



GREEN UNIVERSITY OF BANGLADESH

**Industrial Training**  
**On**  
**A Practical Understanding of Software Quality Assurance**

**Submitted by**  
Sajid Rahman Rifan ( 212902017 )

*An industrial training report submitted to the Department of Computer Science & Engineering for the partial fulfillment of the degree of Bachelor of Science in Computer Science & Engineering*

<b>TRAINING PLACE :</b>	QA Harbor Ltd. House-470, Flat-5B, Road-31, Mohakhli DOHS, Dhaka Bangladesh, 1212
<b>TRAINING DURATION :</b>	26 February 2023 to 31 March 2023
<b>INDUSTRY MENTOR:</b>	Masdur Rahaman
<b>ACADEMIC SUPERVISOR:</b>	Md. Abu Rumman Refat
<b>REPORT DATE :</b>	19 May 2025

# Declaration

Declaring this training report to be founded only on my own research results, I make it clear. References are made to research materials discovered by other researchers. This work has not been submitted for a degree before, either in full or in part.

---

Sajid Rahman Rifan

ID: 212902017

# Certificate

This is to certify that the industrial training report entitled Acquired pragmatic knowledge of software development has been prepared and submitted by **Sajid Rahman Rifan** in partial fulfillment of the requirement for the degree of Bachelor of Science in Computer Science and Engineering.

---

Md. Abu Rumman Refat  
Academic Supervisor

Accepted and approved in partial fulfillment of the requirement for the Bachelor of Science degree in Computer Science and Engineering.

---

Ms. Shamima Akter  
Member

---

Md. Riad Hassan  
Member

# Acknowledgments

I want to thank everyone who made my internship at QA Harbor Limited enjoyable and meaningful. First, I am very grateful to **Masudur Rahaman**, my industry mentor, for his guidance, support, and advice throughout the internship. His knowledge and feedback helped me improve my skills in **software testing**. I also want to thank **Md. Abu Rumman Refat**, my academic supervisor, for his guidance and support during this time. His encouragement and leadership helped me learn and grow. I am thankful to all the team members at QA Harbor Limited for welcoming me warmly and sharing their knowledge. Their help and support made my internship a great experience. Finally, I want to thank QA Harbor Limited for giving me the chance to join their organization. This internship, held from **20/03/2025 to 10/05/2025**, at Level 5, House No: 470, Road No: 31, Mohakhali DOHS, Dhaka, has been life-changing. It gave me valuable skills and experiences that will help me in my career. Thank you to everyone who supported me and made my internship successful.

# **Abstract**

During this industrial training at QA Harbor Limited, I worked as an intern in It Support. During this training period, I was involved in many of the services here. The training session focused on setting up and configuring hardware and software, caring for IT infrastructure, and providing technical support for issues with hardware and software. The objectives of the training program are covered in the document, as well as the tools, software, and hardware that will be used. In addition to discussing the challenges encountered and how they were overcome, the primary knowledge and skills acquired during the training period are also highlighted. The report provides further information on the company that provided the training, such as the types of IT support services provided, the business culture, and the working environment. I have a solid technical foundation thanks to my industrial training as an IT support engineer.

## TABLE OF CONTENTS

<b>Declaration</b> . . . . .	ii
<b>Certificate</b> . . . . .	iii
<b>Acknowledgments</b> . . . . .	iv
<b>Abstract</b> . . . . .	v
<b>List of Figures</b> . . . . .	xi
<b>List of Tables</b> . . . . .	1
<b>1 Introduction</b> . . . . .	1
1.1 Introduction . . . . .	1
1.2 Motivation of Training . . . . .	1
1.3 Objective of Training . . . . .	2
1.4 Identifying The Gap Between Academia and Industry . . . . .	3
1.4.1 Theoretical Knowledge vs. Practical Application . .	4
1.4.2 Industry Tools and Technologies . . . . .	4
1.4.3 Problem-Solving in Real-World Scenarios . . . . .	4
1.5 Internship Report Organization . . . . .	5
1.6 Conclusion . . . . .	5
<b>2 Organization Overview</b> . . . . .	6
2.1 Introduction . . . . .	6
2.2 Company Profile . . . . .	6
2.3 Company Structure/Organogram . . . . .	7

2.4	List of Products/Services . . . . .	9
2.4.1	QA Services . . . . .	9
2.4.2	Performance Testing . . . . .	9
2.4.3	Security Testing . . . . .	9
2.4.4	Usability Testing . . . . .	10
2.4.5	Consulting Services . . . . .	10
2.4.6	Specialized Testing . . . . .	10
2.5	List of Clients . . . . .	10
2.6	Collaboration Environment . . . . .	11
2.7	The Mission of QA Harbor Ltd : . . . . .	11
2.8	The Visions of QA Harbo Ltd : . . . . .	12
2.9	The Values of QA Harbor Ltd : . . . . .	13
2.10	Conclusion . . . . .	14
<b>3</b>	<b>Specific details on training activities . . . . .</b>	<b>15</b>
3.1	Introduction . . . . .	15
3.1.1	Scheduling . . . . .	16
3.2	First Week . . . . .	16
3.2.1	Software Quality Assurance (SQA) Overview . . . . .	16
3.2.2	Seven Principles of Software Testing . . . . .	17
3.2.3	Test Management & SDLC . . . . .	18
3.2.4	Software Testing Life Cycle (STLC) . . . . .	18
3.2.5	V-Model Process . . . . .	20
3.2.6	Task Test Case (QA Harbor Website) . . . . .	20
3.3	Second Week. . . . .	22
3.3.1	Test case formats, and issue/Bug reporting templates . . . . .	22
3.3.2	Agile development test strategy . . . . .	22
3.3.3	Provided a Task to Generate 20 Test Cases and Make Test Case Report for Any Software . . . . .	23
3.3.4	Understanding Agile Methodologies . . . . .	24

3.3.5	Core principles of Scrum . . . . .	25
3.3.6	SCRUM Structure . . . . .	26
3.3.7	Scrum Roles, Events, and Artifacts . . . . .	26
3.3.8	Playwright Automation . . . . .	27
3.3.9	Writing Our First Playwright Test . . . . .	28
3.4	Create a Test File . . . . .	28
3.5	Pseudocode for Playwright Test . . . . .	28
3.6	Automation Test Output . . . . .	29
3.7	GitHub operations . . . . .	30
3.7.1	(My project-Automation testing using Playwright li- brary in python Largest Job Site in Bangladesh, Search Jobs — Bd-jobs.com . . . . .	30
3.7.2	Playwright and perform automation testing in Swag Labs, website by doing login and add to cart . . . . .	41
3.8	Fourth Weeks . . . . .	41
3.8.1	Performance Testing . . . . .	41
3.8.2	Performance Testing Example . . . . .	42
3.8.3	Performance Testing Workflow in JMeter for QA Harbor Web Site . . . . .	43
3.8.4	Test Configuration . . . . .	43
3.9	API Testing . . . . .	45
3.9.1	What is API? . . . . .	45
3.9.2	Types of API . . . . .	45
3.9.3	API Testing . . . . .	47
3.9.4	Types of API Testing . . . . .	47
3.9.5	Steps in API Testing . . . . .	48
<b>4</b>	<b>Learning from the Activities of Internship . . . . .</b>	<b>50</b>
4.0.1	Understanding Software Quality Assurance . . . . .	50
4.1	Key SQA Activities . . . . .	50



4.1.1	Requirement Analysis . . . . .	50
4.1.2	Test Case Design and Execution . . . . .	51
4.1.3	Automation Testing . . . . .	51
4.2	Exposure to Testing Techniques . . . . .	51
4.3	Skills Practice through Tool Work . . . . .	51
4.4	Collaboration and Communication . . . . .	52
<b>5</b>	<b>Conclusion . . . . .</b>	<b>53</b>
5.1	Conclusion . . . . .	53
5.1.1	Skills Developed and Experiences Gained . . . . .	53
5.1.2	The Gap Between Industry and Academia . . . . .	54
5.2	Challenges during Internship . . . . .	55
5.2.1	Adapting to New Testing Tools and Methodologies . . . . .	55
5.2.2	Time Management and Scheduling . . . . .	55
5.2.3	Collaborate with Team Members and Mentors . . . . .	55
5.2.4	Problem-Solving Under Pressure . . . . .	56
5.2.5	Learning New Tools and Technologies . . . . .	56
5.2.6	Sustainability of Learning . . . . .	56
5.3	Future Directions for Upcoming Internship . . . . .	57
5.3.1	Advanced Test Automation . . . . .	57
5.3.2	Mobile and Cross-Platform Testing . . . . .	57
5.3.3	API Testing . . . . .	57
5.3.4	Performance and Load Testing . . . . .	57
5.3.5	Security Testing . . . . .	57
5.3.6	Cloud Testing . . . . .	58
5.3.7	Visual Regression Testing . . . . .	58
	<b>References . . . . .</b>	<b>59</b>

# List of Figures

2.1	Company Structure QA Harbor Limited . . . . .	8
3.1	Gantt Chart . . . . .	16
3.2	Seven principles of software testing . . . . .	18
3.3	6 Phase of the Software Development Life Cycle . . . . .	19
3.4	Test Case QA Harbor website . . . . .	21
3.5	5W1H . . . . .	22
3.6	Manual Test Case(Bdjobs Website) . . . . .	24
3.7	SCRUM Process . . . . .	25
3.8	SCRUM structure . . . . .	26
3.9	Automation test Output . . . . .	29
3.10	GitHub Version Control . . . . .	30
3.11	Main File . . . . .	30
3.12	Automation Testing (Bd-jobs Website) . . . . .	31
3.13	Automation Testing (Bd-jobs Website) . . . . .	32
3.14	Automation Testing (Bd-jobs Website) . . . . .	33
3.15	Automation Testing (Bd-jobs Website) . . . . .	34
3.16	Automation Testing (Bd-jobs Website) . . . . .	35
3.17	Automation Testing (Bd-jobs Website) . . . . .	36
3.18	Automation Testing (Bd-jobs Website) . . . . .	37
3.19	Automation Testing (Bd-jobs Website) . . . . .	38
3.20	Automation Testing (Bd-jobs Website) . . . . .	39
3.21	Automation Testing (Bd-jobs Website) . . . . .	40

3.22	Sauce demo . . . . .	41
3.23	Performance Testing . . . . .	42
3.24	Performance Testing . . . . .	43
3.25	Performance Testing . . . . .	44
3.26	API Work Process . . . . .	45
3.27	SOAP API . . . . .	46
3.28	REST API . . . . .	46
3.29	API Testing Workflow . . . . .	48

# Chapter 1

## Introduction

### 1.1 Introduction

An internship is an organized program that offers people—typically students or recent graduates—practical work experience in a particular subject or business. It gives participants the chance to learn practical information, hone their professional abilities, and investigate potential career possibilities. There are many different industries where internships are available, including business, technology, healthcare, finance, journalism, and many more. They can be full-time or part-time and are typically transitory jobs lasting a few weeks to many months. An internship's main objective is to close the knowledge gap between the classroom and the workplace. Here I was doing an internship in the networking field. Overall, internships provide a singular opportunity for people to learn more about their chosen area, obtain real-world experience, and lay the groundwork for future professional success.

### 1.2 Motivation of Training

Each person's reasons for wanting to pursue an internship will be different, however the following are some typical ones:

- **Professional Development:** Internships offer a platform to hone professional abil-

ities pertinent to a given business or sector.

- **Career Exploration:** A valuable opportunity to investigate various businesses and career pathways is provided by internships.
- **Networking:** Internships give you the ability to create professional networks and meet with people in your field.
- **Resume Enhancement:** Experience from a relevant internship can greatly improve a resume.
- **Skill Development:** Through internships, people can learn and develop new talents that employers want.
- **Process Improvement :** Standardization: Training promotes standardized testing processes and practices, leading to more consistent and reliable results. Agile and DevOps Integration: Understanding how testing fits into Agile and DevOps methodologies is essential for modern software development practices.
- **Collaboration and Communication:** Team Dynamics: Training fosters better collaboration among team members, as testers learn to communicate effectively with developers, product managers, and other stakeholders. Shared Understanding: A common knowledge base helps align the team's goals and expectations regarding software quality.

## 1.3 Objective of Training

Already I said that I am doing an internship in the Software Testing(STQA). Depending on a person's individual goals and aspirations, the goal of a networking internship can change.

- Develop your technical proficiency with testing tools and procedures.
- Understanding the Software Development Lifecycle (SDLC): Acquire knowledge of the stages and integration testing.

- Apply your theoretical understanding to actual testing situations to gain practical experience.
- Cooperation: Enhance communication and cooperation between cross-functional teams.
- Feedback: For ongoing development, take constructive criticism to heart and act upon it.
- Documentation: Draft and keep up-to-date test cases, defect reports, and plans.
- Job Exploration: Determine your hobbies and possible software testing job routes.
- Project Contribution: Add value to projects by improving software quality through effective testing.

## 1.4 Identifying The Gap Between Academia and Industry

The inequalities or variations between the information, abilities, and practices emphasized in academic contexts and those required in actual industry environments are referred to as the "gap between industry and academia." Several causes contribute to this disparity, including:

- **Practical Application:** Academic education usually focusses a higher emphasis on theoretical understanding and fundamental ideas, even if the industry demands practical application and hands-on skills. Even if academic programs give students a solid theoretical foundation, they may not give them all the real-world experience they need to handle problems at work.
- **Pace of Change:** As a result of technology breakthroughs, market demands, and shifting trends, industries are always changing. On the other hand, updating academic courses to reflect the most recent advances in the industry can take longer.

- **Industry-Specific Tools and Technologies :** Specific tools, systems, and technologies that may not be thoroughly addressed in academic degrees may be used by the industry. Graduates might not be familiar with the tools and technologies that are often used in the sector, necessitating further training or self-study to close the knowledge gap.
- **Collaboration and Teamwork :** In professional settings where people work together to achieve common goals, strong cooperation and collaboration abilities are usually required. Academic environments, on the other hand, could place more emphasis on individual achievement and assessment.

#### **1.4.1 Theoretical Knowledge vs. Practical Application**

Academic study puts heavy emphasis on fundamental theoretical concepts whereas professional settings require practical utilization of these academic elements. My theoretical training included Black Box and White Box testing methods although the use of these methods within Agile project implementations demanded practical experience.

#### **1.4.2 Industry Tools and Technologies**

In academia, exposure to industry-standard tools is limited. During my internship, I worked with tools like Postman, JMeter, and Selenium, which were either not covered or only briefly introduced in my academic studies. These tools are essential for efficient software testing and automation.

#### **1.4.3 Problem-Solving in Real-World Scenarios**

The industry presents more complex and undefined problems than those that appear as neatly defined cases within academia. When real-world situations require it, diverse thinking methods and adaptability become fundamental.

## 1.5 Internship Report Organization

This report is structured as follows:

- **Introduction:** Overview of the internship, objectives, and the company.
- **Identifying the Gap Between Academia and Industry:** Reflection on the differences between academic learning and industry practices.
- **Challenges and Solutions:** Discussion on the challenges faced and the solutions implemented during the internship.
- **Benefits and Importance:** Insights into the skills gained and the overall value of the internship.
- **Conclusion:** Summary of the key takeaways and overall.

## 1.6 Conclusion

Over all an internship is very helpful to the student. Students can explore their knowledge from it. They can choose their career from it.



# **Chapter 2**

## **Organization Overview**

### **2.1 Introduction**

In this chapter, we provide an overview of QA Harbor Limited, outlining its background, mission, vision, and the core values that drive the company. As a leading provider of software quality assurance (QA) services, QA Harbor is committed to delivering exceptional quality and ensuring that every project meets the highest standards. Our team of experts works tirelessly to support clients by offering reliable, flexible, and tailored QA solutions to meet the unique demands of each project. Our approach centers on Quality As A Service (QAAS), allowing us to seamlessly integrate into clients' development cycles and contribute to the creation of bug-free, high-performing software. The following sections delve into the key components that make QA Harbor a trusted partner in the tech industry

### **2.2 Company Profile**

QA Harbor Limited is a quality assurance service provider based in Dhaka, Bangladesh, specializing in software testing and process optimization. Established in 2016, the company operates with staff members and delivers services including security testing, usability testing and test automation. The firm uses many testing approaches to cover software applications completely with manual testing together with automated test-

ing together with performance testing together with regression testing. QA Harbor Limited deploys contemporary tools and testing frameworks to both optimize testing performance and accuracy to sustain speedier release deadlines with top-notch quality. Their client base shows their ability to work with startups and SMEs along with large enterprises due to their experience with diverse project sizes and complexities. QA Harbor Limited shows quality commitment by following international standards and certifications to maintain process alignment with worldwide best practices. They perform frequent internal audits combined with assessments to sustain superior levels of service quality. The company actively collects feedback from their customers to improve service quality and strengthen overall customer satisfaction rates. The company's lead team exhibits professional expertise combined with varied experience in software development and quality assurance which creates strategic guidance and promotes innovative work culture. Their goal is to become a premier QA testing company for the entire region known for their deep technical understanding and dependable service combined with superior testing performance. QA Harbor Limited considers diversity alongside inclusion as a primary objective which leads to better creative solutions and problem-solving results. They proactively search for workers from diverse demography backgrounds while creating an environment where all workers develop their full contribution potential.

## 2.3 Company Structure/Organogram

QA Harbor Limited operates under a structured hierarchy designed to ensure smooth communication and efficient project delivery. Below is the organizational chart of the company:

- **Chairman** : The top authority overseeing the entire organization
- **Managing Director (MD)** :Reports to the Chairman and manages overall operations.
- **Directors and CTO**

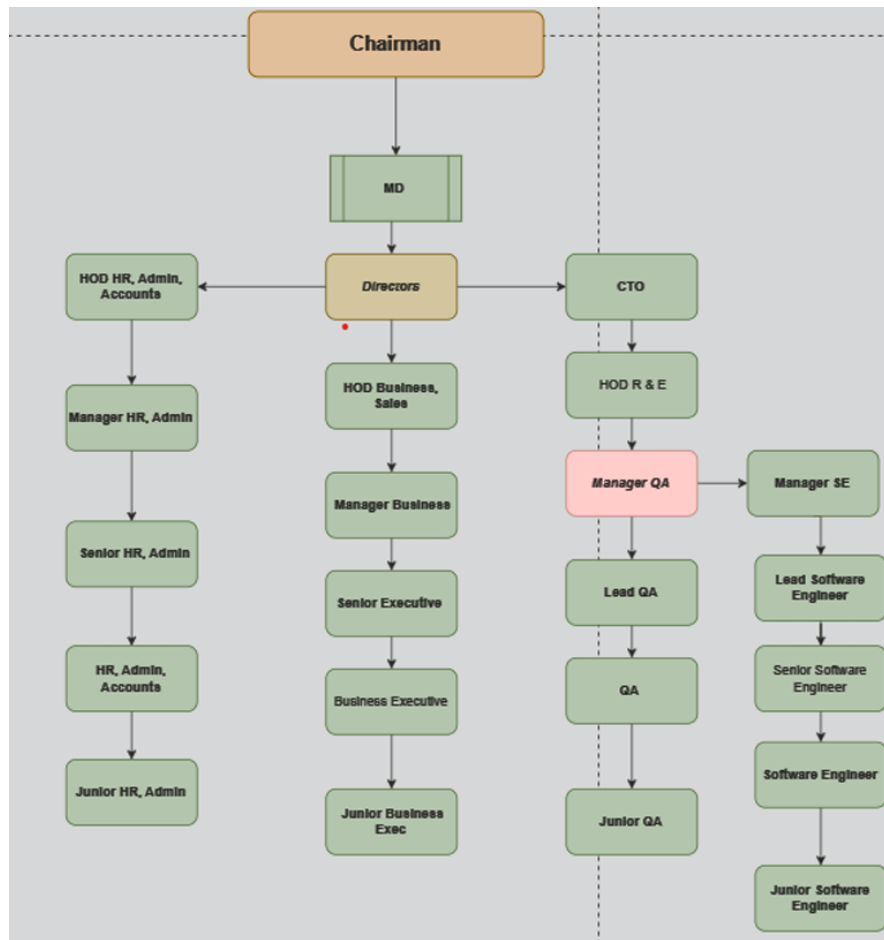


Figure 2.1: Company Structure QA Harbor Limited

a. **Directors** Handle business, sales, HR, admin, and accounts.

b. **CTO** Focuses on research and technology development.

• **Department :**

a. **HOD Business, Sales:** Leads the business team, including managers, senior executives, and junior business executives

b. **HOD R & E (Research and Engineering):** Supervises the QA and software engineering teams, with managers, leads, and junior roles.

c. **HOD HR, Admin, Accounts:** Manages HR and admin teams, with managers, senior staff, and junior roles.

The company's organizational structure includes the following key departments:

- **Management Team:** Responsible for strategic decision-making and overseeing overall company performance.
- **Quality Assurance Team:** Focuses on manual and automated testing to ensure software quality and client satisfaction.
- **Development Team:** Works on internal projects and provides development support for clients.

## 2.4 List of Products/Services

**QA Harbor** provides various software quality assurance solutions to deliver excellent, protected and fast software results. We handle each stage of the software development lifecycle.

### 2.4.1 QA Services

- **Manual Testing:** Full bug finding together with usability verifications.
- **Automated Testing:** Robust test completeness through automated testing procedures.

### 2.4.2 Performance Testing

- **Load & Stress Testing:** Assessing Application capability to manage heavy workload traffic.
- **Scalability Testing:** Verifying software ability to support expanding operations

### 2.4.3 Security Testing

- **Vulnerability & Penetration Testing:** Security flaw detection and repair.

- **Compliance Testing:** Security standard compliance confirmation.

#### **2.4.4 Usability Testing**

- **UX& Accessibility Testing:** Arranging for effortless user access to the website.

#### **2.4.5 Consulting Services**

- **QA Strategy & Process Improvement:** QA Strategy Process Improvement: Meet your needs for enhanced testing methods and strategies
- **Agile QA Consulting:** Integrating QA into Agile environments. QA Team Augmentation
- **Dedicated QA Experts:** On-demand skilled professionals to join your team.

#### **2.4.6 Specialized Testing**

- **Mobile & Cloud Testing:** Ensuring the functionality and security of mobile and cloud apps.

### **2.5 List of Clients**

At **QA Harbor**, we are proud to have partnered with a diverse range of clients across various industries. Our clients trust us to provide top-tier quality assurance services that ensure their software meets the highest standards. Below is a list of some of our valued clients:

- **Styline Collection:** Partner in software quality assurance for e-commerce platforms.
- **Tukana ISPERP:** Provided comprehensive testing solutions for their enterprise software.

- **Ontor Hazary:** Client in the tech industry, where we supported software development through continuous testing.
- **Jucana ISPERP:** Delivered specialized QA services for web and mobile applications

## 2.6 Collaboration Environment

At **QA Harbor**, we ensure seamless communication and efficient teamwork with our clients to deliver high-quality software.

### Key Elements:

- **Clear Communication:** Dedicated channels for regular updates and feedback.
- **Agile Approach:** Iterative testing with flexibility to adapt quickly.
- **Client Involvement:** Active engagement throughout the testing process.
- **Dedicated Teams:** Expert QA teams focused on each project.
- **Advanced Tools:** Utilizing top testing tools like Postman and JMeter.
- **Flexibility:** Adapting to client working hours and requirements.

[1]

## 2.7 The Mission of QA Harbor Ltd :

The mission of **QA Harbor Ltd.** is to deliver high-quality, reliable, and innovative software testing solutions that empower organizations to launch flawless digital products with confidence. We strive to be a trusted QA partner by ensuring excellence at every stage of the software development lifecycle—through expert testing, continuous improvement, and a commitment to client success.

- **Ensure Flawless Software Delivery** To help clients release reliable, bug-free digital products through comprehensive testing strategies.

- **Drive Innovation in QA** To adopt the latest tools and methodologies that improve efficiency, speed, and quality in software testing.
- **Support Agile and DevOps** To integrate seamlessly into modern development workflows, enabling continuous testing and faster releases.
- **Build Long-Term Partnerships** To act as a trusted QA partner, providing value through collaboration, transparency, and consistent performance.
- **Promote Skill and Excellence** To foster a team of skilled professionals committed to continuous learning and quality improvement.

**”Empowering flawless software through expert testing and trusted quality assurance.”**

## **2.8 The Visions of QA Harbo Ltd :**

QA Harbor Ltd. envisions delivering the best customer satisfaction by rapidly providing cost-effective testing services through continuous improvement, driven by teamwork, innovation, and automation efficiency across the Software Testing Life Cycle (STLC) and Software Development Life Cycle (SDLC).

- **Deliver the Best Customer Satisfaction** Our foremost goal is to ensure that our clients are fully satisfied with our services, aiming to exceed their expectations consistently.
- **Rapid and Cost-Effective Testing Services** We strive to provide swift and affordable testing solutions that meet the dynamic needs of our clients, ensuring timely delivery without compromising quality.
- **Emphasize Continuous Improvement** We are committed to the ongoing enhancement of our processes and methodologies, fostering a culture of learning and adaptation to stay ahead in the industry.

- **Foster Teamwork, Innovation, and Automation Efficiency** By promoting collaboration, encouraging innovative approaches, and leveraging automation, we aim to optimize our testing processes for maximum efficiency and effectiveness.
- **Apply Principles Across STLC and SDLC.**Our vision encompasses the integration of these principles throughout the entire Software Testing Life Cycle and Software Development Life Cycle, ensuring comprehensive quality assurance.

## 2.9 The Values of QA Harbor Ltd :

- **Quality Excellence :** We prioritize delivering superior quality products and services that meet and exceed client expectations, ensuring reliability and performance.
- **Customer Satisfaction :** Our clients' satisfaction is paramount. We strive to build lasting relationships by understanding their needs and delivering tailored solutions.
- **Continuous Improvement :** We embrace a culture of continuous learning and process enhancement to stay ahead in the dynamic field of software quality assurance.
- **Innovation and Efficiency :** By fostering innovation and leveraging efficient methodologies, we aim to provide cost-effective and timely solutions without compromising quality.
- **Teamwork and Collaboration :** We believe in the power of teamwork. Our collaborative approach ensures diverse perspectives contribute to the best outcomes for our clients.
- **Integrity and Transparency** We conduct our business with the highest level of integrity, maintaining transparency in all our dealings to build trust with our stakeholders.



- **Commitment to Excellence** Our dedication to excellence drives us to consistently deliver high-quality services and to be a trusted partner in our clients' success.

Overall, these principles represent QA Harbor Ltd.'s dedication to providing its clients with great value, cultivating a happy and supportive work environment for its employees, and using technology to positively benefit society.

## **2.10 Conclusion**

This report was written based on my internship experience. I've finished my internship at QA Harbor Company in IT support. As an assistance care Software Company, QA Harbor Limited, different. There are different services offered here. The aforementioned IT company, Help Care, has acquired knowledge of all the services. In light of this understanding, I prepared the report.

## Chapter 3

### Specific details on training activities

#### 3.1 Introduction

This report summarizes my four-week internship at QA Harbors Limited, starting November 1, 2024, focusing on Software Quality Assurance (SQA). Under expert guidance, I gained practical knowledge in testing and quality assurance, **covering Types of Testing, Testing Fundamentals, and Defect Management**. Key tasks included creating **test cases**, generating **test reports**, defect resolution, and using tools like JMeter, Playwright, and Selenium. I also gained professional skills such as preparing a **student CV**, attending mock viva sessions, and using **LinkedIn, GitHub, and JIRA**. This internship strengthened my technical and problem-solving skills, providing a solid foundation for a career in SQA.

### 3.1.1 Scheduling

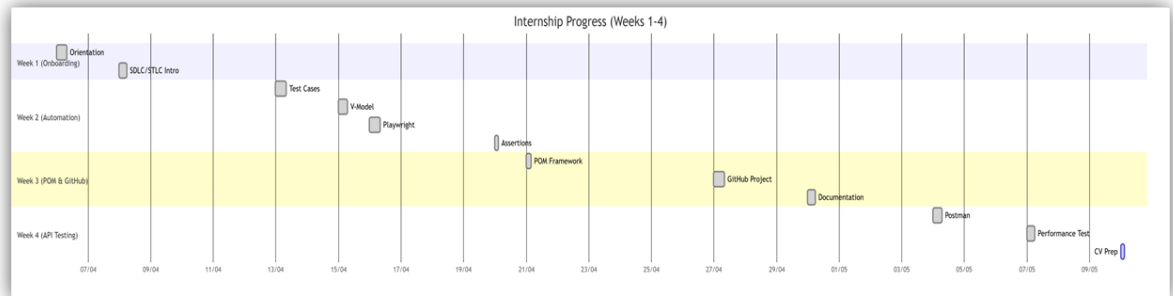


Figure 3.1: Gantt Chart

- Week 1:** Learned fundamental software testing elements and tested control techniques together with strategy management.
- Week 2:** Learned advanced Playwright test automation features.
- Week 3:** Tested APIs with Postman tools and investigated and tested SOAP and REST web services.
- Week 4:** Explored Playwright's Page Object Model (POM) implementation and executed initial load testing and performance testing.

## 3.2 First Week

### 3.2.1 Software Quality Assurance (SQA) Overview

#### Concept of Quality

- **Software Quality:** The extent to which a product meets stakeholder expectations and performs reliably

#### Software and Its Quality

- **Software:** A set of instructions enabling specific tasks on a computer.

- **Software Quality:** Measured by how well the software meets specified requirements and user needs

### **Types of SQA**

- Encompasses activities like planning, testing, code reviews, and adherence to standards.

### **Economic Importance and Causes of Software Failures**

- **Economic Importance:** Essential for operations in telecom, banking, and traffic control systems.
- **Failures:** Caused by human error or external factors (e.g., power outages, environmental interference).

### **Testing for Quality Improvement**

- **Testing:** Identifies defects and improves product quality and development processes.

## **3.2.2 Seven Principles of Software Testing**

- a. **Testing shows the presence of defects:** Proves bugs exist but cannot confirm their absence.
- b. **Exhaustive testing is impossible:** Focus on high-risk areas rather than testing every scenario.
- c. **Early testing:** Detects defects early in the development cycle.
- d. **Defect clustering:** Issues often occur in complex areas, requiring targeted testing.
- e. **Pesticide paradox:** Update test cases regularly to find new defects.

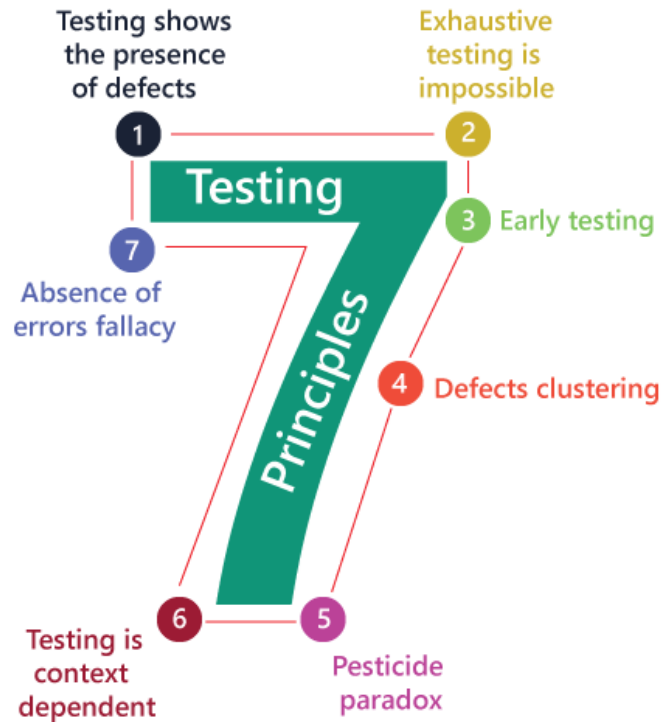


Figure 3.2: Seven principles of software testing

<https://www.nititest.com/seven-principles-of-software-testing/>

[2]

### 3.2.3 Test Management & SDLC

[3]

Test management is a vital aspect of ensuring software quality, involving the planning, tracking, and reporting of testing activities within the **Software Development LifeCycle (SDLC)**. The Software Testing Life Cycle (STLC) is a subset of SDLC that focuses specifically on the testing phases to ensure quality throughout development.

### 3.2.4 Software Testing Life Cycle (STLC)

#### 6 Phase of the Software Development Life Cycle

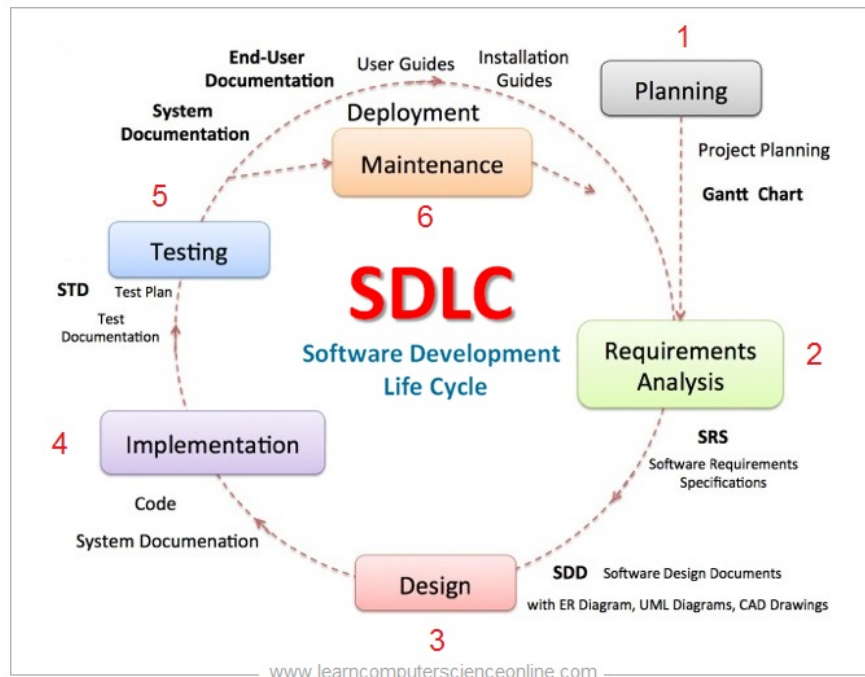


Figure 3.3: 6 Phase of the Software Development Life Cycle

<https://www.learncomputerscienceonline.com>

- **Requirement Gathering:** Assemble client needs, eliminate confusion and document these requirements in the SRS document.
- **Software design:** Develop architectural blueprints from SRS.
- **Software implementation:** Code actualize the planned design.
- **Software testing:** Use SRS to find software problems and match customer needs.
- **Software deployment:** Distribute finished software to production environment or perform User Acceptance Testing(UAT) tests.
- **Post-deployment Maintenance:** Correct software problems and add new features.

The SDLC creates a planned and organized way to build software products.

### 3.2.5 V-Model Process

The **V-Model(Verification and Validation)** is a software development approach that links every development step to its corresponding test phase. The main phases of the V-Model include:

- **Requirements Analysis:** Understand testing requirements.
- **System Design:** Organize testing phases and connectivity.
- **Architecture Design:** Build components so they can be verified.
- **Module Design:** Write test cases of modules.
- **Codification:** Develop and test unit code..
- **Single Unit Testing:** Test one module at a time.
- **Integration Testing:** Test component interactions.
- **System Testing:** Verify overall system behavior.
- **Acceptance Testing:** Ensure the system meets user needs.

### 3.2.6 Task Test Case (QA Harbor Website)

A test case report documents the details of individual test cases executed during the testing process. A typical test case report includes:

- **Test Case ID:** A unique identifier for the test case
- **Test Description:** A brief description of the functionality tested.
- **Test Steps:** The steps taken to execute the test.
- **Expected vs Actual Result:** The expected and actual outcomes to compare success.
- **Pass/Fail Status:** Indicates whether the test passed or failed

Test ID	Test Name	Steps	Expected Result	Actual Result
TC_001	Homepage Load	1. Open a browser. 2. Enter URL: https://qaharbor.com/	Homepage loads within 3 seconds without issues.	Homepage loaded successfully. Layout and navigation displayed correctly.
TC_002	QA Jobs Button	1. Locate QA Jobs button. 2. Click the button.	QA Jobs page loads without errors.	QA Jobs page loaded successfully.
TC_003	Candidate Registration	1. Go to QA Jobs page. 2. Hover over Sign-Up menu. 3. Click Candidate Registration.	Candidate Registration page loads without errors.	Page loaded and worked as expected.
TC_004	Sign-In Functionality	1. Click on Sign-In button.	Sign-In page loads and login works without errors.	403 Forbidden Error occurred.
TC_005	Services Menu	1. Hover over Services menu. 2. Select "Testing QA Services" from dropdown.	Testing QA Services page loads without errors.	Testing QA Services page loaded successfully.

Figure 3.4: Test Case QA Harbor website

### Summary of Results

- Passed Tests:TC001,TC002,TC003,TC005 successfully verified their respective functionalities.
- Failed Test: TC 004 encountered a "403 Forbidden Error," highlighting an issue in the Sign-In functionality



### 3.3 Second Week.

#### 3.3.1 Test case formats, and issue/Bug reporting templates

This is formats of TEST CASES. **Test Case ID,Test Scenario,Test Steps,Test Data,Expected Result,Actual Result,Status (Pass/Fail),Comments.**

#### 3.3.2 Agile development test strategy

The overall approach and objective of the test project are outlined in the test strategy. Explain the testing stage that is being conducted at the moment, the kinds of tests (functional, performance, coverage, etc.) that are being conducted at each level, the testing personnel arrangements, etc.

The **5W1H method** allows us to develop a reasonable test strategy based on the test's goal, scope, start and end times, people organization, and tools.



Figure 3.5: 5W1H

<https://www.alamy.com/5w1h>

- Why: What is the goal of the test, and why do you wish to take it?

- What: Test scope and content, what to test, and the test focus (e.g., RBT based on demand testing)
- When: the test's start and end times, taking into account the variables that impact time.
- where: environmental geology, defect storage, and the location of relevant document storage
- Who: Setting up the testers
- How: Which testing technique and tool to employ

### 3.3.3 Provided a Task to Generate 20 Test Cases and Make Test Case Report for Any Software

Manual Test Case on **Bd-jobs website**: A typical test case report includes:

- **Test Case ID:** A unique identifier for the test case.
- **Test Description:** A brief description of the functionality tested.
- **Test Steps:** The steps taken to execute the test.
- **Expected vs Actual Result:** The expected and actual outcomes to compare success.
- **Pass/Fail Status:** Indicates whether the test passed or failed.

TC_ID	Module Name	Submodule Name	Pre-condition	Test Case Title	Steps	Test Data	Expected Result
TC_RDMOBS_001	Login	User Authentication	User is on the Login page	Verify login with valid credentials	1. Navigate to login page 2. Enter valid credentials 3. Click login	Email: rifurrahma729@gmail.com, Password: correct123	User should be logged in
TC_RDMOBS_002	Login	User Authentication	User is on the Login page	Verify login with invalid credentials	1. Navigate to login page 2. Enter wrong credentials 3. Click login	Email: rifurrahma729@gmail.com, Password: wrongpass	Error message should appear
TC_RDMOBS_003	Job Search	Search Bar	User is on the Job Search page	Search for Software Engineer jobs	1. Go to homepage 2. Type 'Software Engineer' 3. Click search	'Software Engineer'	Relevant job listings should be displayed
TC_RDMOBS_004	Job Filters	Location Filter	User is on the Job Filters page	Apply location filter	1. Perform a job search 2. Select 'Dhaka' in location filter 3. Apply filter	Location: Dhaka	Only jobs in Dhaka should be shown
TC_RDMOBS_005	Job Listing	Job Details	User is on the Job Listing page	Verify job details visibility	1. Click on any job listing 2. Observe job details	Any job listing	Title, company, location, etc.
TC_RDMOBS_006	Job Application	Apply Job	User is on the Job Application page	Apply to a job successfully	1. Click on 'Apply' for a job 2. Fill out the form 3. Submit	Resume, Cover letter	Application should be successful
TC_RDMOBS_007	Resume Upload	Profile Management	User is on the Resume Upload page	Upload resume	1. Login 2. Navigate to profile 3. Upload resume	Resume file (PDF/DOC)	Resume should be uploaded
TC_RDMOBS_008	Account Registration	User Signup	User is on the Account Registration page	Register new account	1. Go to registration page 2. Fill details 3. Submit	Name, Email, Password	Account should be created
TC_RDMOBS_009	Account Settings	Profile Edit	User is on the Account Settings page	Edit user profile	1. Login 2. Go to profile settings 3. Edit and save	Phone number update	Profile should be updated
TC_RDMOBS_010	Job Alerts	Alert Settings	User is on the Job Alerts page	Set up job alert	1. Login 2. Go to Job Alerts 3. Enter criteria 4. Save alert	Keyword: Developer, Location: Dhaka	Job alert should be saved
TC_RDMOBS_011	Job Description	Job Detail View	User is on the Job Description page	View full job description	1. Click on job listing 2. View details	Any job	Full description visible
TC_RDMOBS_012	Employer Profile	Company Page	User is on the Employer Profile page	View employer profile	1. Click on company name from job listing 2. View details	Any listed employer	Employer profile page open
TC_RDMOBS_013	Password Recovery	Forgot Password	User is on the Password Recovery page	Recover password via email	1. Click 'Forgot Password' 2. Enter email 3. Check email	Registered email	Password reset email sent
TC_RDMOBS_014	Mobile Responsiveness	UI Display	User is on the Mobile Responsiveness page	Check mobile layout	1. Open site on mobile 2. Browse key features	Any mobile device	Site should display correctly
TC_RDMOBS_015	Job Recommendations	Suggestions	User is on the Job Recommendations page	Check job recommendations	1. Login	Logged-in user	Relevant jobs displayed

Figure 3.6: Manual Test Case(Bdjobs Website)

### 3.3.4 Understanding Agile Methodologies

**Agile Methodology** Agile is a flexible and collaborative software development approach that emphasizes iterative progress, customer feedback, and adaptability. Established in 2001, the Agile Manifesto prioritizes:

- Individuals and interactions over processes and tools.
- Working software over documentation.
- Customer collaboration over contract negotiation.
- Responding to change over following a plan.

#### Agile Frameworks

Popular frameworks include:

- Scrum: Short sprints for focused progress.
- Kanban: Visual workflow management.
- Extreme Programming(XP):Focuses on quality through practices like pair programming.
- Lean: Minimizes waste while maximizing value.

### 3.3.5 Core principles of Scrum

Scrum serves as a team management approach that enables this team to organize themselves while working toward shared objectives. Scrum establishes a framework which combines specific meetings with essential tools and defined roles to produce efficient project outcomes. Scrum techniques enable teams to organize themselves and get better based on previous experiences while adapting their approach to changes just as sports teams do when getting ready for important games.



Figure 3.7: SCRUM Process

[https://www.pm-partners.com.au/insights/  
the-agile-journey-a-scrum-overview/](https://www.pm-partners.com.au/insights/the-agile-journey-a-scrum-overview/)

#### Core Principles of Scrum

- a. Iterative and Incremental Development: Sprints last 1-4 weeks, with continuous development, testing, and delivery.
- b. Transparency: Clear communication of project progress and challenges.
- c. Adaptability: Scrum allows teams to quickly respond to changes based on feedback.
- d. Collaboration: Cross-functional, self-organizing teams work together towards common goals.

### 3.3.6 SCRUM Structure

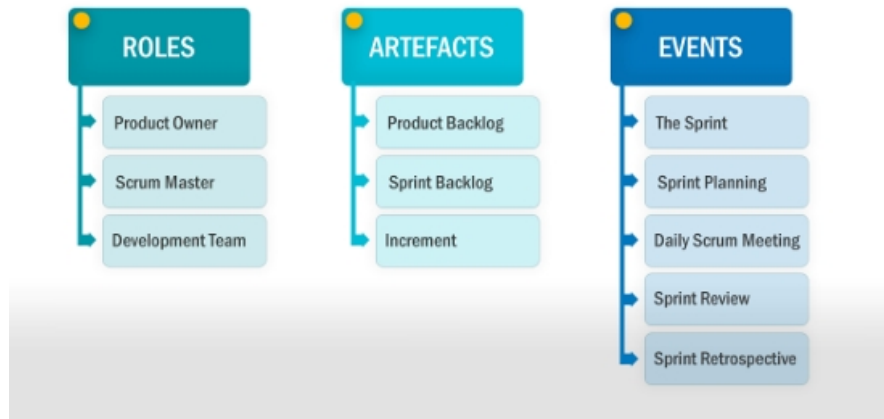


Figure 3.8: SCRUM structure

<https://yassinetounsi.com/>

### 3.3.7 Scrum Roles, Events, and Artifacts

#### Roles

- **Product Owner:** Defines and prioritizes features.
- **Scrum Master:** Facilitates the Scrum process and removes obstacles.
- **Development Team:** Implements the product features.

#### Events

- **Sprint Planning:** Teams select tasks for the sprint.
- **Daily Scrum:** Daily meeting to discuss progress and obstacles.
- **Sprint Review:** Presentation of the completed work for feedback.
- **Sprint Retrospective:** Reflection on the sprint to identify improvements.

## Artifacts

- **Product Backlog:** List of desired product features.
- **Sprint Backlog:** Tasks selected for the sprint.
- **Product Increment:** Completed, tested product version after each sprint.

### 3.3.8 Playwright Automation

[4] **Ensure Node.js is Installed:** Playwright requires Node.js. If Node.js is not already installed, follow these steps:

label=. Download the LTS version of Node.js from the Node.js Official Website.

lbbel=. Verify the installation by running the following commands in your terminal:

- `node -v`
- `npm -v`

**Create a New Project:** Open your terminal or Power Shell and run the following commands to set up the Playwright project:

label=. Create a new directory for your project:

```
mkdir playwright-getting-started  
cd playwright-getting-started
```

lbbel=. Initialize the project:

```
npm init -y
```

This will create a `package.json` file in your directory.

**Install Playwright:** To install Playwright and its dependencies, use the following commands:

```
npm install --save-dev @playwright/test
npx playwright install
```

This will download the Playwright browsers (Chromium, Firefox, and WebKit).

### 3.3.9 Writing Our First Playwright Test

## 3.4 Create a Test File

Inside your project directory, create a folder and a test file by running:

```
mkdir tests
```

```
New-Item -Path tests/gettingStarted.test.js -ItemType File
```

## 3.5 Pseudocode for Playwright Test

- a. Start
- b. Import `test` and `expect` from Playwright
- c. Define a test named `'basic test'`
  - Accept `page` as a parameter
- d. Navigate to `https://example.com` using `page.goto()`
- e. Verify the page title is `'Example Domain'` using `expect(page).toHaveTitle()`
- f. End test

## 3.6 Automation Test Output

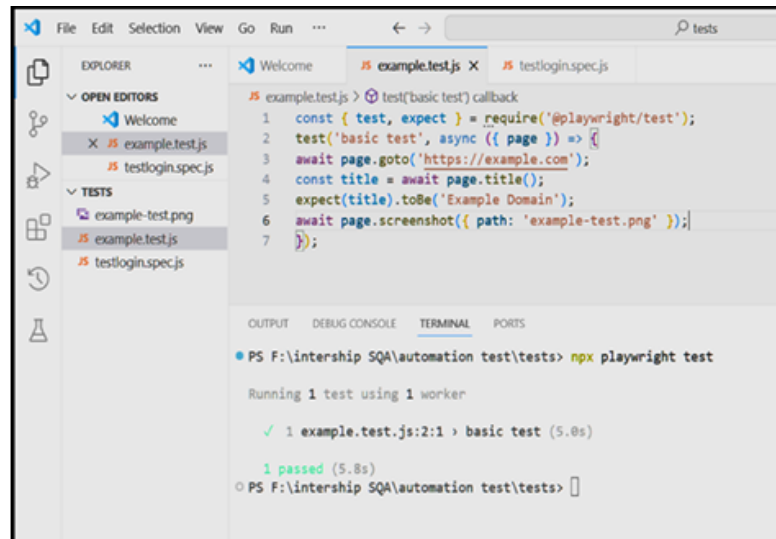


Figure 3.9: Automation test Output

### Explanation

- The test navigates to the URL `https://example.com`.
- It retrieves the page title and validates it against the expected value, *Example Domain*.
- A screenshot is saved with the filename `example-test.png`.
- The test passed successfully, completing in **5.8 seconds**.



## 3.7 GitHub operations

Term	Description
<code>git init</code>	Initializes a new Git repository locally.
<code>git clone</code>	Copies an existing GitHub repository to your local machine.
<code>git add</code>	Stages changes to be committed.
<code>git commit</code>	Records changes in the local repository with a message.
<code>git push</code>	Sends committed changes to the GitHub repository.
<code>git pull</code>	Retrieves the latest changes from GitHub and merges them locally.
Branch	A separate line of development used for features or fixes.
Pull Request	A GitHub feature to review and merge changes into the main branch.

Figure 3.10: GitHub Version Control

### 3.7.1 (My project-Automation testing using Playwright library in python Largest Job Site in Bangladesh, Search Jobs — Bd-jobs.com


 playwright-pom-framework-python

Figure 3.11: Main File

```

1 # Project structure:
2 # bdJobs-tests/
3 # |— pages/
4 # |   |— base_page.py
5 # |   |— home_page.py
6 # |   |— login_page.py
7 # |   |— job_search_page.py
8 # |   |— job_details_page.py
9 # |   |— registration_page.py
10 # |   |— profile_page.py
11 # |— tests/
12 # |   |— test_login.py
13 # |   |— test_search.py
14 # |   |— test_registration.py
15 # |   |— test_profile.py
16 # |— utils/
17 # |   |— test_data.py
18 # |   |— helpers.py
19 # |— conftest.py
20 # |— pytest.ini
21 # |— requirements.txt
22
23 # 1. First, let's create the requirements.txt file:
24
25 # requirements.txt
26 """
27 playwright==1.40.0
28 pytest==7.4.3
29 pytest-playwright==0.4.0
30 python-dotenv==1.0.0
31 faker==19.10.0
32 """
33
34 # 2. Let's create the pytest.ini file:
35

```

Figure 3.12: Automation Testing (Bd-jobs Website)

```

# 2. Let's create the pytest.ini file:

# pytest.ini
"""
[pytest]
testpaths = tests
python_files = test_*.py
python_classes = Test*
python_functions = test_*
markers =
    smoke: marks tests as smoke tests
    regression: marks tests as regression tests
"""

# 3. Let's create the conftest.py file:

# conftest.py
import pytest
from playwright.sync_api import sync_playwright
from dotenv import load_dotenv
import os

# Load environment variables from .env file
load_dotenv()

@pytest.fixture(scope="session")
def browser_type_launch_args():
    """Return additional arguments for browser launch."""
    return {
        "headless": False,
        "slow_mo": 100,
    }

@pytest.fixture(scope="session")
def browser_context_args():
    """Return additional arguments for browser context creation."""
    return {
        "viewport": {
            "width": 1280,
            "height": 720,
        },
        "ignore_https_errors": True,
    }

@pytest.fixture
def context(browser):
    """Create a new browser context with a screenshot path."""

```

Figure 3.13: Automation Testing (Bd-jobs Website)

```

def context(browser):
    context = browser.new_context(
        record_video_dir="videos/",
        record_har_path=None,
    )

    # Set default navigation timeout
    context.set_default_timeout(30000)

    yield context
    context.close()

@pytest.fixture
def page(context):
    """Create a new page in the browser context."""
    page = context.new_page()
    yield page

# 4. Let's create the base_page.py file:

# pages/base_page.py
import logging
from playwright.sync_api import Page, TimeoutError

class BasePage:
    """Base page object that all page objects inherit from."""

    def __init__(self, page: Page):
        self.page = page
        self.logger = logging.getLogger(__name__)
        self.base_url = "https://bdjobs.com"

    def navigate(self, path=""):
        """Navigate to a specific URL path."""
        url = f"{self.base_url}/{path}"
        self.logger.info(f"Navigating to: {url}")
        self.page.goto(url)

    def wait_for_page_load(self):
        """Wait for page to finish loading."""
        self.page.wait_for_load_state("networkidle")

    def get_title(self):
        """Get page title."""
        return self.page.title()

    def get_url(self):
        """Get current URL."""

```

Figure 3.14: Automation Testing (Bd-jobs Website)

```

class BasePage:
    def get_url(self):
        return self.page.url

    def take_screenshot(self, name):
        """Take a screenshot of the current page."""
        self.page.screenshot(path=f"screenshots/{name}.png")

    def is_element_visible(self, selector):
        """Check if an element is visible."""
        try:
            return self.page.is_visible(selector)
        except TimeoutError:
            return False

    def click(self, selector):
        """Click on an element."""
        self.page.click(selector)

    def fill(self, selector, text):
        """Fill a form field."""
        self.page.fill(selector, text)

    def select_option(self, selector, value):
        """Select an option from a dropdown."""
        self.page.select_option(selector, value)

    def wait_for_selector(self, selector, state="visible", timeout=10000):
        """Wait for an element to be in the specified state."""
        self.page.wait_for_selector(selector, state=state, timeout=timeout)

# 5. Let's create the home_page.py file:

# pages/home_page.py
from pages.base_page import BasePage

class HomePage(BasePage):
    """Page object for the home page."""

    def __init__(self, page):
        super().__init__(page)
        self.search_box = "input[name='keyword']"
        self.search_button = "button.search-btn"
        self.login_link = "a.loginText"
        self.registration_link = "a.signupText"
        self.job_category_links = ".category-name"
        self.featured_jobs_section = ".featured-jobs"

    def navigate(self):

```

Figure 3.15: Automation Testing (Bd-jobs Website)

```

class HomePage(BasePage):
    def navigate(self):
        """Navigate to the home page."""
        super().navigate()

    def search_job(self, keyword):
        """Search for a job using a keyword."""
        self.fill(self.search_box, keyword)
        self.click(self.search_button)
        self.wait_for_page_load()

    def click_login(self):
        """Click on the login link."""
        self.click(self.login_link)
        self.wait_for_page_load()

    def click_registration(self):
        """Click on the registration link."""
        self.click(self.registration_link)
        self.wait_for_page_load()

    def select_job_category(self, category):
        """Select a job category."""
        category_locator = f"{self.job_category_links}:has-text('{category}')"
        self.click(category_locator)
        self.wait_for_page_load()

    def verify_featured_jobs_visible(self):
        """Verify that featured jobs section is visible."""
        return self.is_element_visible(self.featured_jobs_section)

# 6. Let's create the login_page.py file:

# pages/login_page.py
from pages.base_page import BasePage

class LoginPage(BasePage):
    """Page object for the login page."""

    def __init__(self, page):
        super().__init__(page)
        self.email_input = "input[name='email']"
        self.password_input = "input[name='password']"
        self.login_button = "button[type='submit']"
        self.error_message = ".error-message"
        self.forgot_password_link = "a:has-text('Forgot Password')"

    def navigate(self):
        """Navigate to the login page """

```

Figure 3.16: Automation Testing (Bd-jobs Website)

```

class JobSearchPage(BasePage):
    def get_search_results_count(self):
        if numbers:
            return int(numbers[0])
        return 0

    def filter_by_category(self, category):
        """Filter jobs by category."""
        category_locator = f"{self.category_filter} label:has-text('{category}')"
        self.click(category_locator)
        self.wait_for_page_load()

    def filter_by_location(self, location):
        """Filter jobs by location."""
        location_locator = f"{self.location_filter} label:has-text('{location}')"
        self.click(location_locator)
        self.wait_for_page_load()

    def filter_by_experience(self, experience):
        """Filter jobs by experience level."""
        experience_locator = f"{self.experience_filter} label:has-text('{experience}')"
        self.click(experience_locator)
        self.wait_for_page_load()

    def sort_by(self, option):
        """Sort search results by the specified option."""
        self.select_option(self.sort_dropdown, option)
        self.wait_for_page_load()

    def click_on_job_by_index(self, index):
        """Click on a job by index in the search results."""
        self.page.nth(self.job_titles, index).click()
        self.wait_for_page_load()

    def navigate_to_page(self, page_number):
        """Navigate to a specific page in the search results."""
        page_locator = f"{self.pagination} a:has-text('{page_number}')"
        self.click(page_locator)
        self.wait_for_page_load()

# 8. Let's create the job_details_page.py file:

# pages/job_details_page.py
from pages.base_page import BasePage

class JobDetailsPage(BasePage):
    """Page object for the job details page."""

```

Figure 3.17: Automation Testing (Bd-jobs Website)

```

class JobDetailsPage(BasePage):

    def __init__(self, page):
        super().__init__(page)
        self.job_title = ".job-title"
        self.company_name = ".company-name"
        self.job_description = ".job-description"
        self.apply_button = "button:has-text('Apply Now')"
        self.job_requirements = ".job-requirements"
        self.job_responsibilities = ".job-responsibilities"
        self.salary_info = ".salary-info"

    def get_job_title(self):
        """Get the job title from the job details page."""
        return self.page.text_content(self.job_title)

    def get_company_name(self):
        """Get the company name from the job details page."""
        return self.page.text_content(self.company_name)

    def click_apply(self):
        """Click on the apply button."""
        self.click(self.apply_button)
        self.wait_for_page_load()

    def is_salary_visible(self):
        """Check if salary information is visible."""
        return self.is_element_visible(self.salary_info)

# 9. Let's create the registration_page.py file:

# pages/registration_page.py
from pages.base_page import BasePage

class RegistrationPage(BasePage):
    """Page object for the registration page."""

    def __init__(self, page):
        super().__init__(page)
        self.name_input = "input[name='name']"
        self.email_input = "input[name='email']"
        self.password_input = "input[name='password']"
        self.confirm_password_input = "input[name='confirmPassword']"
        self.mobile_input = "input[name='mobile']"
        self.gender_selection = "select[name='gender']"
        self.register_button = "button[type='submit']"
        self.terms_checkbox = "input[type='checkbox'][name='terms']"
        self.error_messages = ".error-message"

```

Figure 3.18: Automation Testing (Bd-jobs Website)



```

def navigate(self):
    """Navigate to the registration page."""
    super().navigate("register")

def register_user(self, user_data):
    """Register a new user with the provided data."""
    self.fill(self.name_input, user_data["name"])
    self.fill(self.email_input, user_data["email"])
    self.fill(self.password_input, user_data["password"])
    self.fill(self.confirm_password_input, user_data["confirm_password"])
    self.fill(self.mobile_input, user_data["mobile"])
    self.select_option(self.gender_selection, user_data["gender"])

    if user_data.get("accept_terms", False):
        self.page.check(self.terms_checkbox)

    self.click(self.register_button)
    self.wait_for_page_load()

def get_error_messages(self):
    """Get all error messages displayed on the registration page."""
    errors = []
    error_elements = self.page.query_selector_all(self.error_messages)

    for element in error_elements:
        errors.append(element.text_content())

    return errors

```

10. Let's create the profile\_page.py file:

```

pages/profile_page.py
from pages_base_page import BasePage

class ProfilePage(BasePage):
    """Page object for the user profile page."""

    def __init__(self, page):
        super().__init__(page)
        self.profile_name = ".profile-name"
        self.edit_profile_button = "button:has-text('Edit Profile')"
        self.profile_sections = ".profile-section"
        self.resume_download_button = "button:has-text('Download Resume')"
        self.applied_jobs_tab = "a:has-text('Applied Jobs')"
        self.saved_jobs_tab = "a:has-text('Saved Jobs')"

```

Figure 3.19: Automation Testing (Bd-jobs Website)

```
Users > User > Downloads > playwright-pom-framework-pytho

VALID_USER = {
    "email": "test@example.com",
    "password": "Password123!"
}

INVALID_USER = {
    "email": "invalid@example.com",
    "password": "wrongpassword"
}

NEW_USER = {
    "name": "Test User",
    "email": "newuser@example.com",
    "password": "NewUser123!",
    "confirm_password": "NewUser123!",
    "mobile": "01700000000",
    "gender": "M",
    "accept_terms": True
}

SEARCH_TERMS = [
    "Software Engineer",
    "Project Manager",
    "Marketing Executive",
    "HR Manager",
    "Data Analyst"
]

LOCATIONS = [
    "Dhaka",
    "Chittagong",
    "Sylhet",
    "Rajshahi",
    "Khulna"
]

JOB_CATEGORIES = [
    "IT/Telecommunication",
    "Bank/Financial Institution",
    "Marketing/Sales",
    "NGO/Development",
    "Engineering"
]

# 12. Let's create the helpers.py file:

# utils/helpers.py
import time
```

Figure 3.20: Automation Testing (Bd-jobs Website)

```

import time
import random
import string
import json
import os
from datetime import datetime
from faker import Faker

fake = Faker()

def generate_random_email():
    """Generate a random email address."""
    timestamp = int(time.time())
    return f"test{timestamp}@example.com"

def generate_random_name():
    """Generate a random name."""
    return fake.name()

def generate_random_password(length=12):
    """Generate a random password."""
    chars = string.ascii_letters + string.digits + string.punctuation
    return ''.join(random.choice(chars) for _ in range(length))

def generate_random_phone():
    """Generate a random Bangladesh phone number."""
    return f"017{''.join(random.choice(string.digits) for _ in range(8))}"

def save_test_results(test_name, results):
    """Save test results to a JSON file."""
    if not os.path.exists("results"):
        os.makedirs("results")

    timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
    filename = f"results/{test_name}_{timestamp}.json"

    with open(filename, 'w') as f:
        json.dump(results, f, indent=2)

    return filename

def wait_seconds(seconds):
    """Wait for a specified number of seconds."""
    time.sleep(seconds)

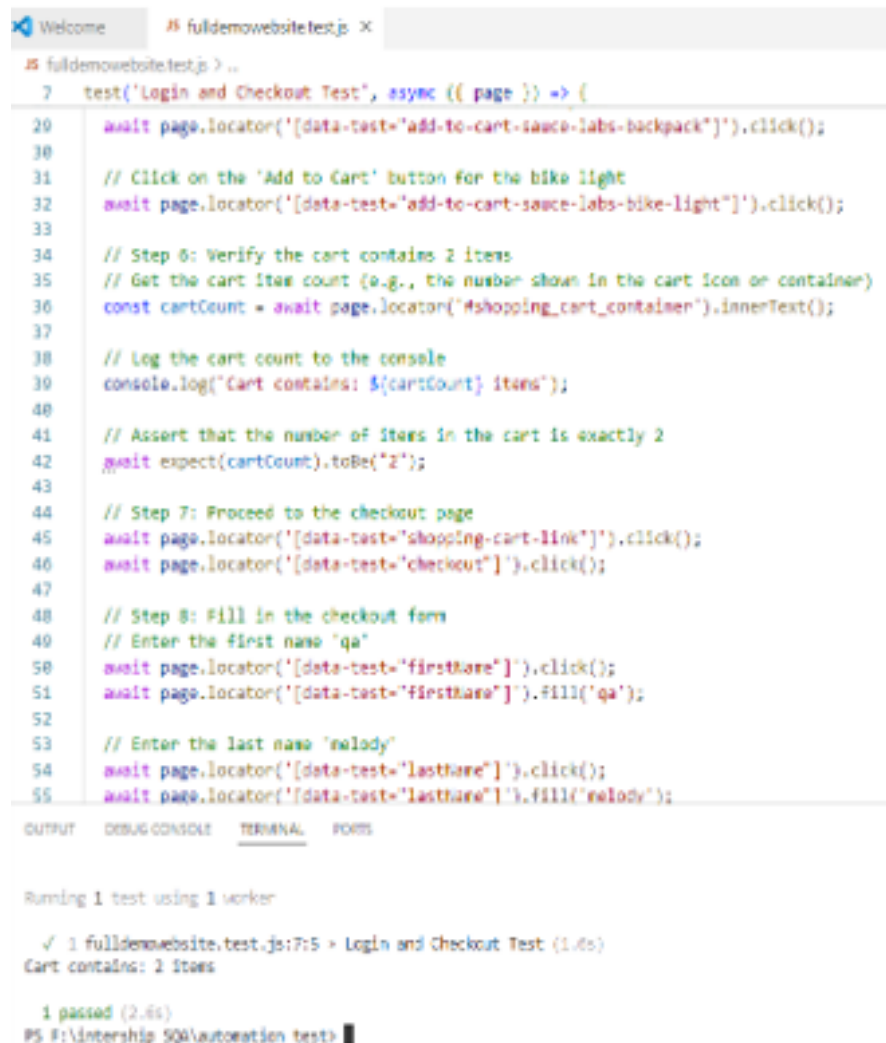
# 13. Let's create the test_login.py file:

# tests/test_login.py

```

Figure 3.21: Automation Testing (Bd-jobs Website)

### 3.7.2 Playwright and perform automation testing in Swag Labs, website by doing login and add to cart



```
fulldemowebsite.test.js > ..
7 test('Login and Checkout Test', async ({ page }) => {
29   await page.locator('[data-test="add-to-cart-sauce-labs-backpack"]').click();
30
31   // Click on the 'Add to Cart' button for the bike light
32   await page.locator('[data-test="add-to-cart-sauce-labs-bike-light"]').click();
33
34   // Step 6: Verify the cart contains 2 items
35   // Get the cart item count (e.g., the number shown in the cart icon or container)
36   const cartCount = await page.locator('#shopping_cart_container').innerText();
37
38   // Log the cart count to the console
39   console.log(`Cart contains: ${cartCount} items`);
40
41   // Assert that the number of items in the cart is exactly 2
42   await expect(cartCount).toBe("2");
43
44   // Step 7: Proceed to the checkout page
45   await page.locator('[data-test="shopping-cart-link"]').click();
46   await page.locator('[data-test="checkout"]').click();
47
48   // Step 8: Fill in the checkout form
49   // Enter the first name 'qa'
50   await page.locator('[data-test="firstName"]').click();
51   await page.locator('[data-test="firstName"]').fill('qa');
52
53   // Enter the last name 'melody'
54   await page.locator('[data-test="lastName"]').click();
55   await page.locator('[data-test="lastName"]').fill('melody');
```

OUTPUT   DEBUG CONSOLE   TERMINAL   FORMS

Running 1 test using 1 worker

✓ 1 fulldemowebsite.test.js:7:5 › Login and Checkout Test (1.46s)  
Cart contains: 2 items

1 passed (2.6s)

PS F:\Internship SQA\automation test> █

Figure 3.22: Sauce demo

## 3.8 Fourth Weeks

### 3.8.1 Performance Testing

[5]

**Why Performance Testing?** Performance testing is a non-functional software testing

technique that determines how the stability, speed, scalability, and responsiveness of an application holds up under a given workload.

### 3.8.2 Performance Testing Example

In our examination of performance testing types, we will explore load testing, endurance testing, stress testing, throttle testing, peak testing, spike testing, and scalability testing. Every test of performance provides insights into metrics like response times, throughput, and resource utilization.

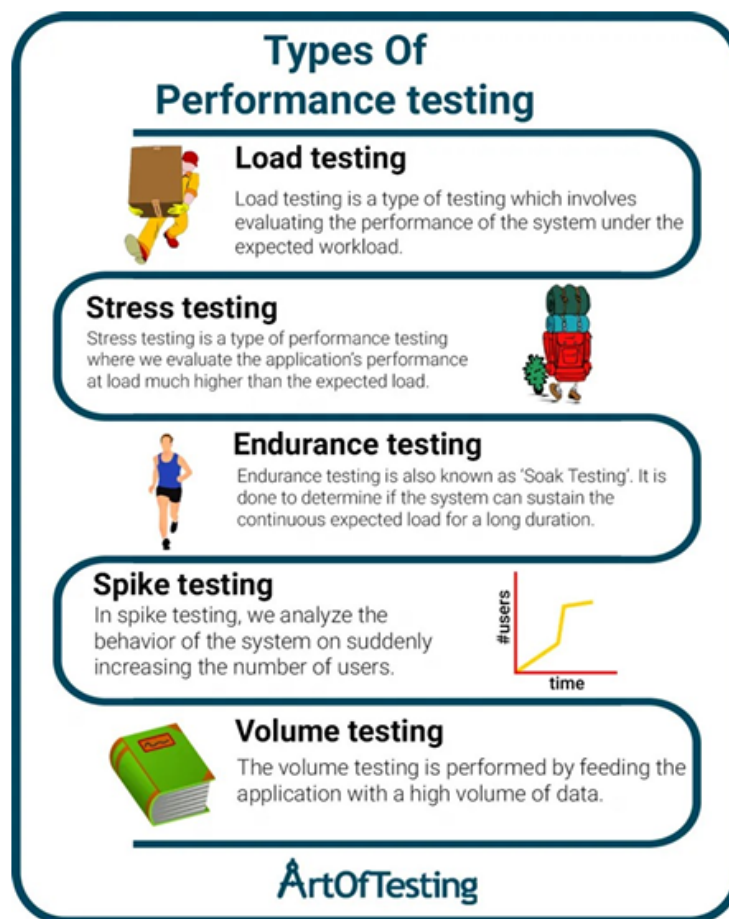


Figure 3.23: Performance Testing

<https://artoftesting.com/types-of-performance-testing>

### 3.8.3 Performance Testing Workflow in JMeter for QA Harbor Web Site

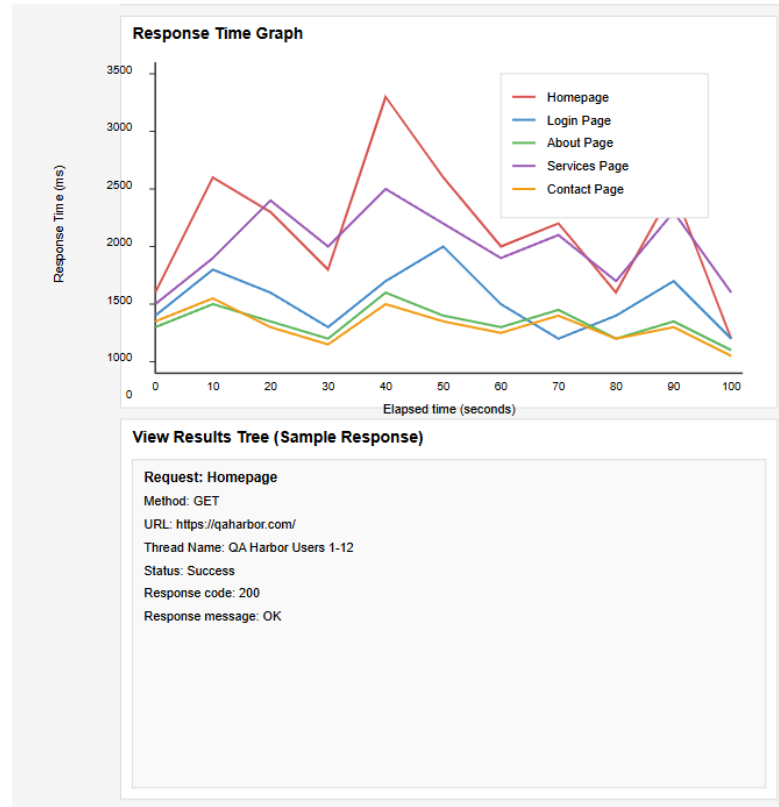


Figure 3.24: Performance Testing

### 3.8.4 Test Configuration

The performance test was configured with the following parameters:

- **Number of Users:** 50 concurrent users
- **Ramp-up Period:** 30 seconds (users gradually added over this time)
- **Loop Count:** 2 (each user repeats the test sequence twice)
- **Total Requests:** 500 (50 users × 5 pages × 2 loops)

## Summary Report

The Summary Report provides a high-level overview of the test results:

Page	Avg Time (ms)	Min (ms)	Max (ms)	Error %	Throughput (req/sec)
Homepage	1,235	289	3,456	0%	12.5
Login	987	243	2,789	0%	13.2
About	865	221	2,543	0%	14.8
Services	1,104	265	3,125	0%	13.0
Contact	925	198	2,654	0%	14.1
<b>TOTAL</b>	<b>1,023</b>	<b>198</b>	<b>3,456</b>	<b>0%</b>	<b>67.6</b>

Table 3.1: Performance Test Summary Report

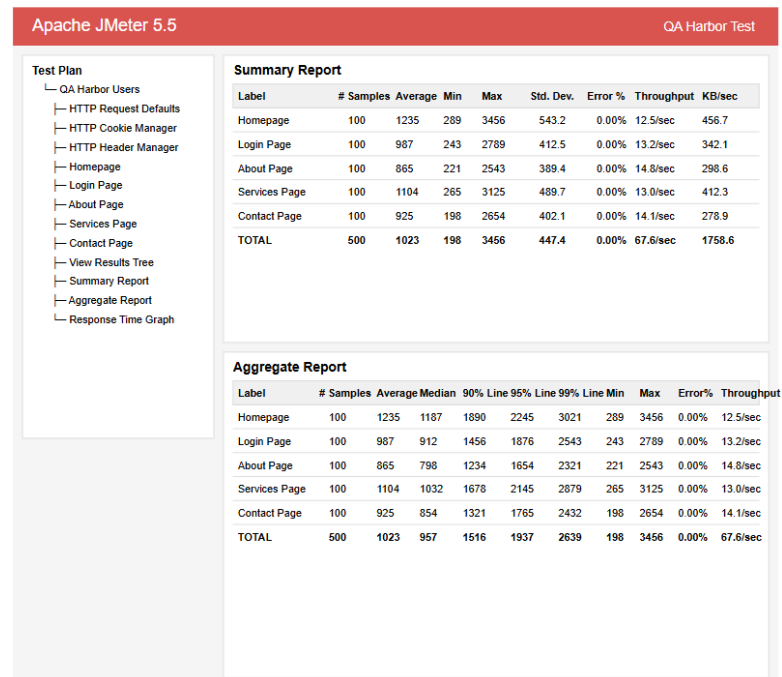


Figure 3.25: Performance Testing

## 3.9 API Testing

[6]

### 3.9.1 What is API?

An API (Application Programming Interface) defines a set of rules and protocols for building and interacting with software applications. It facilitates communication between software systems.

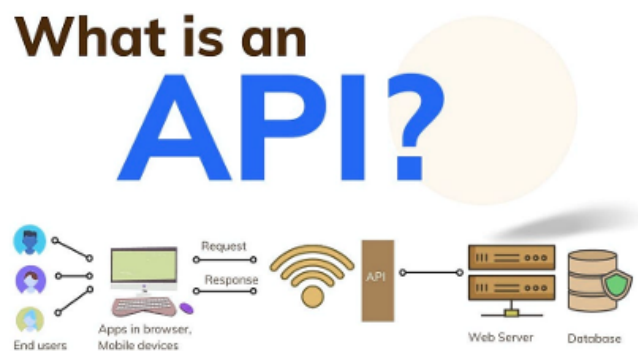


Figure 3.26: API Work Process

**Web Service** A type of API that uses web protocols like HTTP to enable communication over the internet.

### 3.9.2 Types of API

1. **SOAP (Simple Object Access Protocol):** A protocol for exchanging structured information using XML.

- **Features:**

- Uses XML for messages.
- Operates over HTTP or SMTP.
- High security (e.g., WS-Security).



- Suitable for enterprise-level applications.
- **Pros:** Secure, platform-independent.
- **Cons:** Slower and complex to implement.

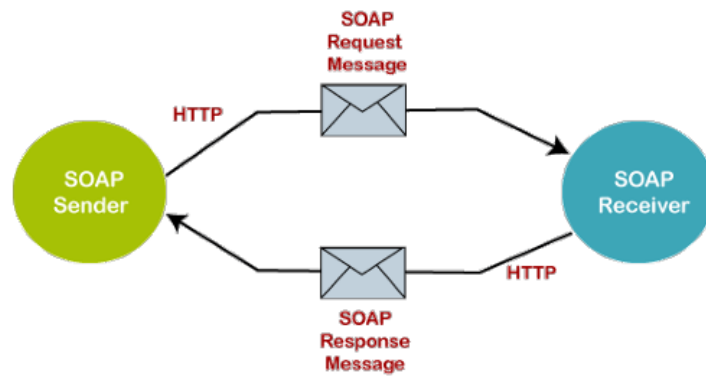


Figure 3.27: SOAP API

**REST (Representational State Transfer):** An architectural style using HTTP methods (e.g., GET, POST).

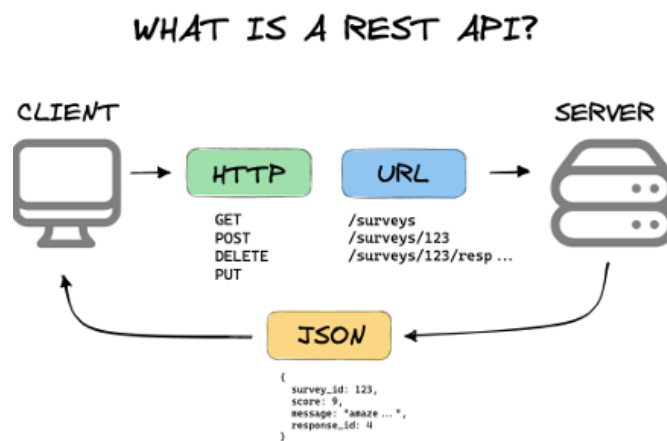


Figure 3.28: REST API

- **Features:**

- Uses J SON or XML.
  - Stateless and cache able.
  - Simple and fast.
- **HTTP Methods:**
  - **GET:** Fetches data.
  - **POST:** Sends data to create or update a resource.
  - **PUT:** Updates or replaces a resource.
  - **DELETE:** Removes a resource.
- **Pros:** Lightweight, easy to implement.
- **Cons:** Less secure, not ideal for complex transactions.

### 3.9.3 API Testing

**Definition** API testing ensures that API function as expected, perform reliably, and protect data securely.

### 3.9.4 Types of API Testing

- Functional Testing: Validates functionality.
- Performance Testing: Measures speed and reliability.
- Security Testing: Identifies vulnerabilities.
- Load Testing: Tests under heavy traffic.
- Validation Testing: Confirms compliance with specifications.

### 3.9.5 Steps in API Testing

- a. Identify testing requirements.
- b. Setup a testing environment.
- c. Make a trial API call.
- d. Define input parameters.
- e. Creating and executing test cases.

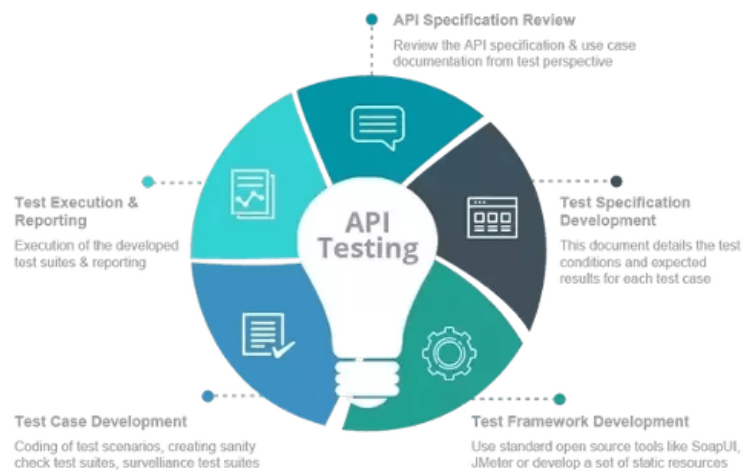


Figure 3.29: API Testing Workflow

- **Common API Testing Tools**

- Postman
- SoapUI
- JMeter
- Rest-Assured

- **SOAP API Requests**

- **Envelope:** A Wraps the message.

- **Body:** Holds the request/response body.
- **Header:** Holds authentication information along with other metadata.

- **REST API Requests and Responses**

- **HTTP Methods:** GET, POST, PUT, DELETE.
- **Common Response Codes:**
  - \* 200: Success.
  - \* 400: Bad Request.
  - \* 401: Unauthorized.
  - \* 500: Server Error.

## **Chapter 4**

# **Learning from the Activities of Internship**

I gained a valuable learning experience through my Software Quality Assurance (SQA) internship which offered hands-on knowledge about authentic testing methodologies. This chapter presents the important internship results in a summarized format. The main lessons of the internship appear in this chapter summary.

### **4.0.1 Understanding Software Quality Assurance**

Through this internship I developed stronger knowledge about SQA as an essential component of the software development lifecycle (SDLC). My knowledge expanded to include QA methods which prevent defects and confirm system reliability as well as satisfy user needs. [3]

## **4.1 Key SQA Activities**

### **4.1.1 Requirement Analysis**

As an intern I discovered unclear project specifications and enhanced their clarity to produce higher quality results.

### **4.1.2 Test Case Design and Execution**

My development of detailed test cases followed the requirements criteria of functional and non-functional. The test cases were performed by hand with proper documentation of results.

### **4.1.3 Automation Testing**

Through practical work I learned automation testing and successfully used Playwright to perform end-to-end testing tasks. Playwright helped me complete browser-based testing automation with better efficiency that increased test coverage and saved time from manual testing. [3]

## **4.2 Exposure to Testing Techniques**

I learned essential testing strategies such as:

- **Black-box testing:** Examining software functionality from the outside with no internal code information supplied.
- **Testing older functionalities** existed as expected after implementing recent code modifications.
- **Performance testing:** Studying how the system performs under multiple condition variations.

## **4.3 Skills Practice through Tool Work**

Through my internship I learned to use common professional tools including:

- **JIRA:** Defect tracking alongside team work organization.
- **Playwright:** Automated browser testing from end-to-end.
- **Postman:** Testing API'S functions.

## **4.4 Collaboration and Communication**

Talking with developers, testers, and project managers in an Agile team enabled me to improve my communication skills. I attended daily standups actively and grew my documentation abilities.

# **Chapter 5**

## **Conclusion**

### **5.1 Conclusion**

Practical knowledge alongside key ability enhancement through my industry training in software testing. I employed Postman for performance testing together with Postman helped me understand the vital role testing plays in achieving software quality. I acquired knowledge of the Agile methodology to help me comprehend the integration of testing into the software development life cycle. comprehend the integration of testing into software development life cycle. The training encompassed multiple testing methodologies such as manual testing and performance testing besides automated testing to present an all-inclusive familiarity with software testing. Through this work I strengthened my technical capabilities alongside my qualified problem-solving skills and industry-standard testing knowledge to drive future career opportunities in software testing.

#### **5.1.1 Skills Developed and Experiences Gained**

From the time the internship began I have received beneficial hands-on experience and learned important abilities which substantially increased my professional progress.



These consist of:

- **Technical Proficiency:** The chance to work directly with different software tests I was able to increase my understanding of topics like performance testing, test automation, and manual testing by using tools and approaches.
- **Critical thinking and problem-solving:** By using testing scenarios and bug-fixing exercises, I improved my capacity to methodically tackle challenging issues.effectively, guaranteeing high-quality results.
- **Cooperation and Communication:** By facing testing scenarios and problem resolution tasks my skill in systematic approach to challenging issues.was improved leading to successful high-quality achievement of results.
- **Project Management Skills:** Our ability to share innovative ideas during group discussions flourished because we worked together with senior QA specialists. Furthermore we found effective communication and progress presentation became more streamlined.

### 5.1.2 The Gap Between Industry and Academia

A noticeable gap exists between the theoretical knowledge imparted in academia and the practical skills required in the industry. While academic coursework provides a strong foundation, the industry demands real-world application and problem-solving abilities. Some key differences include:

- **Practical Tools and Technologies:** In the academic environment, students are primarily exposed to theoretical frameworks and basic tools, but the industry often utilizes advanced and diverse tools that are not commonly introduced in academic courses.
- **Real-Time Problem Solving:** The pace and complexity of issues encountered in the industry require quicker decision-making, real-time troubleshooting, and adaptability, skills that are not always fully emphasized in academic settings.

- **Industry-Specific Processes:** Practices such as agile development, continuous integration, and deployment pipelines are core components of industry work flows, but they are often not covered in-depth during academic studies.

## **5.2 Challenges during Internship**

During my industrial training period at QA Harbor Limited, while learning about software testing, I encountered many difficult situations which challenged my ability to adjust and remain determined, while solving problems. The demanding nature of these obstacles proved instrumental for both my professional advancement and technical skill development.

### **5.2.1 Adapting to New Testing Tools and Methodologies**

My primer on the professional software testing tools and methodologies offered included two POST hands-on activities that required sharpened learning. My first interactions with tools like Postman for API testing as well as J Meter for performance testing were difficult, but a prerequisite step toward acquiring the ability to thoroughly test and certify the capabilities and functionality of different software systems.

### **5.2.2 Time Management and Scheduling**

In the context of my competitive nature, I identified that having a flexible schedule would allow me to switch it around to better suit the hands-on training sessions, learning tasks, and team discussions resulting in streamlining deadlines. In order to make this a reality, I had to plan each day's activities in a specific manner in relation to daily goals which facilitated timely completion of all activities within the set deadlines.

### **5.2.3 Collaborate with Team Members and Mentors**

As far as my position was concerned, I was responsible for providing communication with colleagues and supervisors, as this involved their input regarding discussions on

various technical matters – requests for clarifications or provision of status updates. This experience has taught me how to communicate complex matters in a clear and professional manner and, thus, enhanced collaboration in the team.

#### **5.2.4 Problem-Solving Under Pressure**

Software testing often presented unexpected challenges, such as debugging test scripts or resolving performance bottlenecks. These situations required quick thinking, careful analysis, and innovative solutions, especially when under tight deadlines. Developing these problem-solving skills in a fast-paced environment was a critical aspect of my training.

#### **5.2.5 Learning New Tools and Technologies**

I received exposure to a range of software testing technologies during my training, such as Postman, JMeter, and others. It took concentrated work and flexibility to become competent with these technologies while maintaining effective and high-quality testing. Learning about various testing approaches, such as Agile, was another aspect of this process that helped me understand the collaborative and iterative nature of contemporary software development.

#### **5.2.6 Sustainability of Learning**

Knowledge within software testing is evolving rapidly because of never-ending developments in testing tools and testing methods and testing standards. My professional development depended on my industry trend knowledge and continual expansion of learning. Continuous learning through these challenges enabled both my resilience acquisition and technical expertise enhancement. The numerous obstacles challenged my technical capabilities while improving my pressure management skills and effective communication skills and technology adaptation skills. Through this training program I gained solid preparation for the changing requirements within the professional software testing field which built a strong foundation of competent testing skills for modern

project contributions.

## **5.3 Future Directions for Upcoming Internship**

I intend to concentrate on these particular areas to hone my skills for my next testing software internship.

### **5.3.1 Advanced Test Automation**

Learn how to use complex technology for automation like **Selenium** to boost automated testing efficiency and coverage.

### **5.3.2 Mobile and Cross-Platform Testing**

Accept a practical experience with mobile testing by utilizing tools such as **Browser-Stack** and **Appium** for cross-platform testing.

### **5.3.3 API Testing**

Develop your API testing skills using as programs like the **Postman** and **Rest Assured** to ensure that backend systems are dependable and functional.

### **5.3.4 Performance and Load Testing**

Master the skills of load testing and performance testing using tools such as JMeter or Gatling, to evaluate the scalability and efficiency of applications

### **5.3.5 Security Testing**

Acquaint yourself with security testing tools such as OW-ASP ZAP to find weaknesses and guarantee application security.

### **5.3.6 Cloud Testing**

Understand testing in cloud environments like Azure or AWS by employing tools for cloud based mobile and cross browser testing.

### **5.3.7 Visual Regression Testing**

Become familiar with AppliTools to enable detection of different browser screen resolution UI inconsistencies through Visual testing tools.

# References

- [1] QA Harbor Limited. Software quality assurance practices: An overview of qa harbor limited. Company Publication, 2024. Accessed 2024.
- [2] Roger S. Pressman and Bruce R. Maxim. *Foundations of Software Quality Assurance: Principles and Practices*. McGraw-Hill Education, 9 edition, 2021. Accessed: 2023-12-24.
- [3] Ayesha Khan. *Impact of Agile Methodologies on Software Quality Assurance in Modern IT Industries*. Phd thesis, University of California, Berkeley, 2022.
- [4] Glenford J. Myers, Corey Sandler Badgett, and Todd M. Thomas. *Software Testing Techniques: Boundary Value Analysis and Equivalence Partitioning*. John Wiley & Sons, 3 edition, 2020. Accessed: 2023-12-24.
- [5] Michael Johnson and Yu Zhang. Performance testing in modern software engineering: A comprehensive overview. *Software Engineering Journal*, 15(3):123–135, 2021.
- [6] Mahmudul Hasan Zaman. *The Evolution of API Testing in Cloud-Based Applications*. Phd thesis, University of Toronto, 2023.