Quality Assurance Testing from Beginner to Paid Professional

Part 1

Everything You Need to Know to Start a
Career in Manual and Automated QA Testing

Written By
Bolakale Aremu

Quality Assurance Testing from Beginner to Paid Professional, Part 1:
Everything You Need to Know to Start a Career in Manual and Automated QA Testing

Published in the United States

Limit of Liability/Disclaimer of Warranty
Both the author and publisher have made diligent efforts to ensure the accuracy of the information and instructions provided. However, they disclaim any liability for errors or omissions, including any potential damages arising from the use or reliance on this content. Readers use the information and instructions provided at their own risk. If this book includes code samples or references to technology that are governed by open-source licenses or other intellectual property rights, it is the reader's responsibility to ensure compliance with those licenses and rights.

Table of Contents

Part 1: Introduction to Software Testing

1. Overview of Quality Assurance

Every time you open an app on your phone or computer, have you ever stopped to wonder why it works so smoothly? It's not just luck—there's a dedicated team behind the scenes making sure everything functions perfectly. That's where **Manual QA Engineers** come in.

Here's how it works: Developers create the app, but before it reaches users like you, Manual QA Engineers step in to test every feature. They click buttons, navigate menus, and explore every function—just like a real user would. If they find errors, glitches, or anything that seems off, they report them so developers can fix the issues before the app goes live.

But testing isn't just about finding bugs. As a Manual QA Engineer, you also help improve the user experience. Should the checkout button be easier to find? Would the app feel more intuitive if a menu was placed differently? You'll be asking these questions and shaping how users interact with software.

The best part? **This is a high-demand job that pays well.** Companies are always looking for QA testers because every app, website, or software product needs thorough testing before release. Many companies also offer remote work options, giving you the flexibility to work from home or in a hybrid setup.

If you're interested in remote user testing jobs, here are a couple of reputable platforms you can explore:

1. Turing: This is a company that values top talent. Turing is looking for skilled remote software testing engineers to design and implement precise test procedures, ensuring optimal software performance. This is an excellent opportunity to collaborate with leading experts while working with top U.S. firms. https://www.turing.com/jobs/remote-software-testing-engineer.

2. **uTest**: A global community where testers can participate in various testing projects, from usability to functional testing, and get paid for their contributions. https://www.userlytics.com/user-experience-research/paid-ux-testing/.

3. **Test IO**: Offers freelance testers the opportunity to work on testing the latest apps and websites, providing flexibility and the chance to earn extra income. https://test.io/company/become-a-tester.

These platforms allow individuals to engage in testing activities and earn money, making them accessible options for those looking to start a career in software testing.

If you're preparing for a **QA Automation or SDET** interview, the following job-oriented Udemy course provides **expert-curated questions and detailed answers** to help you ace your interview with confidence:

Top 200+ QA Automation Interview Questions & Resume Tips - https://www.udemy.com/course/qa-automation-interview-prep/.

Now, if you're curious about how software works and love the idea of ensuring smooth user experiences, this practical book is the perfect starting point. You'll learn everything from the essential tools and testing processes to career growth strategies.

Packed with a bunch of **code samples** and **hands-on practice exercises**, this book allows you to apply what you learn in real-world scenarios, helping you build confidence and technical skills.

By the time you finish reading, you'll have a clear roadmap to launching your career in **Manual and Automated QA Testing**—no prior experience needed. With this book as your guide, you can take the first step toward your new career today!

1.1. What is Software Testing?

When you interact with a software application, whether it's a mobile app, a website, or a desktop program, you expect it to work flawlessly. However, software is complex, and mistakes can happen during development. That's where software testing comes in.

Software testing is the process of evaluating a software application to ensure it meets the required specifications and functions correctly. As a software tester, your job is to identify bugs, defects, or any issues that could affect the user experience. By systematically testing software, you help developers create reliable, high-quality applications that work as expected.

You can think of software testing as a quality assurance process that validates whether an application performs correctly under different conditions. It involves executing a program to find defects before it reaches the hands of users. Without proper testing, software may contain errors that could lead to security vulnerabilities, performance failures, or usability issues.

To be an effective tester, you must adopt a systematic approach. This means:

1. Understanding Requirements: Before you start testing, you need to know what the software is supposed to do. This involves reviewing specifications, user stories, or any documentation that outlines expected behavior.

2. Planning and Designing Tests: Once you understand the requirements, you create test cases—structured scenarios that outline what to test, how to test it, and what the expected outcome should be.

3. Executing Tests: This is where you run the software and check if it behaves as expected. You may perform manual testing (where you interact with the application yourself) or automated testing (where scripts and tools help run tests more efficiently).

4. Identifying and Reporting Defects: If you find a bug, you document it and report it to the development team so they can fix it.

5. Re-Testing and Regression Testing: After bugs are fixed, you test again to ensure that previous issues have been resolved and that no new problems have been introduced.

By following this structured approach, you contribute to the software development

lifecycle (SDLC) and help deliver a product that is functional, reliable, and user-friendly. Learn more about SDLC here: https://en.wikipedia.org/wiki/Systems_development_life_cycle. Whether you are testing a small mobile app or a large enterprise system, the goal remains the same—ensuring software quality and customer satisfaction.

Later, you'll explore the different types of software testing and how each plays a critical role in the development process.

1.2. Importance of Quality Assurance in Software Development

Software development is not just about writing code—it's about creating reliable, high-performing, and secure applications that meet user expectations. Quality Assurance (QA) plays a crucial role in achieving this goal. Without proper QA, software can be prone to bugs, security vulnerabilities, and performance issues that can lead to customer dissatisfaction, financial losses, and even legal consequences.

1.2.1. Why Quality Assurance Matters

1. Prevents Costly Errors
Fixing bugs in the later stages of development—or worse, after release—can be extremely expensive. QA helps identify issues early, reducing the cost and effort needed for fixes.

2. Ensures Software Reliability
Users expect software to function correctly under various conditions. QA ensures that applications perform consistently, minimizing crashes, errors, and unexpected behaviors.

3. Enhances Security
Cybersecurity threats are a major concern in today's digital world. QA includes security testing to identify vulnerabilities before hackers can exploit them, protecting sensitive user data.

4. Improves User Experience (UX)
A well-tested application provides a seamless and intuitive user experience. QA helps eliminate usability issues, making software more accessible and enjoyable to use.

5. Builds Customer Trust and Brand Reputation
High-quality software leads to satisfied customers who trust your product. A well-tested application enhances your company's reputation and encourages user loyalty.

6. Ensures Compliance with Industry Standards
Many industries require software to meet specific regulations and standards (e.g., ISO, GDPR, HIPAA). QA helps ensure compliance, preventing legal and regulatory issues.

1.2.2. The Role of QA in the Software Development Lifecycle (SDLC)

QA is not a one-time activity—it is integrated throughout the entire SDLC:

- **Requirement Analysis:** QA teams review requirements to ensure they are clear,

complete, and testable.

- **Design & Development:** Test plans and test cases are prepared alongside development.
- **Testing Phase:** Manual and automated tests are executed to detect defects.
- **Deployment & Maintenance:** QA continues with regression testing and monitoring for any post-release issues.

By making QA a priority in software development, organizations can build products that are not only functional but also robust, secure, and user-friendly.

1.3. Manual vs. Automated Testing – When to Use Each

When testing software, you have two primary approaches: **manual testing** and **automated testing**. Each method has its strengths and is suited for different testing scenarios. To ensure software quality, you must understand when to use manual testing and when automation is the better option.

1.3.1. What is Manual Testing?

Manual testing is the process of executing test cases **without** the use of automation tools. A tester manually interacts with the application, following predefined test cases or exploring the software dynamically to uncover defects.

When to Use Manual Testing

Manual testing is best suited for scenarios that require **human observation and adaptability**, such as:

1. **Exploratory Testing:** When testing a new feature, testers rely on intuition and experience to find defects that might not be covered by predefined test cases.
2. **Usability Testing:** Evaluating the user interface (UI) and user experience (UX) requires human judgment to ensure the software is intuitive and user-friendly.
3. **Short-Term or One-Time Testing:** If a feature is only tested once or twice, automating it may not be cost-effective.
4. **Ad-hoc Testing:** When no formal test cases exist, testers perform spontaneous testing to uncover unexpected defects.
5. **Small-Scale Projects:** If a project has limited scope, manual testing can be more practical than setting up automation frameworks.

Pros of Manual Testing:

- Requires no additional tools or setup
- Ideal for exploratory and usability testing
- Allows for quick adaptation to UI changes

Cons of Manual Testing:

- Time-consuming and repetitive
- Prone to human error
- Not scalable for large projects

1.3.2. What is Automated Testing?

Automated testing uses scripts and tools to execute test cases, compare actual outcomes with expected results, and generate reports. It is useful for repetitive and large-scale testing.

When to Use Automated Testing

Automation is most effective when dealing with repetitive, time-sensitive, or large-scale testing scenarios, such as:

1. **Regression Testing**: Ensures that recent code changes do not break existing functionality.
2. **Performance Testing**: Simulating thousands of users interacting with an application cannot be done manually.
3. **Load and Stress Testing**: Helps determine how software behaves under heavy usage.
4. **Repeated Test Scenarios**: Tests that need to be run frequently, such as login verification, can be automated to save time.
5. **Complex Scenarios with Large Data Sets**: Automating tests for applications that handle extensive data processing is more efficient than manual testing.

Pros of Automated Testing:

- Faster execution of test cases
- Increases accuracy and consistency
- Enables large-scale and repetitive testing

Cons of Automated Testing:

- Requires initial investment in tools and setup
- Cannot replace human intuition for exploratory or usability testing
- Scripts must be maintained as the application evolves