International Journal of Research in Engineering, Science and Management
Volume-3, Issue-2, February-2020
www.ijresm.com | ISSN (Online): 2581-5792

Software Testing for Mobile Applications

Sumandeep Kaur
Assistant Professor, Department of Computer Science, Govind National College, Ludhiana, India

***Abstract*: Software testing refers to the process of executing programs and applications in order to find software bugs and other defects. This process of technical investigation, performed on the behalf of stakeholders is intended to reveal quality-related information about the product with respect to the context in which it is intended to operate. There are almost 4 million mobile apps available for download. Meaning customers have a bevy of options to choose from. This highly-competitive supply and demand means you must ensure that the quality, usability, and security of your mobile app not only meets expectations but exceeds them. This is the power of mobile app testing. With the popularity of mobile applications exploding, software product companies face a tantalizing opportunity to expand their market reach by offering mobile products. Yet, the twin side of this opportunity is the danger associated with releasing a mobile product too soon, before all the major bugs have been discovered and fixed; today's consumers are so demanding that a poor experience with mobile can often lead them to reject the entire brand. This paper will help the learners to learn the different aspect of the up-trending mobile application testing. You will get familiar with many useful tools for a mobile application.**

***Keywords*: Emulators, Security, Usability, Quality, CPU Utilization.**

1. Introduction

Mobile application testing is the process every application developed for handheld devices has to go through. This, obviously, is to assure a certain level of quality before an application is released into the market place (app store/ play store). Mobile application development life cycle generally tends to be much shorter than others, hence, heavily depend on mobile application testing for their success. Applications get tested on the basis of security, their functionality, usability etc. This increases the general efficiency of the application on all fronts while also increasing the reliability factor amongst users. Mobile application testing can be an automated or manual type of testing. Mobile applications either come pre-installed or can be installed from mobile software distribution platforms. Before testing any mobile apps, decide what testing is required to test the specific mobile app: functional, usability, compatibility, performance, security, etc. Decide on which target devices to use and what functional requirements should be tested. You must also determine what target devices to include. You can do this by:

- Figuring out what devices the application will support
- Determining the earliest version of relevant operating systems will be supported.
- Identifying the most popular models for the target audience.
- Selecting some additional devices with different screen sizes.
- Deciding if you'll test mobile apps on real devices or emulators.

2. Testing tools

Checklist for selecting Mobile testing tools,

- *Cross-Platform (and Multi-Version) Support* – Mobile testing tools will need to support numerous devices with various platforms, as well as multiple versions of a given OS.
- *Cross-Browser Support* – Tools will need to test different versions of Internet Explorer, Chrome, Safari, Firefox and (possibly) Opera on multiple operating systems, to validate a consistent experience for all users.
- *Reusability* – Mobile testing tools need to support reusable test scripts to save time and effort. Look for avenues for early automation, because most mobile projects are Agile.
- *Data-Driven Automation Support* – Look for tools that will support iterations in the execution, to increase coverage and ROI. Good candidates for automation are test cases that have repeated business logic, functional flows or data-driven tests.
- *Cost* – Several factors will enter into your tool cost analysis, including maximizing automation (reducing costs), support across multiple platforms, script reusability, and total cost of ownership. Ensure ROI per each mobile platform and duration of execution.
- *Ease of Use* – The more difficult–to-use tools will cost more in terms of training time.
- *Support from the Tool Vendor* – Mobile testing tools should be supported by their manufacturer upon installation, as well as for ongoing maintenance, with less lead time expected for new OS versions and device availability on the tool.
- *Hybrid Tool Combinations* – Select tools that support both platform simulators and actual devices, because you can mix and match devices and simulators to optimize runs on different platforms.
- *Support for Non-Functional Requirements* – It's critical for mobile testing tools to address NFR adherence,

including the following areas: battery use, memory use (with respect to volatile, non-volatile, and memory card usage), memory leaks, CPU and network usage, and bandwidth use (the tool should be functional across 2G, 3G, and Wi-Fi).

- *Battery Consumption* - On mobile devices, the battery is a scarce and valuable resource. Battery consumed by AUT (Application under test) is very important to measure and it cannot be done without the support of a tool.
- *Memory Usage* - Most Mobile devices have low memory space. So, it is important to measure memory usage while running the AUT. A good testing tool can help us monitor memory usage.
- *Memory Leaks* - The device should never run out—or run low—of memory. When a program needs to store some temporary information during execution, it dynamically requests a chunk of memory from the system from its fixed amount of total memory available. If one application uses up all of the system's free memory, then other applications will not be able to obtain the memory that they require. It is the responsibility of each application to free dynamically requested memory when it is finished using it, so that the memory returns to the system and can be allocated to other applications as required. If the application does not free the memory after finishing its execution, this situation is called memory leaks. The testing too should detect such memory leaks as they can leads to slowing down or hanging of the device.
- *CPU Utilization* - The tools should be capable of calculating per-process CPU usage statistics for all device ranges – small (slower CPU and small RAM), mid (average/optimal CPU, RAM) and high (dual/code CPU, higher RAM). CPU usage and network usage are the two major sources of battery drainage.

3. The winning mobile application testing strategy

Here are five key factors to developing a winning mobile app testing strategy that will ensure that your quality assurance activities align with customer expectations, business goals, and industry best practices.

1) Mobile Device Selection

The leading concern before beginning mobile app testing activities is to choose the ways of testing the app. This primary testing method can be a difficult decision to make as it corresponds directly to the market and reach for your app. Within device selection, there are two choices to be made: selecting the gadget model or choosing between emulators and substantial devices. The factors below require consideration during device selection:

- *OS Version:* test your mobile app on all stable OS versions.
- *Screen Resolution:* use a mix of different screens to test by size and resolution
- *Form Factor:* if the app is compatible with smartphones and tablets, test for form factors

As needed, numerous other factors such as memory size, connectivity options, etc. need to be accounted for while selecting the device model.

2) Emulators vs. Physical Devices

You can also decide among physical devices or emulators. Predominantly in the opening stages of development, device emulators are extremely useful because they assist rapid and efficient testing, especially in an agile development environment.

Device emulators are also very cost-effective. Mobile device emulators are very useful for basic application functionality testing and during feature development. They provide excellent options for network bypass, a pseudo-live environment, and test scripting languages.

Using mobile device emulators does not mean that you should avoid using physical devices altogether. Testing on physical devices is imperative; it allows the understanding of application activities in real-life scenarios. Mobile device testing is all about using a right mix of emulators and physical devices to get the best results, quickly, and efficiently.

Physical device testing leads to beta testing as an extremely helpful method of mobile app testing which gives you admittance to real-world testers, real devices, actual networks, and a wider geographic coverage.

Beta testing is a major area where emulators fail in comparison to physical mobile devices. Beta testing gives you a chance to test your mobile app for factors like:

- Network density
- How the app behaves on specific devices
- How real-world users interact with the app
- Different battery states on the devices
- Multiple networks (Wi-Fi, 4G, 3G, etc.)

The real-world testing environment in beta testing is nearly impossible to create in a test lab.

3) Mobile App Testing on Cloud

Cloud-based mobile application testing makes potentially infinite scenario combinations easier to manage. Cloud-based testing is a desirable option for testing mobile applications. Cloud computing provides a web-based mobile testing environment where applications can be deployed, tested, and managed. Besides providing on-demand access to the diversity of mobile devices, cloud testing environment helps curtail the project costs while increasing ROI.

Cloud-based application testing delivers the following benefits:

- Highly synchronized and pre-configured architectures
- Reduction in defects associated with unstable test configurations
- Saves businesses from setting up on-premise test conditions that take lots of time
- No additional needs for advanced tools, server settings, licensing

- Supports complex applications, which solves the problems of testing in-house
- Scalability to leverage an application's capacity threshold
- Results in real-time, which means defects can be analyzed while tests run

Cloud-based mobile application testing reduces time to market (TTM) and significantly augment testing competence.

4) Network Connectivity

Network connectivity significantly affects mobile applications. The majority of mobile applications depend on network connectivity to work correctly. Therefore, testing mobile apps in actual network environments is necessary to get the real picture of the application's behavior.

There is a multitude of network simulation tools available to test mobile apps against network speeds, bandwidths limitations, connection drop outs, and more. These network simulation offerings add exceptional value to the testing activities. For a mobile app to win, it needs to deliver consistent performance across diverse network environments.

5) Manual vs. Automated

Manual Testing vs. Automated Testing—who wins? Automation is key to successful regression testing during development stages. However, automated testing requires a substantial amount of initial investment.

Therefore, test automation should be done only in scenarios where:

- The application is growing
- The mobile development lifecycle is long
- The scale and frequency of regression testing are high
- A significant portion of test cases includes obtainable functionality test cases. With automation, mobile application testing for the following becomes easy:
- Verifying application compatibility with newly released operating systems
- Validating backward compatibility during application upgrades

6) Mobile App Performance Testing

It's good practice to test your application for performance and scalability issues. With large storage capacity being available at affordable prices, it's not uncommon for users to have large amounts of data or content on their smartphone. Have you checked to see if the performance of your mobile application degrades with an increase in the size of mailboxes, albums, messages or any other content relevant to the application?

Users even store SMS for several years on their smartphones. If your application has user generated content or data associated with it (e.g. photos, music, etc.) which can grow to enormous proportions over the application's lifetime. Your testing should include these scenarios to see how the application performs. In case the application has a server side component, you should also test the application with increasing number of users.

In case the application has a server side component, you should also test the application with increasing number of users. While this testing can be done manually, we have tools like Little Eye and Neo Load that can help you with performance and load testing of your mobile app.

7) Mobile App Security Testing

Security and data privacy aren't optional. Users are worried about their data and credentials being exposed through vulnerable applications that happen all too often. These are eight questions you'll need to answer if you're aiming to win at mobile application testing:

1. Is your application storing payment information or credit card details?
2. Does your application use secure network protocols?
3. Can they be switched to insecure ones?
4. Does the application ask for more permissions than it needs?
5. Does your application use certificates?
6. Does your application use a Device ID as an identifier?
7. Does your application require a user to be authenticated before they are allowed to access their data?
8. Is there a maximum number of login attempts before they are locked out?

Applications should encrypt username and passwords when authenticating the user over a network. One way to test security related scenarios is to route your mobile's data through a proxy server like OWASP Zed Attack Proxy and look for vulnerabilities.

4. Mobile app testing types

1) Functional Testing

Functional software testing ensures that the application is, well, functioning, correctly. This type of testing focuses on the main purpose and flow of the app. In addition to the mobile app's specific functionality, there are other scenarios one should test for to limit errors, including but not limited to:

- The application installs and launches correctly
- The users can sign-up and login
- Text boxes and buttons function properly
- Push notifications render correctly

Only 4 out of 100 unhappy customers will complain directly to a company — the other 96 will churn without providing feedback. Since it's 6-7 times more expensive to acquire a new customer than keep an existing one, unlocking that silence is key. – thinkJar

2) Usability Testing

Known as user experience testing, usability testing checks how user-friendly the app is in terms of ease of use and intuitiveness. Ideally, usability testing revolves around the entire app-driven customer experience with insights that include the identification of bugs and recommendations for ways to improve the customer experience, both in and out of the app.

Engineers, marketers and product people all want to test whether or not the end-to-end "app-driven" experience is world-class. To that end, it's important for app usability testing to be done with real people, on real devices to quickly identify and fix usability issues prior to app release. Because usability testing is subjective, you should understand your target end-users and their preferences. Consider asking them to test the product themselves.

Other best practices for usability testing include:

1. The thoughtful design of usability test scripts and feedback questionnaires
2. Incorporate usability questionnaires within test cycles so testers understand the usability testing instructions, can access the online questionnaires and can provide feedback as part of their testing tasks.
3. Analyze results and create feedback summary with actionable insights and recommendations for improving the overall customer experience.

3) Compatibility Testing

Compatibility testing is a type of non-functional testing and is critical as it ensures your mobile app works on various operating systems, a plethora of devices and applications, network environments, and with particular internal hardware specifications.

Specifically, you should know if:

- The app is compatible with different operating systems and their various versions (iOS, Android, Windows, etc.).
- The app performs well with varying networks and their parameters (bandwidth, operating speed, etc.).
- The app is compatible with different browsers (Google, Firefox, Safari, etc.).

4) Performance and load testing

Performance testing checks how well the mobile application performs under a particular workload. These tests are important to ensure your app isn't malfunctioning.

Performance and load tests check for the following:

- *Device performance:* Start-up time, battery consumption, memory consumption
- *Network performance:* Delays or errors in receiving information
- *API / Server performance:* How quickly and in what format data is transferred

Additionally, your app should have built-in back-up and recovery functions that save or recover user data that could be lost for any reason. This is where you would test that capability compatible with different devices (screen size, data storage, etc.)

5) Installation Testing

Installation testing is used to check the mobile application is installing and uninstalling properly. Additionally, installation testing ensures updates are also uninterrupted and error-free. This includes understanding what happens if a user doesn't update an app.

6) Localization Testing

Consumers routinely skip past apps that don't consider graphical or UI elements aligned with their culture, language, or device accessibility. And when an app attempts to translate, it may sound awkward to a native speaker. At the same time, localization testing continues to be a challenge as half of all QA teams lack access to the resources needed to test localization.

When understanding your end-user, you must consider the cultural and geographic aspects of your audience. For example, consumers routinely skip past apps that don't consider graphical or UI elements aligned with their culture, language, or device accessibility. From translating in multiple languages to converting to local currencies and adhering to local regulations and legal requirements, it's important to ensure the app is accessible and usable in a wide variety of markets. Equally important to check that localization efforts haven't caused new breaks.

Want to beat your competitors on the global market? According to research conducted by AppAnnie, fully localized apps is how you do that.

7) Manual Testing

Mobile app testing is a complex process. Sometimes, real human experience can deliver the results you want. QA teams use manual testing to ensure that the final product really works as intended. With a specific role to play, manual testing is used to explore use-cases that may not be all that obvious. Additionally, we simply can't automate some types of tests… and shouldn't. These include:

- Physical interface tests
- Complex tests
- Exploratory testing

8) Automated Testing

As we've pointed out before, there are some cases where manual testing is the better option. However, some QA tests are either too tedious or too complex for human testers. That's why many apps are now practicing automated testing, executed continuously and alongside manual tests, to assure quality and release better products, faster.

A few automated testing best practices and challenges include:

- The thoughtful design, build, and maintenance of accurate test scripts
- The alignment and integration of existing engineering workflows with your automated testing process
- The creation and maintenance of your test automation framework, including infrastructure
- The management of test runs and setups
- Rigorous reviews to validate test results and defects
- Careful monitoring and rapid response to noise and flakey tests

9) Mobile-Device Testing

Mobile apps would not exist without hardware and operating systems. So, we also need to think about mobile-device testing. There are several testing types specific to mobile including:

- Interruptions – Interrupt testing evaluates how an app reacts to interruptions and if it resumes to its prior state. Common mobile app interruptions include loss of battery power, in incoming phone call or text, notifications, and app updates.
- Location-based Services (LBS) – Using geo-data from a mobile device, location-based services provide real-time information, entertainment or security. They are also used by consumers to "check in" while experiencing life on the go, like a visit to the local Starbucks or while attending a concert.
- Biometric – Mobile devices often include biometric sensors that include face recognition, fingerprint and hand geometry, iris recognition, and even DNA or insulin levels.
- NFC payments – Near Field Communications (NFC) allows mobile devices to communicate with a payment terminal enabling contactless payments.

The result of intelligent effort. As author John Ruskin exclaimed, "Quality is never an accident; it is always the result of intelligent effort."

Now that you have a better understanding of different mobile application testing types, it's important to create a plan of action. If you're ready to put in the effort and take your testing to a whole new level, schedule a quick demo today.

5. Conclusion

We can't ignore the significance of a smooth mobile application testing as part of a mobile app's success. Despite that, the mere occurrence of a testing strategy does not ensure the mobile application's quality and performance. By planning your selection of target devices through a mix of emulators and physical devices—combined with other mobile testing strategies—you move towards a successful release.

References

[1] Anthony I. Wasserman, "Software Engineering Issues for Mobile Application Development", Proceedings of the FSE/SDP workshop on Future of software engineering research, pp. 397-400, 2010.
[2] Sarafaraz Ahmad Momin, "Survey on Mobile Automation Testing Tools", International Journal of Application or Innovation in Engineering & Management, Volume 4, Issue 1, pp. 191-193, January 2015.
[3] https://testlio.com/blog/step-step-mobile-application-testing-process/
[4] https://aws.amazon.com/device-farm
[5] https://www.gurock.com/testrail/software-testing-tools
[6] https://www.nagarro.com/en/blog/post/18/are-you-selecting-the-best-tools-for-mobile-software-testing-checklist-
[7] https://www.outsource2india.com/services/software_testing.asp
[8] https://www.axelerant.com/resources/articles/7-ways-win-mobile-application-testing
[9] https://www.tutorialspoint.com/mobile_testing/mobile_testing_tutorial.pdf
[10] https://www.scribd.com/document/207690398/Mobile-App-Testing-PDF
[11] https://en.wikipedia.org/wiki/Mobile_application_testing
[12] https://smartbear.com/product/testcomplete/mobile-testing/
[13] https://www.edureka.co/blog/mobile-application-testing/
[14] https://saucelabs.com/blog/what-is-mobile-application-testing