# SOFTWARE TESTING

Is your software ready to face the real world,
or will a single bug cost you millions?

# Definition

Software Testing is the process of evaluating a software application or system to detect errors, bugs, or defects and ensure it meets the specified requirements. It involves validating that the software behaves as expected under various conditions and scenarios.

# Importance

- Ensures software quality and reliability by detecting defects early.
- Reduces the risk of software failure in production.
- Enhances user satisfaction by delivering high-quality software.
- Validates that the product meets customer requirements and expectations.

# WHY IS TESTING IMPORTANT?

- Prevents critical failures and data loss.
- Ensures software reliability, stability, and performance.
- Improves customer satisfaction and trust.
- Reduces maintenance and support costs.
- Identifies and mitigates potential security risks.
- Helps maintain a positive brand reputation.

# OBJECTIVES OF SOFTWARE TESTING

- Detect defects and bugs before software release.
- Improve software quality and reliability.
- Validate functional and non-functional requirements.
- Ensure compliance with technical and business requirements.
- Identify and mitigate risks before deployment.
- Enhance the overall user experience.

# TYPE OF SOFTWARE TESTING

- **Functional Testing:** Focuses on verifying the functional aspects of a software application to ensure it behaves as expected based on the requirements.
- **Non-Functional Testing:** Evaluates the non-functional aspects like performance, security, usability, and reliability of the application.
- **Manual Testing:** Involves human testers manually executing test cases without automation tools.
- **Automation Testing:** Uses scripts and automated tools to execute tests, reducing manual effort.
- **White Box Testing:** Tests the internal structures and logic of the code, focusing on paths, conditions, and loops.
- **Black Box Testing:** Focuses on validating the software's functionality without knowing the internal code structure.
- **Gray Box Testing:** Combines both white and black box testing techniques to assess the internal structure while focusing on the functional behavior.

# SOFTWARE TESTING LIFE CYCLE (STLC)

- **Requirement Analysis:** Understanding what needs to be tested.
- **Test Planning:** Defining the test strategy, scope, and resources.
- **Test Case Development:** Writing test cases and scripts.
- **Test Environment Setup:** Preparing the test environment and data.
- **Test Execution:** Running tests and logging results.
- **Test Reporting:** Documenting issues and test outcomes.
- **Test Closure:** Final evaluation, documentation, and project sign-off.

# Manual Testing

- **Definition:** Manual Testing is the process of manually executing test cases without using any automation tools to identify bugs, defects, or issues in the software. It relies on the skills, intuition, and experience of human testers.
- **Characteristics:**
  - Involves human judgment and intuition.
  - Best for short-term projects and exploratory testing.
  - Detects user experience and usability issues effectively.
  - Time-consuming but flexible in approach.
- **Pros:**
  - No need for advanced technical knowledge or programming skills.
  - Better for identifying user interface and user experience issues.
  - Allows for more creative and exploratory testing.
- **Cons:**
  - Time-intensive and prone to human error.
  - Less consistent and repeatable than automated testing.
  - High ongoing costs for large projects.

# Automation Testing

**Definition:** Automation Testing is the process of using software tools and scripts to execute pre-defined test cases automatically. It is designed to reduce manual effort, increase test coverage, and improve test accuracy.

- **Characteristics:**
  - Requires initial setup and scripting but reduces long-term costs.
  - Consistent, repeatable, and faster execution.
  - Best suited for regression testing and high-volume, repetitive tests.
- **Pros:**
  - Faster execution and turnaround time.
  - Reusable test cases and scripts.
  - Better accuracy and consistency compared to manual testing.
  - Ideal for continuous integration and delivery (CI/CD) pipelines.
- **Cons:**
  - High initial setup cost and maintenance effort.
  - Requires programming skills and technical expertise.
  - Limited in handling complex and exploratory scenarios.

# AUTOMATION TESTING TOOLS

- **Selenium:** Widely used for web application testing.
- **JUnit, TestNG:** Commonly used for Java application testing.
- **Appium:** Mobile application testing.
- **Postman:** API testing and automation.
- **Cucumber:** Behavior-Driven Development (BDD) testing.

# Manual vs. Automation Testing

| Feature | Manual Testing | Automation Testing |
| --- | --- | --- |
| Speed | Slow, time-consuming | Fast, efficient |
| Cost | Low initial cost, high ongoing cost | High initial cost, low long-term cost |
| Accuracy | Prone to human error | High accuracy, repeatable |
| Flexibility | High, creative | Limited to predefined scripts |
| Human Insight | Yes | No |
| Test Coverage | Limited | High, broader scope |

# BEST PRACTICES IN SOFTWARE TESTING

- Write clear, reusable, and maintainable test cases.
- Automate repetitive and high-impact tests.
- Use bug-tracking tools for efficient issue management.
- Regularly update and optimize test scripts.
- Collaborate closely with development teams.

# What Happens If We Skip Testing?

**High Financial Loss:** Costly bugs can lead to major financial damage. For example, the NASA Mars Climate Orbiter failed due to a unit conversion error, resulting in a $327.6 million loss.

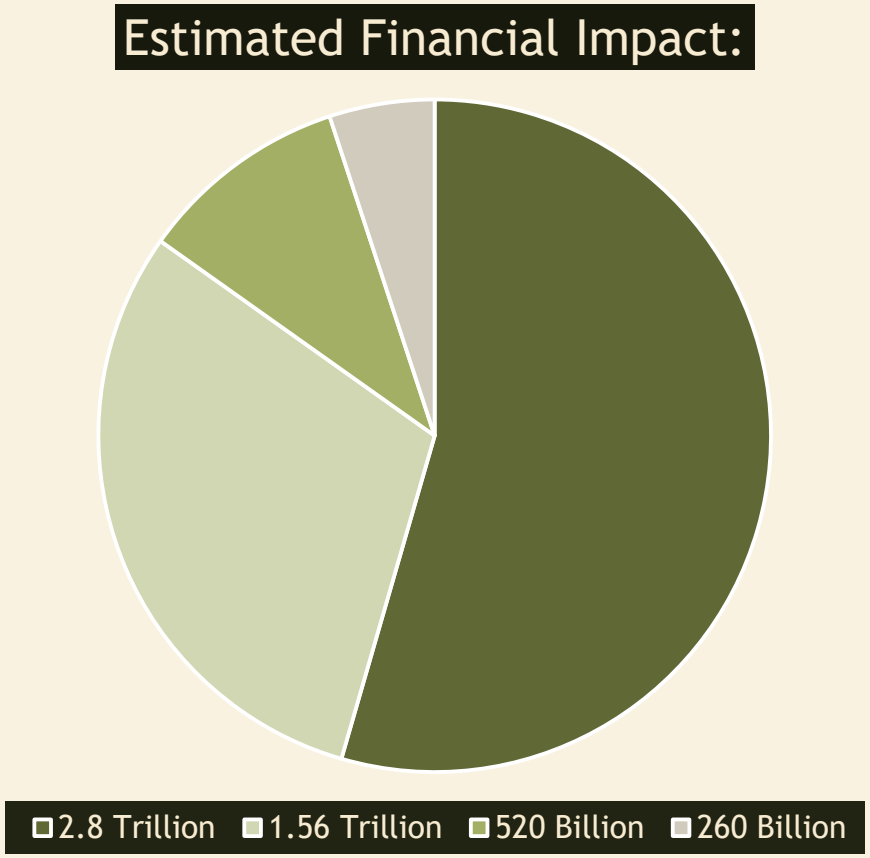
**Poor User Experience:** Unresolved bugs can frustrate users and lead to negative reviews, reducing customer loyalty.

- **Security Vulnerabilities:** Lack of security testing can expose systems to hacking and data breaches, potentially costing millions in fines.
- **Reputational Damage:** Companies like Equifax faced significant reputational damage after major security breaches.
- **Legal Consequences:** Non-compliance with industry regulations can lead to legal actions and fines.

# Impact of Skipping Software Testing

## Cost of Software Failures:

- **2.08 Trillion USD**: Estimated cost of poor software quality in the U.S. in 2020 (IT-CISQ).
- **260 Billion USD**: Cost of failed IT projects.
- **520 Billion USD**: Issues with legacy systems.
- **1.56 Trillion USD**: Operational software failure.



## Cost of Post-Launch Failures:

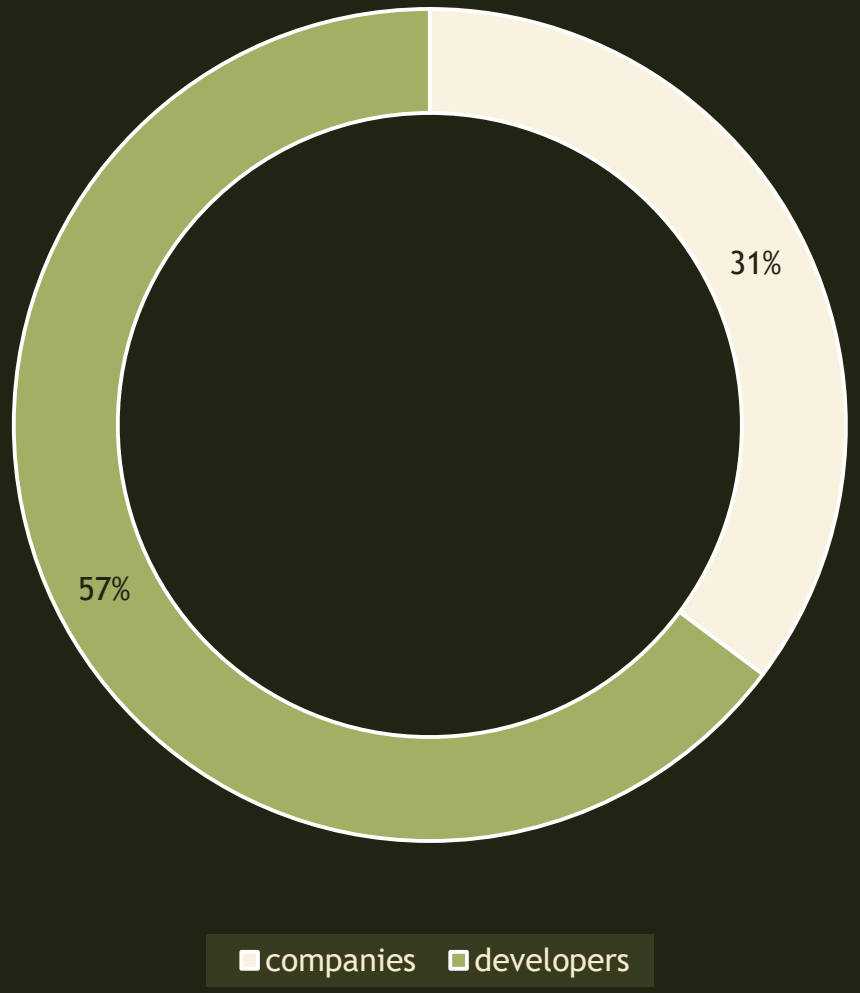
- *$61 Billion USD annually due to software failure.*
- *Developers spend 26% of their time fixing post-release errors (Undo.io).*

## Increasing Cost of Errors Over Time:

- 100x more expensive to fix issues after deployment.
- 50x more expensive during testing phase.
- 10x more expensive during development phase.
- 5x more expensive in the design stage (ShiftAsia).

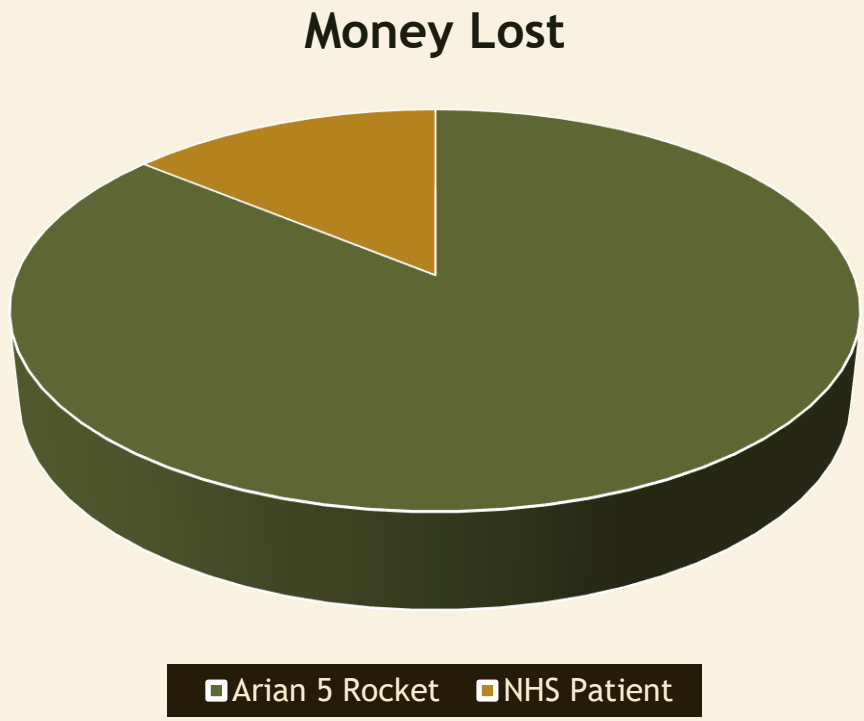
# Impact on Productivity

- 31% of companies spend more time addressing issues post-launch (WiFi Talents).
- 57% of developers spend 5+ hours/week fixing production bugs.



# Real-World Failures Due to Poor Testing:

- **Arian 5 Rocket:** Explosion costing $370 Million USD due to software bug.
- **NHS Patient Scheduling System:** Lost 709,000 messages, costing £6.6 Million GBP (Devlo).



# USER EXPERIENCE IMPACT:

- 42% of bugs come from user interface issues.
- 82% of bugs discovered after release impact user satisfaction (WiFi Talents).
- Prevents critical failures and data loss.
- Ensures software reliability, stability, and performance.
- Improves customer satisfaction and trust.
- Reduces maintenance and support costs.
- Identifies and mitigates potential security risks.
- Helps maintain a positive brand reputation.

# What is Sauce Demo?

- Sauce Demo is a popular demo e-commerce website designed for practicing software testing, test automation, and performance testing.
- It includes a range of features such as login authentication, product listings, shopping cart functionality, checkout processes, and order confirmation, making it ideal for comprehensive test scenarios.

# Project Goal:

- Test the Sauce Demo website using various testing techniques to ensure functionality, usability, performance, and security.
- Identify potential bugs, document issues, and provide improvement recommendations.

# Project Team Contributions and Responsibilities

## Team Members and Their Roles:

**Reem Hossam Ezzeldein (Test Plan and API Testing):**

- Developed the comprehensive test plan for the Sauce Demo project.
- Defined the scope, objectives, and testing approach for the entire project.
- Conducted API testing to validate backend functionality and data integrity.
- Used Postman for creating and executing API requests.
- Validated response status codes, response times, and data accuracy.
- Automated API test cases using Postman collections and Newman for continuous testing.

- **Usama Sdeik(Test Case Design):**
  - *Designed detailed test cases covering key functionalities such as login, product search, cart updates, and checkout processes.*
  - *Used tools like Excel and Google Sheets to organize and track test cases.*
  - *Conducted risk analysis to prioritize critical test cases and scenarios.*

- **Mahmoud zayed (Manual Testing and Bug Reporting):**
  - *Executed manual test cases to validate the functional behavior of the website.*
  - *Focused on exploratory testing to uncover hidden issues.*
  - *Logged defects in a bug tracking tool (e.g., JIRA, Excel), providing detailed descriptions, severity levels, and steps to reproduce.*
  - *Verified UI consistency and responsiveness across different devices and browsers.*

## Jonmana Mohamed (Automation Testing - Selenium):

- Developed automated test scripts using Selenium WebDriver with Java.
- Created reusable page objects and test methods to ensure maintainable test code.
- Implemented data-driven tests for login, cart, and checkout functionalities.
- Used TestNG for test execution and report generation.
- Integrated Selenium with Maven for build automation and Jenkins for continuous integration.

# TEAM MEMBERS:

- **Reem Hossam Ezzeldein:** Test Planning , API Testing and Postman Integration.
- **Usama Hamdy Sdeik:** Test Cases Design.
- **Mahmoud Hamdy Abdelsalam Zayed:** Manual Functional Testing and Bug Reporting.
- **Joumana Mohamed Hassan abozeid:** Automation Testing using Selenium.

# Test Plan Overview:

- **Test Plan Purpose:**

  To define the testing approach, scope, objectives, and deliverables for the Sauce Demo project.

- **Scope of Testing:**

  Functional Testing, Non-Functional Testing, UI Testing, API Testing, and Performance Testing.

- **Testing Levels:**

  Unit Testing, Integration Testing, System Testing, and Acceptance Testing

- **Roles and Responsibilities:**

  **Reem Hossam:** Created the test plan and Performed API testing using Postman and analyzed API responses.

  **Usama Hamdy:** Created test cases based on the project requirements.

  **Mahmoud Zayed:** Conducted manual testing and documented defects.

  **Joumana Abozwid:** Developed and executed automated test scripts in Selenium

# TEST CASES AND EXECUTION

- **Test Case Development:**
  - Test cases were designed to cover all critical paths, including login, product search, cart updates, and checkout processes.
- **Test Execution:**
  - Testers executed manual and automated test cases, logging defects as they were discovered.
- **Sample Test Cases:**
  - Login with valid credentials.
  - Add multiple items to the cart and verify total price.
  - Checkout process with valid and invalid information.

# Bug Reporting and Management

- **Bug Tracking Process:**
  - Bugs were logged and tracked using a bug tracking tool (e.g., JIRA, Excel).
  - Each bug report included a detailed description, severity level, steps to reproduce, expected and actual results, and screenshots.
- **Sample Bug:**
  - **Title:** Product price mismatch in cart.
  - **Severity:** High
  - **Description:** The total price in the cart does not match the sum of individual product prices.
  - **Steps to Reproduce:** Add two items to the cart and proceed to checkout.

## Automation Testing (Selenium)

- **Test Automation Framework:**
  - *Used Selenium WebDriver with Java for automated testing.*
- **Automation Goals:**
  - *Reduce manual effort, improve test coverage, and ensure consistency.*
- **Sample Automated Test:**
  - *Automated login tests with positive and negative scenarios.*
  - *Automated cart functionality and checkout tests.*

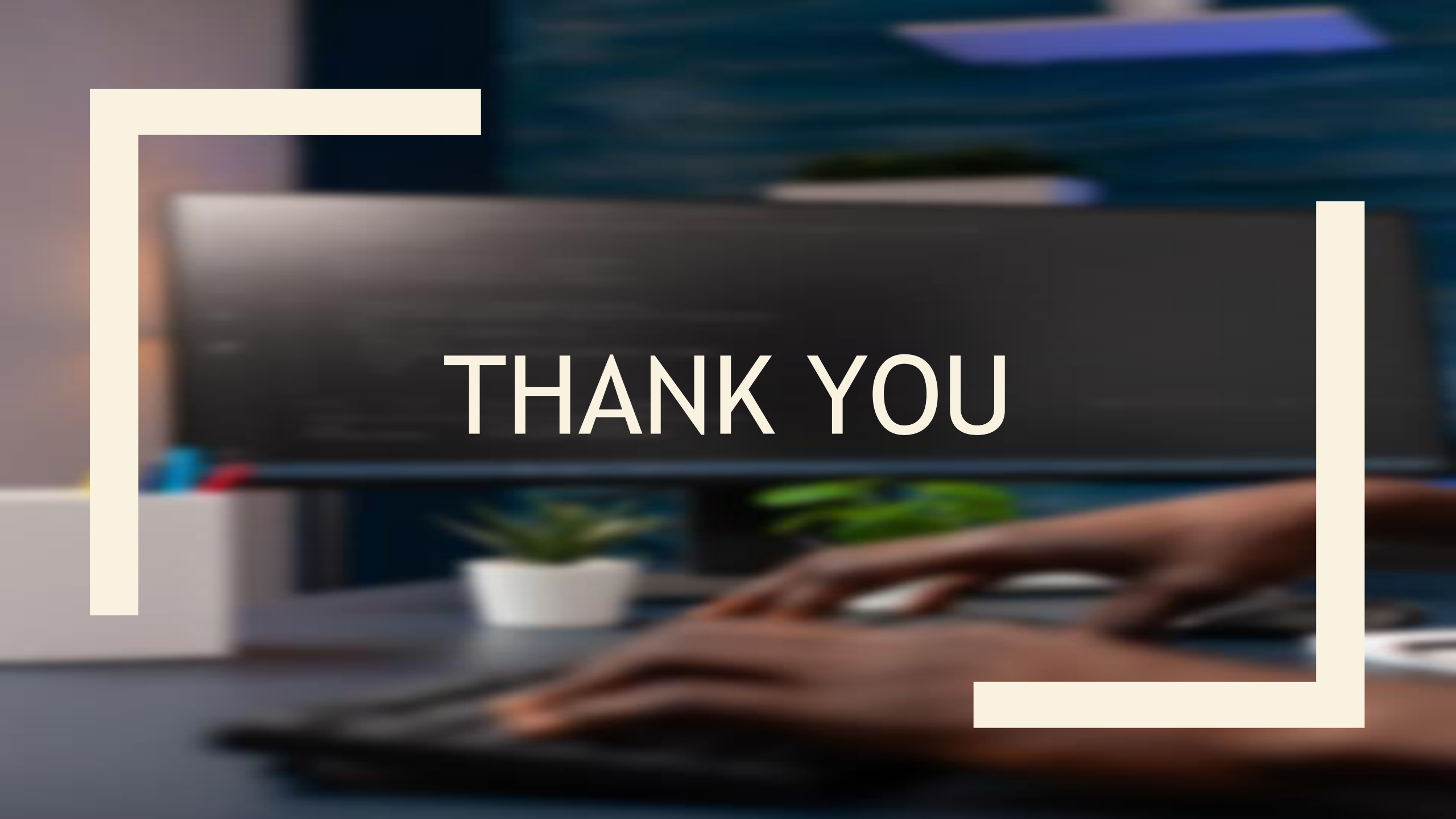
## API Testing (Postman)

- **Why API Testing?**
  - *To verify the reliability and performance of backend services.*
- **Tools Used:** Postman for REST API testing.
- **Sample API Test:**
  - *Validate product listing API for correct responses and response times.*

# Project Outcomes and Key Takeaways

- **Project Outcome:**
  - *Identified multiple critical bugs and performance bottlenecks.*
  - *Improved test coverage through automation and API testing.*
  - *Delivered a comprehensive test report to the development team.*
- **Key Takeaway:**
  - *Thorough testing ensures higher product quality, customer satisfaction, and business success.*

THOROUGH SOFTWARE TESTING IS ESSENTIAL FOR ENSURING THE FUNCTIONALITY, PERFORMANCE, AND SECURITY OF ANY APPLICATION. THROUGH OUR TESTING OF THE SAUCE DEMO WEBSITE, WE WERE ABLE TO UNCOVER CRITICAL ISSUES, VALIDATE BACKEND FUNCTIONALITY, AND OPTIMIZE THE USER EXPERIENCE. BY EMPLOYING A RANGE OF TESTING TECHNIQUES — INCLUDING MANUAL, AUTOMATION, AND API TESTING — WE ENSURED A HIGHER QUALITY PRODUCT, CONTRIBUTING TO IMPROVED USER SATISFACTION AND BUSINESS SUCCESS.


THANK YOU