

# *WELCOME TO OUR PRESENTATION*

Project Name: ParaBank QA

SUBMITTED TO:

SIR Faisal Hafeez

SUBMITTED BY:

- Aleena Khan <ROLLNO# BSITM-23-65>
- Aiman Akbar <ROLLNO# BSITM-23-19>
- Arisha Khan <ROLLNO# BSITM-23-63>

SUBJECT:

Software Testing & Quality Assurance.

DEPARTMENT:

BSIT(morning SEC A) 5<sup>th</sup> semester.

**01**

## DESCRIPTION

- **Project Name:** ParaBank QA
- **Goal:** To perform a complete Quality Assurance check using Manual, Automation, and API testing.

**02**

## ACKNOWLEDGEMENT

- Special thanks to **Mr. Faisal Hafeez** for his guidance.
- This project helped us learn how to use professional testing tools in a real-world banking scenario.

**03**

## Roles

- Aleena Khan  
Project Documentation,  
Manual Testing, Project Oversight.
- Aiman Akbar  
Automation Engineer
- Arisha Khan  
API & Defect Manager

## What is ParaBank?

**ParaBank is a sophisticated, realistic online banking web application specifically designed for Software Testing and Quality Assurance (STQA) training.**

**It provides a robust environment to simulate real-world financial operations, allowing aspiring QA professionals to hone their skills in a practical setting.**



### 🔒 Securing Login

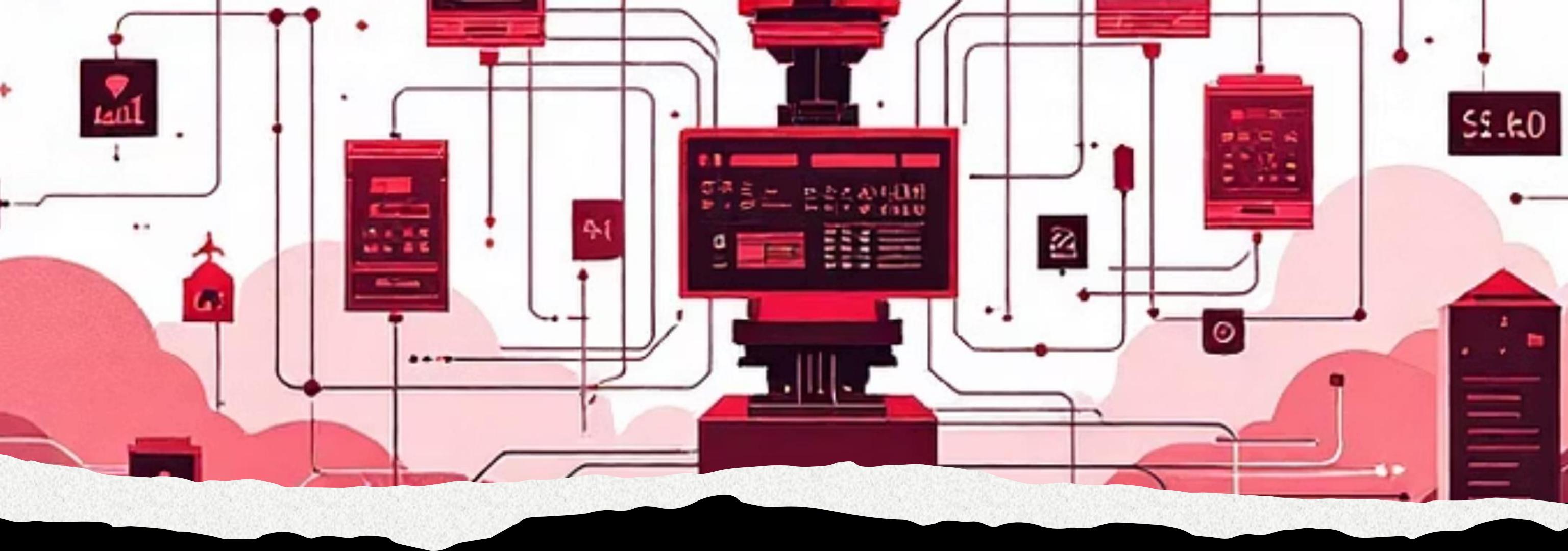
Secure Banking, Renewal login More

🔍 Save easier

Login

For New User | Continue od Banking Log

New Inquiry Inquiry



# Why ParaBank?

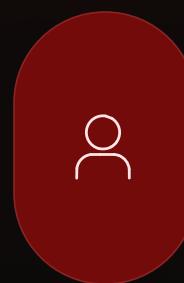
We selected ParaBank for its fidelity to a real-world financial environment. This allowed our team to apply professional tools—such as Playwright for advanced automation and Postman for comprehensive API testing—to a system that handles complex business logic and sensitive data simulations. It served as an ideal platform for practical, in-depth learning.

# Core Modules Tested



## User Management

Registration, login, and profile updates.



## Account Services

Opening new accounts, viewing balances.



## Fund Transfers

Initiating and verifying money transfers.



## Transaction History

Viewing and filtering past transactions.



# Manual Testing Strategy

(Project Lead & Manual QA)



## Requirement Engineering (SRD)

- Identified 11 main rules for how ParaBank should work.
- It gave the team a clear checklist so we didn't miss any important features like "Bill Pay" or "Login."
- Created a solid foundation for all our testing activities.

## Quality Strategy (Test Planning)

- Chose the tools (Playwright, Postman, Jira) and assigned roles to each member.
- It kept the team organized and ensured we had a plan for finding and fixing bugs.
- A professional workflow that combined manual, automation, and API testing.

## Traceability & Coverage (RTM)

- Created a map (RTM) that links every requirement to its specific test case.
- It proves to the instructor that we tested 100% of what we promised in the SRD.
- No requirement was left untested by the team.

## Final Quality Recommendation

- Looked at all the results and decided if the software is safe to use.
- My final advice is "No-Go"—the system is not ready for real customers yet.

# Manual Testing

The image is a collage of screenshots from the ParaBank website, illustrating various user interface elements and functionality:

- Customer Login:** Shows the login form with fields for Username (lenakhan) and Password, and buttons for LOG IN, Forgot login info?, and Register.
- Transfer Funds:** Shows the transfer funds form with fields for Amount (\$), From account # (17118), and To account # (17118), and a TRANSFER button.
- Transfer Complete:** Shows a confirmation message stating "\$0.00 has been transferred from account #17118 to account #17118." It also includes a link to See Account Activity for more details.
- Customer Login (Again):** Shows the customer login form with fields for Username (leena\_khan) and Password, and buttons for LOG IN, Forgot login info?, and Register.
- Sign Up:** Shows the sign-up form with fields for First Name (leena), Last Name (khan), Address (Housing 3), City (layah), State (punjab), Zip Code (31200), Phone # (03226578987), SSN (98765432567), and a REGISTER button.
- Welcome Page:** Shows the welcome page for leena\_khan, featuring a large blue header, a sidebar with links like Solutions, About Us, Services, Products, Locations, Admin Page, and a main content area with sections for ATM Services, Online Services, and Latest News.
- Admin Page:** Shows the admin page with a sidebar containing links for Open New Account, Accounts Overview, Transfer Funds, Bill Pay, Find Transactions, Update Contact Info, Request Loan, and Log Out.
- Logout:** Shows the logout confirmation message "You are now logged out. You can log back in at any time by visiting the login page." It includes links for Home, About Us, Services, Products, Locations, Forum, Site Map, and Contact Us.

# Manual Testing Outcomes & Verdict:

- **Comprehensive Coverage:** Identified 11 main functional requirements for ParaBank, creating a clear checklist to ensure no critical features (e.g., "Bill Pay," "Login") were overlooked.
- **Structured Approach:** Established a solid foundation for all testing activities by selecting tools (Playwright, Postman, Jira) and assigning precise roles, ensuring an organized and efficient bug-finding process.
- **Full Traceability:** Developed a Requirements Traceability Matrix (RTM) to link every requirement to its specific test case, guaranteeing that 100% of the SRD specifications were tested.



# Test Automation & Scripting

**Script Development:** Automated scripts for 8 critical test cases Registration, Login/Logout, and Navigation.



**Regression Testing:** Ensured that the most important "Happy Path" features remain stable and error-free through automation.



**Cross-Browser Validation:** Successfully ran tests across Chromium, Firefox, to ensure the site works on all browsers.



**Automation Framework:** Setup & configured Playwright framework in VS Code for automated testing.



Playwright Test Report

localhost:9323/#?testId=2e64c76ac68b6f0954a1-d13d2302e89625f60518

All 30 Passed 15 Failed 15 Flaky 0 Skipped 0

## TC\_01 Valid Registration

parabank.spec.js:10

chromium

Run

Errors

Test timeout of 30000ms exceeded.

Error: page.fill: Test timeout of 30000ms exceeded.  
Call log:  
- waiting for locator('#customer.firstName')

```
11 | await page.click('text=Register');
12 |
> 13 | await page.fill('#customer.firstName', 'Test');
     |
14 | await page.fill('#customer.lastName', 'User');
15 | await page.fill('#customer.address.street', 'Street 1');
16 | await page.fill('#customer.address.city', 'City');
at C:\playwright\tests\parabank.spec.js:13:14
```

Copy prompt

## Navigation links (TC\_22)

- Failed due to a Redirect Bug.
- Finding: The "Locations" link incorrectly sends users to an external site (parasoft.com) instead of the bank's page.

Test Steps

- > ✓ Before Hooks
- > ✓ Click locator('text=Register') — parabank.spec.js:11
- > ✘ Fill "Test" locator('#customer.firstName') — parabank.spec.js:13
- > ✓ After Hooks
- > ✓ Worker Cleanup

893n 26.111n 62n

**TC\_01 Valid Registration**

```

Error: page.fill: Test timeout of 30000ms exceeded.
Call log:
- waiting for locator("#customer.firstName")

11 | await page.click('text=Register');
12 | await page.fill('#customer.firstName', 'test');
13 | await page.fill('#customer.lastName', 'User');
14 | await page.fill('#customer.address.street', 'Street 1');
15 | await page.fill('#customer.address.city', 'City');
at C:\playwright\tests\parabank.spec.js:13:14

```

**TC\_02 Password Mismatch**

```

Error: page.fill: Test timeout of 30000ms exceeded.
Call log:
- waiting for locator("#customer.password")

22 | await page.click('text=Register');
23 | await page.fill('#customer.password', 'Pass123');
24 | await page.fill('#repeatedPassword', 'Pass456');
25 | await page.click('#input[value="Register"]');
26 |
at C:\playwright\tests\parabank.spec.js:34:14

```

**TC\_03 Missing Last Name**

```

Error: page.fill: Test timeout of 30000ms exceeded.
Call log:
- waiting for locator("#customer.firstName")

43 | await page.click('text=Register');
44 | await page.fill('#customer.firstName', 'Test');
45 | await page.click('#input[value="Register"]');
46 |
47 |
48 | await expect(page.locator('text=Last name is required')).toBeVisible();
at C:\playwright\tests\parabank.spec.js:45:14

```

- **TC\_01, 02, 03 (Registration): All failed due to Timeouts (30s).**
- **Finding:** The Registration page is unstable; automation cannot reliably find the input fields.

## Successful Login (TC\_05):

- Failed due to a "Strict Mode Violation".
- Playwright found two elements called "Accounts Overview," causing the script to stop because it didn't know which one to click.

testId=2e64c76ac68b6f0954a1-3e952ec48cac227fb8ec

All 30 | ✓ Passed 15 | ✗ Failed 15 | Flaky 0 | Skipped 0 | ↻ | ⚙️

« previous next »

### TC\_05 Successful Login

parabank.spec.js:52

chromium

✖ Run

Copy prompt

Errors

```
Error: expect(locator).toBeVisible() failed
Locator: locator('text=Accounts Overview')
Expected: visible
Error: strict mode violation: locator('text=Accounts Overview') resolved to 2 elements:
  1) <a href="overview.htm">Accounts Overview</a> aka getByRole('link', { name: 'Accounts Overview' })
  2) <h1 class="title">Accounts Overview</h1> aka getByRole('heading', { name: 'Accounts Overview' })

Call log:
- Expect "toBeVisible" with timeout 5000ms
- waiting for locator('text=Accounts Overview')

55 |   await page.click('input[value="Log In"]');
56 |
57 | ➤ 58 |   await expect(page.locator('text=Accounts Overview')).toBeVisible();
58 | });
59 |
60 | /* TC_06 - Invalid Password */
at C:\playwright\tests\parabank.spec.js:57:56
```

Test Steps

- ✓ Before Hooks 3.2s
- ✓ Fill "john" locator('input[name="username"]') — parabank.spec.js:53 78ms
- ✓ Fill "demo" locator('input[name="password"]') — parabank.spec.js:54 14ms
- ✓ Click locator('input[value="Log In"]') — parabank.spec.js:55 842ms

A screenshot of a test report interface. At the top, a navigation bar shows a URL, a search bar, and status filters: All 30, Passed 15, Failed 15, Flaky 0, Skipped 0, and a refresh icon. Below the filters, it says "3.4s". The main title is "TC\_06 Invalid Password" with the subtitle "parabank.spec.js:61". A "chromium" button is highlighted. A "Run" button is present. Under "Test Steps", there is a list of actions with their execution times: Before Hooks (2.8s), Fill "john" locator('input[name="username"]') — parabank.spec.js:62 (61ms), Fill "wrongpass" locator('input[name="password"]') — parabank.spec.js:63 (15ms), Click locator('input[value="Log In"]') — parabank.spec.js:64 (485ms), Expect "toBeVisible" locator('text=The username and password could not be verified') — parabank.spec.js:68 (146ms), and After Hooks (30ms). Navigation links "« previous" and "next »" are at the bottom.

## Negative Login Tests (TC\_06, TC\_07):

- Both Passed.
- The system correctly displayed error messages for "Invalid Password" and "Empty Login" attempts.

A screenshot of a test report interface, similar to the one above. It shows a URL, search bar, and status filters: All 30, Passed 15, Failed 15, Flaky 0, Skipped 0, and a refresh icon. Below the filters, it says "2.5s". The main title is "TC\_07 Empty Login" with the subtitle "parabank.spec.js:72". A "chromium" button is highlighted. A "Run" button is present. Under "Test Steps", there is a list of actions with their execution times: Before Hooks (2.0s), Click locator('input[value="Log In"]') — parabank.spec.js:73 (394ms), Expect "toBeVisible" locator('text=error') — parabank.spec.js:75 (143ms), and After Hooks (21ms). Navigation links "« previous" and "next »" are at the bottom.

Playwright Test Report

localhost:9323/#?testId=2e64c76ac68b6f0954a1-bcaff355ff178a0223c2

All 30 Passed 15 Failed 15 Flaky 0 Skipped 0 ⚙️ ⚙️

« previous next »

## TC\_08 Logout

parabank.spec.js:79 4.3s

chromium

✓ Run

Test Steps

> ✓ Before Hooks	2.6s
> ✓ Fill "john" locator('input[name="username"]') — parabank.spec.js:80	53ms
> ✓ Fill "demo" locator('input[name="password"]') — parabank.spec.js:81	11ms
> ✓ Click locator('input[value="Log In"]') — parabank.spec.js:82	785ms
> ✓ Click locator('text=Log Out') — parabank.spec.js:84	847ms
> ✓ Expect "toBeVisible" locator('input[value="Log In"]') — parabank.spec.js:85	29ms
> ✓ After Hooks	19ms

### Logout Functionality (TC\_08):

- Passed.
- The script successfully logged in and then logged out, returning the user to the login screen as expected.

# Automation Testing

## Automation Conclusion

- Stable Successes
- Systemic Failures
- Critical Catch
- Technical Verdict

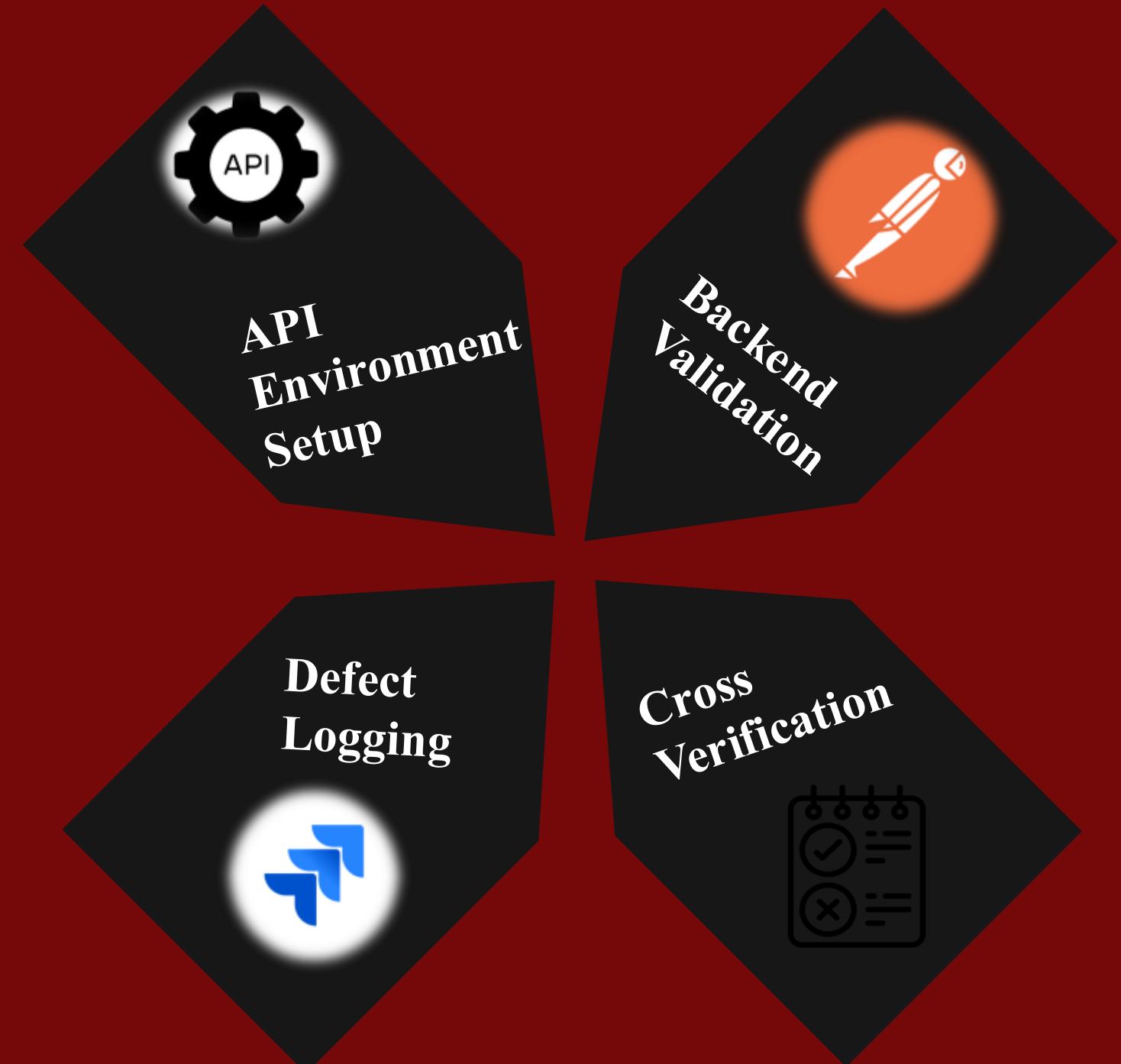
## Selection of Testing tool

### Defining the scope of Automation



# API & Defect Management

## (Backend QA Engineer)



# API Environment Setup



- **Tool Configuration:** Set up Postman to interact with the ParaBank REST API.
- **Endpoint Identification:** Identified the key URLs (endpoints) for services like Login, Registration, and Account Creation.

## Backend Validation (Postman)



- **Request Testing:** Sent POST and GET requests to verify that the server processes data correctly.
- **Response Analysis:** Checked for Status Codes (like 200 OK for success or 401 Unauthorized for failed logins).
- **Data Integrity:** Confirmed that the backend data (like account balances) matched what was shown on the frontend UI.

## Defect Logging (Jira)



- **Defect Manager:** logging all 7 identified bugs into Jira.
- **Evidence Collection:** Attached screenshots
- **Severity Rating:** Assigned priority levels (Critical, High, Medium) to help the "developers" know which bugs to fix first.

## Cross-Verification



- **Manual & Automation Support:** Verified the bugs found at the API level to see if the issue was in the UI or the database.
- **RTM Mapping:** Linked all API test results to the Requirements Traceability Matrix to ensure full backend coverage.

The screenshot shows two instances of the Postman application side-by-side. The left instance displays a POST request to {{baseUrl}}/transfer with parameters fromAccountId=14010, toAccountId=18561, and amount=1000. The response is a 400 Bad Request with the message "Transaction Failed: Insufficient funds for account 14010." The right instance shows a GET request to {{baseUrl}}/accounts/{{newAccountId}} returning a 200 OK status with XML content.

Left Window (Failed Call):

- Method: POST
- URL: {{baseUrl}}/transfer?fromAccountId=14010&toAccountId=18561&amount=1000
- Params:
  - fromAccountId: 14010
  - toAccountId: 18561
  - amount: 1000
- Body:

```
1 <error>
2   <message>Transaction Failed</message>
3   <details>Insufficient funds for account 14010.</details>
4 </error>
```
- Status: 400 Bad Request

Right Window (Successful Call):

- Method: GET
- URL: {{baseUrl}}/accounts/{{newAccountId}}
- Params:

Key	Value	Description
- Body:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?><account><id>18561</id><customerId>14010</customerId><balance>1000</balance><status>Active</status></account>
```
- Status: 200 OK

➤ General Access: Passed; the main index page returned a 200 OK status.

The screenshot shows a POST request to {{baseUrl}}/createAccount with parameters customerId=invalid\_customer, newAccountType=INVALID\_TYPE, and fromAccountId=non\_existent\_account. The response is a 200 OK status with XML content.

Method: POST

URL: {{baseUrl}}/createAccount?customerId=invalid\_customer&newAccountType=INVALID\_TYPE&fromAccountId=non\_existent\_account

Params:

Key	Value	Description
customerId	invalid_customer	
newAccountType	INVALID_TYPE	
fromAccountId	non_existent_account	

Body:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?><account><id>18561</id><customerId>14010</customerId><balance>1000</balance><status>Active</status></account>
```

Status: 200 OK

➤ Login & Account Creation:  
Passed; backend successfully processed login for user and created new account IDs.

➤ Negative Testing:  
Passed; the API correctly returned 400 Bad Request for invalid customer IDs or insufficient funds.

This screenshot shows the Postman interface with a successful POST request. The URL is `https://parabank.parasoft.com/parabank/service/bank/login?customerid=1&password=123456`. The response status is 200 OK, and the response body is XML containing account details.

This screenshot shows the Postman interface with a successful GET request. The URL is `https://parabank.parasoft.com/parabank/service/bank/login?customerid=1&password=123456`. The response status is 200 OK, and the response body is XML containing user information.

This screenshot shows the Postman interface with a successful GET request. The URL is `https://parabank.parasoft.com/parabank/service/bank/customers/1/accounts`. The response status is 200 OK, and the response body is XML containing account details.

This screenshot shows the Postman interface with a successful GET request. The URL is `https://parabank.parasoft.com/parabank/index.htm`. The response status is 200 OK, and the response body is HTML content.

**System allows \$0.00 transfer to same account ID.****Description**

ID: BUG\_001

Issue type: Bug

PRIORITY: High

Environment: Chrome

**Description:**

The banking system contains a logic error where it allows a user to initiate a fund transfer of \$0.00 to the exact same account ID.

**Steps to Reproduce:**

1. Login to the application and go to "Transfer Funds".
2. Select Account #17118 for both "From" and "To" fields.
3. Enter "0" as the amount.
4. Click the "Transfer" button.

**Actual Result:**

The system displays a "Transfer Complete!" message.

**Expected Result:**

The system should show an error message and the transaction should NOT be processed.

**Attachments** 1

**Navigation Inconsistency: "Home" link in footer does not redirect to the public landing page**

**Description**

Defect ID: BUG\_002

**Issue type:** Bug

**Priority:** Medium

**Steps to Reproduce:**

1. Log in to the ParaBank application.
2. Scroll down to the footer section at the bottom of the page.
3. Click on the "Home" link.

**Expected Result:**

The user should be redirected to the main public landing page (Main Index).

**Actual Result:**

Clicking "Home" simply refreshes the current logged-in dashboard instead of going to the main index.

**Environment:**

Web Browser (Chrome)

**Attachments** 1


  
Prove.jpeg  
20 Jan 2026, 02:55 AM

**Login fails with error message even when using valid credentials**

**Description**

Defect ID: Bug\_003

**Issue type:** Bug

**Priority:** High

**Steps to Reproduce:**

1. Open the ParaBank application.
2. Go to the Login section.
3. Enter a valid Username.
4. Enter a valid Password.
5. Click on the "Log In" button.

**Expected Result:**

The user should be logged in successfully and redirected to the Account Dashboard.

**Actual Result:**

The system displays an error message: "The username and password could not be verified" and the user is unable to login.

**Environment:**

Chrome

**Attachments** 1



# Jira Issue Summary

- **BUG\_001: Invalid Fund Transfer**
- **BUG\_002: Navigation Inconsistency**
- **BUG\_003: Login Instability**

## "Phone Number" field accepts alphabetic characters



### Description

Defect ID:Bug\_004

Issue type: Bug

Priority: Medium

### Steps to Reproduce:

1. Log in to ParaBank.
2. Click on "Update Contact Info".
3. In the "Phone #" field, enter alphabetic text (e.g., "InvalidPhone") instead of numbers.
4. Click "Update Profile".

### Expected Result:

The system should show an error: "Please enter a valid number".

### Actual Result:

The system accepts the text and updates the profile successfully.

## Request Loan: System accepts negative value for "Down Payment"



### Description

Defect ID:Bug\_005

Issue type: Bug

Priority: High

### Steps to Reproduce:

1. Log in to ParaBank.
2. Click on "Request Loan".
3. Enter "1000" in Loan Amount.
4. Enter "-100" (negative value) in Down Payment.
5. Click "Apply Now".

### Expected Result:

The system should reject the application and show: "Down payment cannot be negative".

### Actual Result:

The loan is approved successfully despite the negative down payment.

### Environment:

# Jira Issue Summary

- **BUG\_004: Validation Failure**
- **BUG\_005: Loan Process Error**



*Thank you!*