfaces

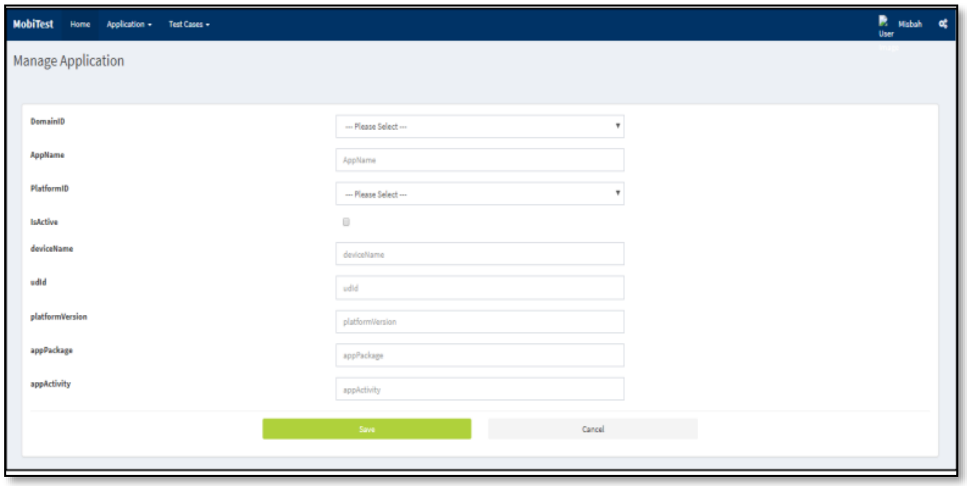
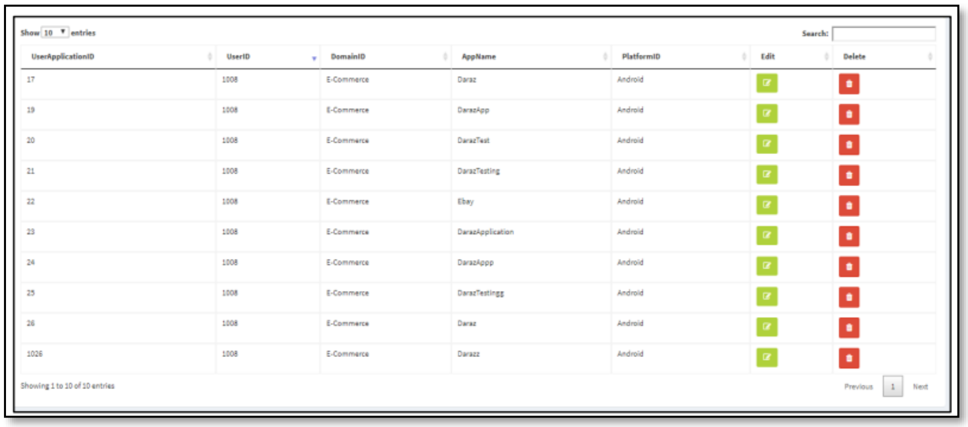


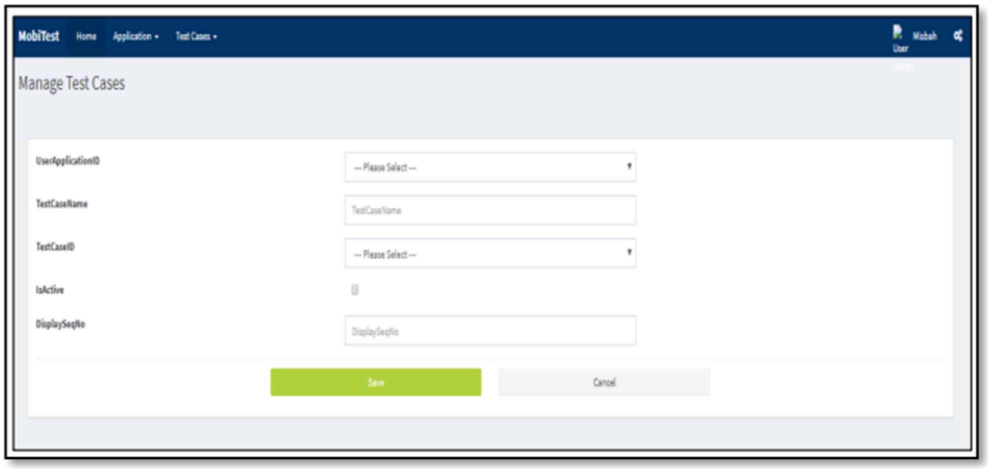
Figure 15. Create Application Interface



UserApplicationID	UserID	DomainID	AppName	PlatformID
17	1008	E-Commerce	Daraz	Android
19	1008	E-Commerce	DarazApp	Android
20	1008	E-Commerce	DarazTest	Android
21	1008	E-Commerce	DarazTesting	Android
22	1008	E-Commerce	Ebay	Android
23	1008	E-Commerce	DarazApplication	Android
24	1008	E-Commerce	DarazAppp	Android
25	1008	E-Commerce	DarazTestingg	Android
26	1008	E-Commerce	Daraz	Android
1026	1008	E-Commerce	Darazz	Android

Showing 1 to 10 of 10 entries

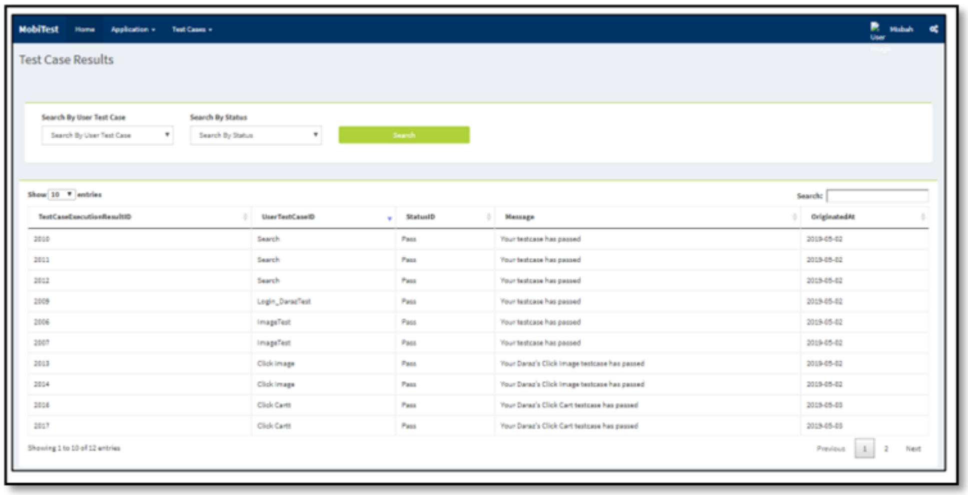
Figure 16. View Application Interface



Manage Test Cases

UserApplicationID	-- Please Select --
TestCaseName	TestCaseName
TestCaseID	-- Please Select --
IsActive	
DisplaySeqNo	DisplaySeqNo

Figure 17. Selection of Test Case Interface



Test Case Results

Search By User Test Case: Search By User Test Case
Search By Status: Search By Status

TestCaseExecutionResultID	UserTestCaseID	StatusID	Message	OriginatedAt
2010	Search	Pass	Your testcase has passed	2019-05-02
2011	Search	Pass	Your testcase has passed	2019-05-02
2012	Search	Pass	Your testcase has passed	2019-05-02
2009	Login_DarazTest	Pass	Your testcase has passed	2019-05-02
2006	ImageTest	Pass	Your testcase has passed	2019-05-02
2007	ImageTest	Pass	Your testcase has passed	2019-05-02
2013	Click Image	Pass	Your Daraz's Click Image testcase has passed	2019-05-02
2014	Click Image	Pass	Your Daraz's Click Image testcase has passed	2019-05-02
2016	Click Cartt	Pass	Your Daraz's Click Cart testcase has passed	2019-05-03
2017	Click Cartt	Pass	Your Daraz's Click Cart testcase has passed	2019-05-03

Showing 1 to 10 of 12 entries

Figure 18. Results Interface

8. EVALUATION OF MOBITEST

Testing mobile apps is more intricate than testing software or online apps due to the unique development standards involved, such as operating system platforms, devices, and screen resolutions. Nevertheless, we have successfully devised and structured a mobile application testing matrix, ranging from [25] to encompass Test Techniques, Testing Environment, Test Level, and Test Scopes, as seen in Figure 19.

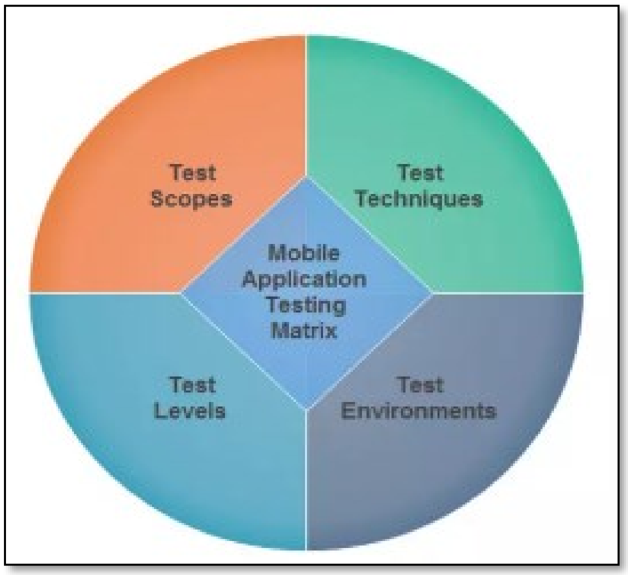


Figure 19. MobiTest Testing Matrix

MobitTest is designed to perform critical indicators to help analyze mobile application performance through testing. The indicators that are crucial for MobiTest are:

8.1 Latency/Response Time

Latency, also known as response time, is the period that elapses between a user's activity within the application and the application's corresponding response to that action. Reduced latency improves the overall user experience. Users anticipate prompt responses to their interactions, and any delays can result in annoyance. MobiTest was tested for its response time according to the user's needs.

8.2 Speed of loading

Load speed refers to the duration it takes for an application to start and become operational following user activation. Enhanced loading speeds create favorable user experiences and potentially impact user retention.

8.3 Displaying visual content on a screen

Screen rendering refers to the duration required to show content on the screen accurately following user interaction. Efficient and flawless rendering is essential for delivering a smooth user interface.

8.4 Throughput

Throughput quantifies the capacity of a system to process a specific quantity of transactions or operations within a specified time frame. High throughput is crucial for applications that handle a significant volume of users or data transactions simultaneously.

8.5 Rate of error

Error rate refers to the frequency at which users encounter faults or bugs while engaging with the application. A low mistake rate is a clear indication of the stability and dependability of an application.

8.6 The application experiences frequent crashes.

App crashes happen when the application abruptly ceases to function. The regular occurrence of system failures can significantly adversely affect user contentment and the application's overall ratings.

8.7 Performance of the Device

Device performance pertains to the efficacy of the application across diverse devices with variable specifications. To accomplish this, we conducted comprehensive mobile application testing across various devices, browsers, platforms, and software versions [23, 24, 25].

During the evaluation of MobiTest, we conducted a comparative experiment to assess its effectiveness in detecting faults in mobile applications. This experiment involved comparing the use of MobiTest with manual procedures. In addition, we conducted a comparison between the actual behavior and the simulated behavior by executing apps on tangible devices and software emulators. The evaluation aimed to assess the degree of increase in flaw detection efficacy using MobiTest, determine the expenses associated with MobiTest, and gather feedback on potential future enhancements of the tool. During the review process, we installed applications on many mobile devices. We included various applications from the public domain for assessment, as indicated in Table 2.

Table 2. Evaluation Report

Instruction No	Action	Component Name	Component Type	Expected Output	Actual Output	Test Result
1	Memory Testing	ToDoManager	Command	Any Value of the type long	18310	PASS
2	Press OK	OK	Command	Notes	Notes	PASS
3	Checking that the note was not added because it has no subject.	Notes	List	Name= (no notes), Value= (no notes)	Name=, Value=;	FAIL
4	Press SAVE	Save Note	Command	Notes	Notes	PASS
5	Check ALARM	Alarm	Choice Group	0	0	PASS

Several test cases were run to check the performance of the Mobitest. Black box and white box testing approaches were implemented on Mobitest to test the product's accountability. In general, testing approaches are divided into two main categories: functional (black box) and structural (white box) [25]. The black box test cases for a few screens are shown in Table 3 and Table 4.

Table 3. Test Case of Login

Test Case ID	TC-01
Description	1. In this test case, the user can log in if registered already. 2. The user should be logged in successfully if already a user. 3. The user must be able to see the domains, platforms, etc, in the MobiTest application. 4. If not, the page will not be processed until the user registers a new account.

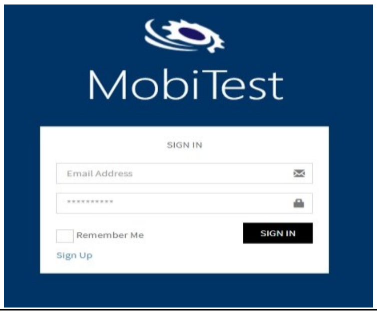
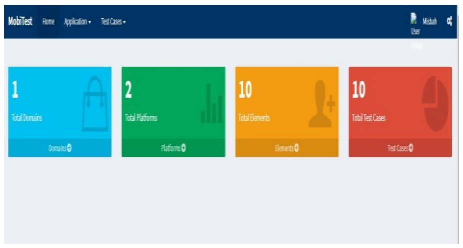
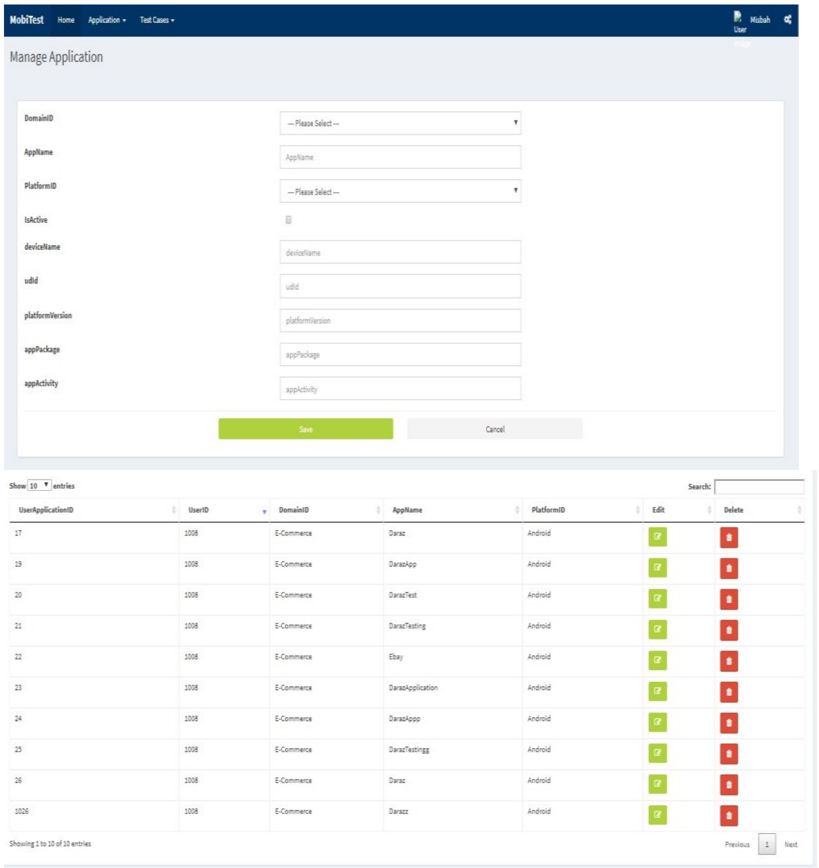
Expected Outcome	 
Result	**Passed**

Table 4. Test Case of Application

Test Case ID	TC-02
Description	1. In this test case, the user can create an application to test. 2. The user will enter the credentials of the application 3. After successfully entering credentials, the entered application will be seen in the data table to the user.
Expected Outcome	
Result	**Passed**

Manage Application

DomainID	--- Please Select ---
AppName	AppName
PlatformID	--- Please Select ---
IsActive	
deviceName	deviceName
udid	udid
platformVersion	platformVersion
appPackage	appPackage
appActivity	appActivity

UserApplicationID	UserID	DomainID	AppName	PlatformID	Edit	Delete
17	1008	E-Commerce	Daraz	Android		
19	1008	E-Commerce	DarazApp	Android		
20	1008	E-Commerce	DarazTest	Android		
21	1008	E-Commerce	DarazTesting	Android		
22	1008	E-Commerce	Ebay	Android		
23	1008	E-Commerce	DarazApplication	Android		
24	1008	E-Commerce	DarazAppp	Android		
25	1008	E-Commerce	DarazTestingg	Android		
26	1008	E-Commerce	Daraz	Android		
1026	1008	E-Commerce	Darazz	Android		

Showing 1 to 10 of 10 entries

9. CONCLUSION AND FUTURE WORK

According to testers, the time estimated for testing increases because of more effort, which causes stress and results in performance issues. This application ensures that applications are efficient, high in quality, and free from bugs, which saves a lot of labor, time, and project costs. This application ensures easy and practical testing. Resources are saved. With all these promising growth numbers and trends, learning mobile app development and testing will be worth it and will have huge demand. Testing is crucial in the fast-growing field of software engineering as it ensures a product is ready for users. Based on the evidence presented, a conclusion can be drawn. Testing applications and writing test cases for each of them is a time-consuming task, given the significance of this field. The presence of mobile devices ensures that mobile testing will continue to be relevant. With mobile phone applications rising in popularity, developers face increased competition. They strive to create apps that cater to various needs and simplify users' daily lives.

Meanwhile, developers often rush through the software development life cycle, prioritizing quick launches and financial gain over thoroughness. With the increasing number of mobile applications, there has been a corresponding rise in both successful outcomes and failures throughout the development process. Launching an application is just the beginning. To ensure success and user satisfaction, it's essential to consider the complete life cycle. Therefore, it was determined that MobiTest is an application that effectively addresses the mentioned issues using user-friendly technology, making it accessible to the intended users.

Additionally, it successfully incorporates the concept of codeless testing. Software Quality Engineers can test multiple applications simultaneously in a single day [23, 24, 25].

MobiTest is an emerging application in the market with the potential to expand its reach to a wide range of consumers and apps for testing purposes. Several examples of these include:

- Domain Expansion: Currently, the domain is limited to Android, but it has the potential to be expanded to include iOS to reach a broader range of mobile applications.
- Platform Expansion: Various types of apps (such as news portals, online chats, etc.) exist, and each of them should be individually targeted soon to broaden the reach and enhance usability among users.
- Bug causality detection: Currently, MobiTest determines if a particular application's test case has passed or failed. It can identify the root cause of a problem, providing developers with more apparent knowledge of the issue and enabling them to resolve it more effectively in the future [24].

ACKNOWLEDGMENT

We want to extend our sincere appreciation to all those who played a crucial role in helping us refine and enhance these concepts, taking them from their initial simplicity to a level of alignment. We are embracing the truth. We want to extend our heartfelt appreciation to the administrations of our respective universities for their invaluable support in facilitating and overseeing our research endeavors with unwavering dedication. The support and understanding offered by our loved ones are invaluable in accomplishing our goals, regardless of their complexity, as they contribute to our overall happiness. We want to extend our gratitude to the support personnel who played a crucial role in helping us compile this article. Their assistance in gathering materials, collecting data, and conducting research was invaluable. Ultimately, the study's authors extend their gratitude and admiration to their fellow contributors for their invaluable support and efficient communication, even in the face of geographical barriers. The collaborative effort significantly contributed to the research's uniqueness and novelty.

FUNDING STATEMENT

This research received no specific grant from any funding agency

AUTHOR CONTRIBUTIONS

Ayesha Anees Zaveri: Conceptualization, Data Curation, Methodology, Validation, Writing – Original Draft Preparation;

Ramsha Mashood: Reviewing Literature;
Nabiha Faisal: Methodology, Writing – Original Draft Preparation;
Misbah Parveen: Graphics and Tables;
Naveera Sami: Analyzing and Designing.
Mobeen Nazar: Proofreading, Structuring, Evaluation.
Saba Imtiaz: Future Work, Concluding, Use Cases, Writing – Review & Editing.

CONFLICT OF INTERESTS

No conflict of interest was disclosed.

ETHICS STATEMENTS

Our publication ethics follow The Committee of Publication Ethics (COPE) guidelines. https://publicationethics.org/

Software quality engineers were surveyed to validate the results.

REFERENCES

[1] V. Garousi and F. Elberzhager, "Test Automation: Not Just for Test Execution," in *IEEE Software*, vol. 34, no. 2, pp. 90-96, 2017, doi: 10.1109/MS.2017.34.

[2] A. Méndez-Porras, C. Quesada-Lopez and M. Jenkins, "Automated testing of mobile applications: A systematic map and review," *CIBSE 2015 - XVIII Ibero-American Conference on Software Engineering*, 2015.

[3] C. M. Prathibhan, A. Malini, N. Venkatesh and K. Sundarakantham, "An automated testing framework for testing Android mobile applications in the cloud," *2014 IEEE International Conference on Advanced Communications, Control and Computing Technologies*, 2014, pp. 1216-1219, doi: 10.1109/ICACCCT.2014.7019292.

[4] K. Haller, "Mobile Testing," *ACM SIGSOFT Software Engineering Notes*, vol. 38, no. 6, pp. 1–8, Nov. 2013, doi: 10.1145/2532780.2532813.

[5] J. Gao, X. Bai, W. -T. Tsai and T. Uehara, "Mobile Application Testing: A Tutorial," *Computer*, vol. 47, no. 2, pp. 46-55, 2014, doi: 10.1109/MC.2013.445.

[6] M. Satyanarayanan, "Fundamental challenges in mobile computing," *Proceedings of the fifteenth annual ACM symposium on Principles of distributed computing - PODC '96*, 1996, doi: https://doi.org/10.1145/248052.248053.

[7] G. Chen and D. Kotz, "A Survey of Context-Aware Mobile Computing Research," *Computer Science Technical Report*, 2000. Available: https://digitalcommons.dartmouth.edu/cgi/viewcontent.cgi?article=1187&context=cs_tr

[8] B. Kirubakaran and V. Karthikeyani, "Mobile application testing — Challenges and solution approach through automation," *International Conference on Pattern Recognition, Informatics and Mobile Engineering*, 2013, pp. 79-84, doi: 10.1109/ICPRIME.2013.6496451.

[9] A. Schmidt, "Implicit human computer interaction through context," *Personal and Ubiquitous Computing*, pp. 191–199, 2000, doi: 10.1007/BF01324126.

[10] A. K. Dey and G. Abowd, "Towards a better understanding of context and context-awareness," *Workshop on the what, who, where, when, and how of context-awareness*, vol. 4, pp. 1–6, 2000. Available: https://www.researchgate.net/publication/274074382_Towards_a_Better_Understanding_of_Context_and_Context-Awareness

[11] H. Muccini, A. Di Francesco and P. Esposito, "Software testing of mobile applications: Challenges and future research directions," *International Workshop on Automation of Software Test (AST)*, Zurich, Switzerland, 2012, pp. 29-35, doi: 10.1109/IWAST.2012.6228987.

[12] A.R. Faqih, A. Taufiqurrahman, J.H. Husen, and M.K. Sabariah, "Empirical Analysis of CI/CD Tools Usage in GitHub Actions Workflows", *Journal of Informatics and Web Engineering*, vol. 3, no. 2, pp. 251–261, 2024.

[13] S. Zein, N. Salleh, and J. Grundy, "A systematic mapping study of mobile application testing techniques," *Journal of Systems and Software*, vol. 117, pp. 334-356, 2016, doi: 10.1016/j.jss.2016.03.065.

[14] A. Kaur, "Review of Mobile Applications Testing with Automated Techniques," *International Journal of Advanced Research in Computer and Communication Engineering*, p. 5, 2015. Available: https://www.ijarcce.com/upload/2015/october-15/IJARCCE%20114.pdf

[15] G. Yan, and Z. Zhan, "Testing of mobile applications: A review of Industry practices," *Blekinge tekniska hogskola*, no. 15 January, 2017. https://www.diva-portal.org/smash/get/diva2:1313292/FULLTEXT02

[16] O. Starov, S. Vilkomir, A. Gorbenko, V. Kharchenko, "Testing-as-a-Service for Mobile Applications: State-of-the-Art Survey," *Springer Link*, vol. 307, pp. 55-71, 2016, doi: 10.1007/978-3-319-08964-5_4.

[17] D. Krupennikov, W. Hester, D. Cortes, C.V. Cole, "Mobile Applications On-Device Testing at Google scale," *Google Research*, p. 24, 2022, doi: 10.1007/s10664-024-10472-6.

[18] M. Akour, B. Falah, A.A. Al-Zyoud, S. Bouriat, K. Alemerien, "Mobile Software Testing: Thoughts, Strategies, Challenges, and Experimental Study," *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 6, 2016, doi: 10.14569/IJACSA.2016.070602.

[19] K.S. Said, L. Nie, A.A. Ajibode, X. Zhou, "GUI testing for mobile applications: objectives, approaches, and challenges," *Digital Library*, pp. 51-60, 2020, doi: 10.1145/3457913.3457931.

[20] R. R. Nimbalkar, "Mobile Application testing and Challenges," *International Journal of Science and Research, vol. 2*, no. 7, pp. 50-60, 2013. https://www.ijsr.net/archive/v2i7/MDIwMTM1OA==.pdf

[21] E. Beck, M. Christiansen, J. Kjeldskov, N. Kolbe and J. Stage, "Experimental Evaluation of Techniques for Usability Testing of Mobile Systems in a Laboratory Setting.," Aalborg Universitet, p. 11, 2003. Available: https://homes.cs.aau.dk/~jesper/pdf/papers/OzCHI03-final.pdf

[22] P. Kong, L. Li, J. Gao, K. Liu, T. F. Bissyandé and J. Klein, "Automated Testing of Android Apps: A Systematic Literature Review," *IEEE Transactions on Reliability*, vol. 68, no. 1, pp. 45-66, 2019, doi: 10.1109/TR.2018.2865733.

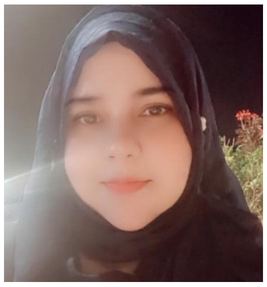
[23] M. T. Baldassarre, D. Caivano, D. Fucci, S. Romano, and G. Scanniello, "Affective reactions and test-driven development: Results from three experiments and a survey," *Journal of Systems and Software*, vol. 185, 111154, ISSN 0164-1212, 2022.

[24] S. Summers and A. Watt, "Quick and dirty usability testing in the technical communication classroom," *IEEE International Professional Communication Conference (IPCC)*, Limerick, Ireland, 2015, pp. 1-4, doi: 10.1109/IPCC.2015.7235831.

[25] S. She, S. Sivapalan and I. Warren, "Hermes: A Tool for Testing Mobile Device Applications," *Australian Software Engineering Conference*, 2009, pp. 121-130, doi: 10.1109/ASWEC.2009.17.

BIOGRAPHIES OF AUTHORS

	Ayesha Anees Zaveri was born in Karachi, Pakistan in 1988. She received her BS and MS in software engineering from Bahria University, Karachi, Pakistan, in 2011 and 2013. She is pursuing a PhD in Information Technology from the University of Kuala Lumpur, Malaysia, in 2022. From 2011 to the present, she has lectured at different institutions in Pakistan and Malaysia. She also worked as an academic writer, researcher, and editor for various organizations. She is the book's co-author, has over twelve articles, and more than 16 projects. Her research interests include Software Engineering, Artificial Intelligence, Agile Development, Machine Learning, Game Development and Business intelligence. The author has won the best presenter award in IPC 2022, organized by Malaysian Universities. She also won the best research paper in the MMU research writing competition. She has participated in conferences and presented her papers in a few.
	Nabiha Faisal received her bachelor's and master's degrees in computer engineering from NED University Karachi, Pakistan, in 2004 and 2007, respectively. She is an educationist at the university level. She is associated with universities like Bahria University Karachi, Pakistan, and Muhammad Ali Jinnah University Karachi, Pakistan. She has authored a book, Computer Graphics Practical Approach for Beginners in Karachi, Pakistan [Lambert Academic Publishing, 2012]. Ms. Faisal's contribution to research has resulted in the publication of 9 journal papers and five conference papers. Her significant area of interest is in the fields [with several publications] of Embedded Systems, Software Engineering, Artificial Intelligence, Machine Learning, game development, business Intelligence and Computer Graphics.
	Mobeen Nazar received her bachelor's and master's degrees in software engineering. She is pursuing a PhD in information technology with the Malaysian Institute of Information Technology, Universiti Kuala Lumpur, Malaysia. Mobeen Nazar is associated with the Department of Software Engineering at Bahria University, Karachi campus, as a faculty member with 7.5 years of teaching experience. Her areas of interest include Artificial Intelligence & Human-Computer Interaction.
	Misbah Parveen was a lecturer at Bahria University Karachi. She also worked as an academic researcher, student affairs advisor, and cluster head in the software engineering department. She is a co-author of over nine articles and more than 12 projects. Her research interests include software engineering, Software construction, Data analytics, Artificial intelligence, Data Mining, Data science, and business Intelligence. The author has attended workshops on research paper working and can be contacted at MisbahParveen.bukc@bahria.edu.pk.

	Ramsha Mashood's contributions to the field are marked by her extensive work in data analysis, machine learning, and software development. Her research has been published in the Journal of Informatics and Web Engineering, showcasing her expertise and dedication to advancing technology. Ramsha's excellence in her field is further recognized through awards at IPC 2022 and IEEE events, highlighting her innovative contributions and commitment to excellence. With a passion for programming, teaching, and public speaking, Ramsha Mashood continues to inspire and lead in computer science and engineering.
	Naveera Sami, bachelor's in computer and information systems engineering, NED University (2011). M.Eng. (Computer Architecture and Systems Design), NED University (2015). Her Discipline is Computer Engineering, and her Skills and expertise are in Information and Communication Technology and Signal, Image, and Video Processing. She has written six research papers.
	Saba Imtiaz is a highly motivated professional with a bachelor's degree in computer science and a master's degree in software engineering. With a strong passion for teaching, she is currently working as a Data Science Instructor, imparting her knowledge and skills to students in Compiler Data Structure. She is keenly interested in the latest trends and technologies in data science. She strives to keep herself updated with the latest advancements. She has excellent communication skills and is adept at conveying complex concepts simply and understandably. She is a dedicated individual committed to delivering high-quality education to her students and ensuring their success in their academic and professional endeavors.