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) 5th 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

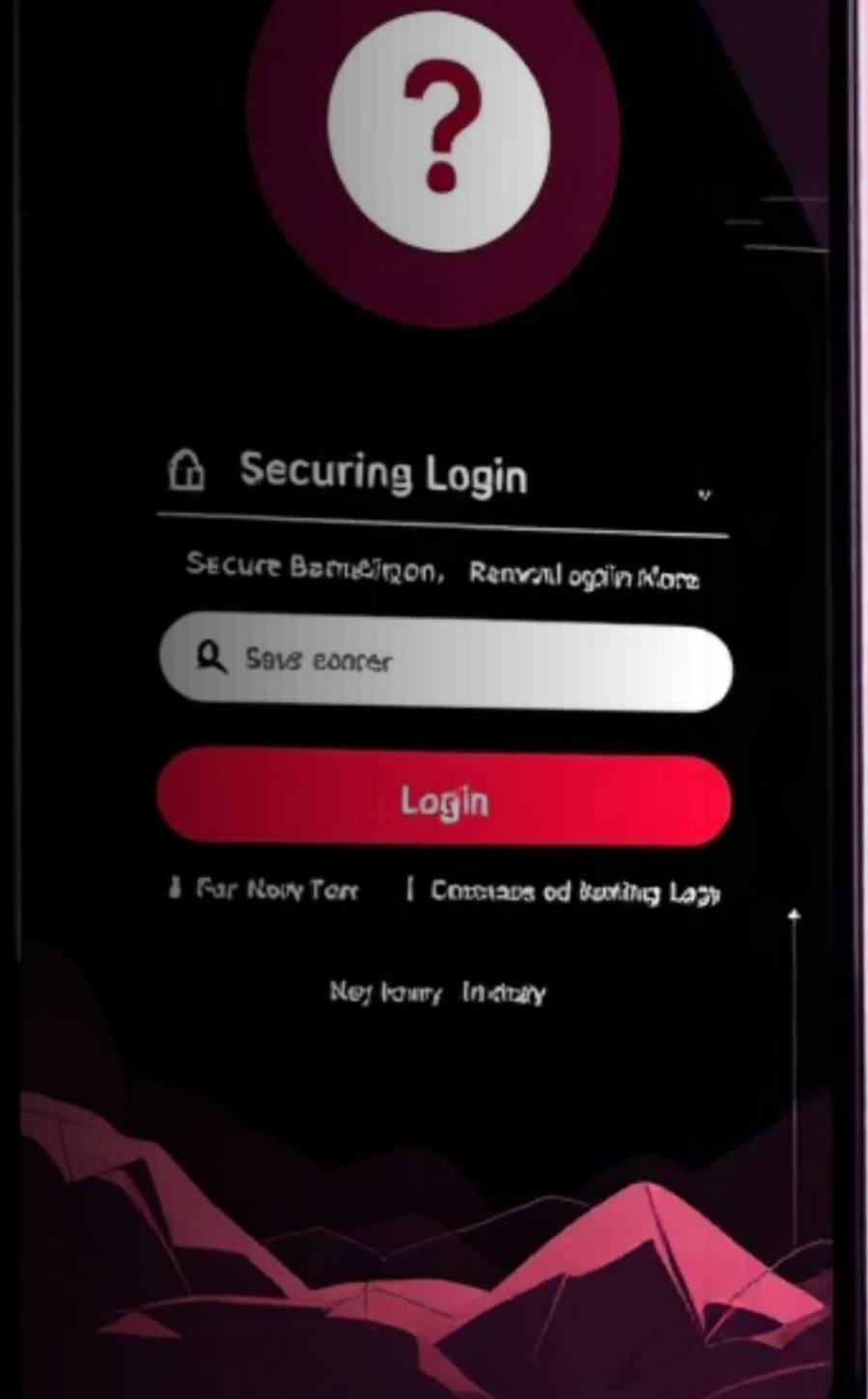
Roles

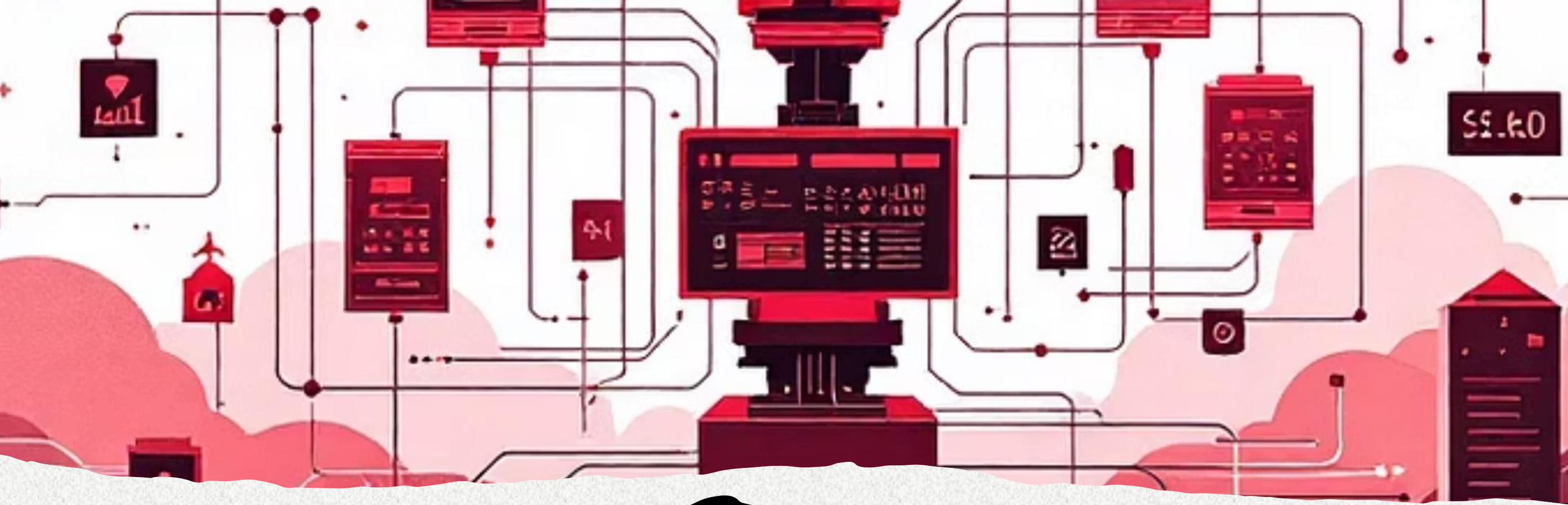
- **Aleena Khan**
  Project Documentation, Manual Testing, Project Oversight.
- **Aiman Akbar**
  Automation Engineer
- **Arisha Khan**
  API & Defect Manager

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.





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

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.

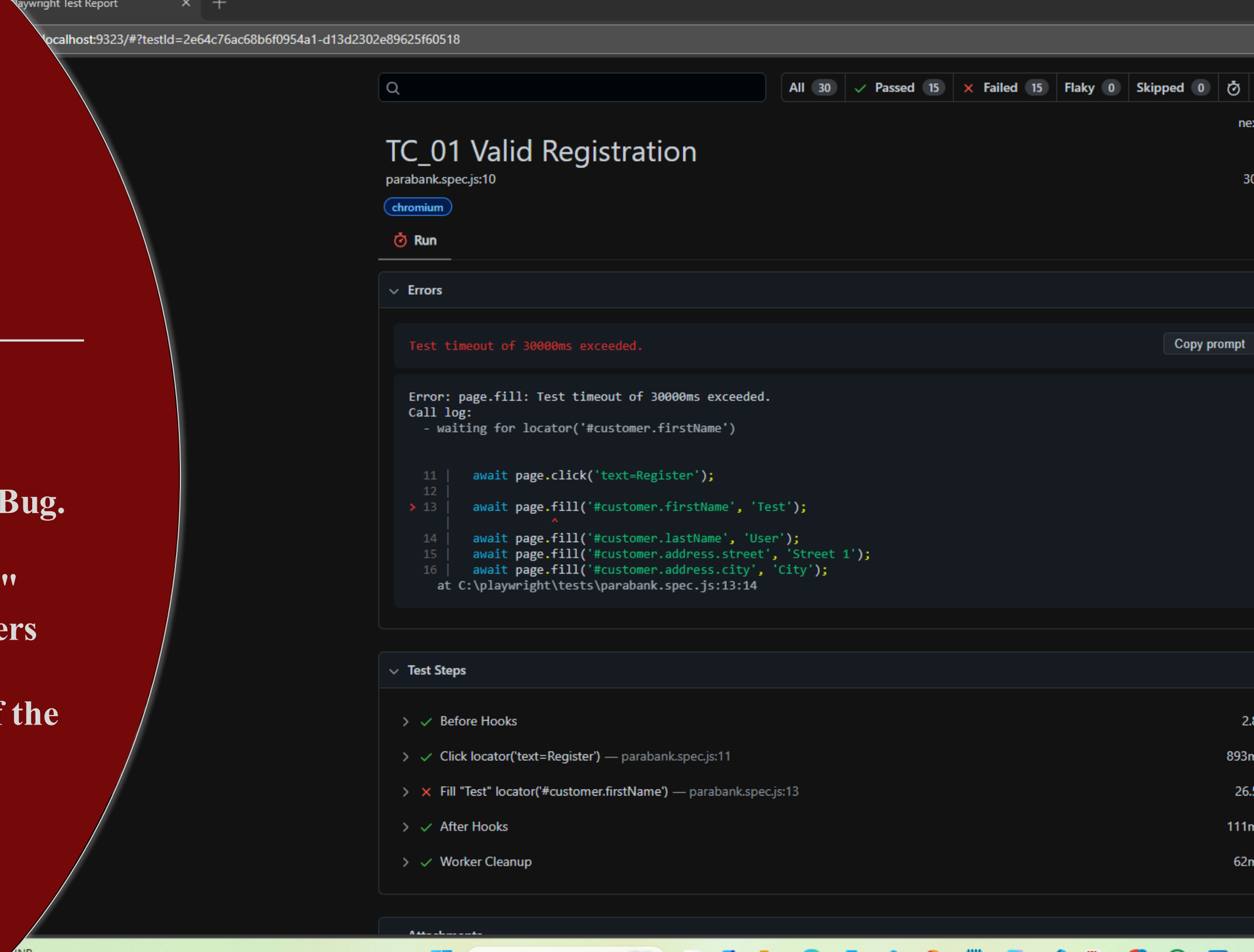
Regression Testing: Ensured that the most important "Happy Path" features remain stable and error-free through automation.

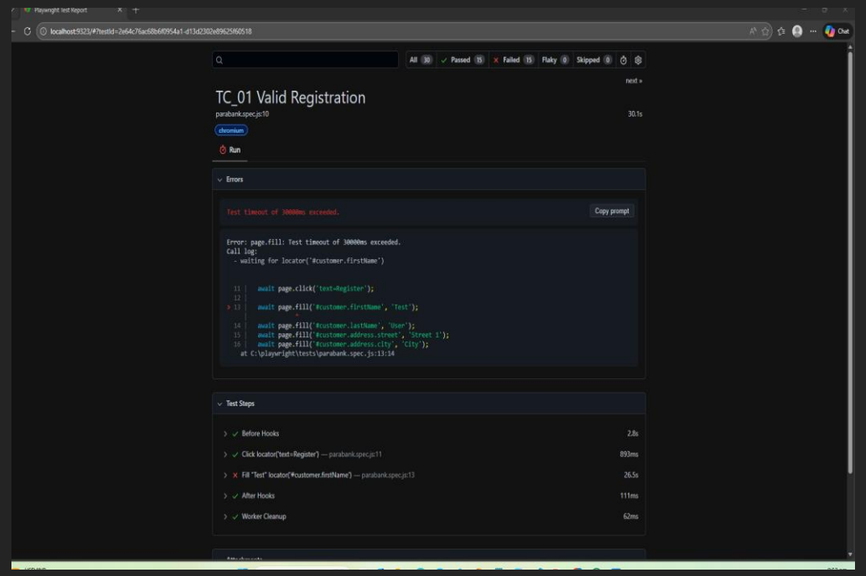
Cross-Browser Validation: Successfully ran tests across Chromium, Firefox,to ensure the site works on all browsers.

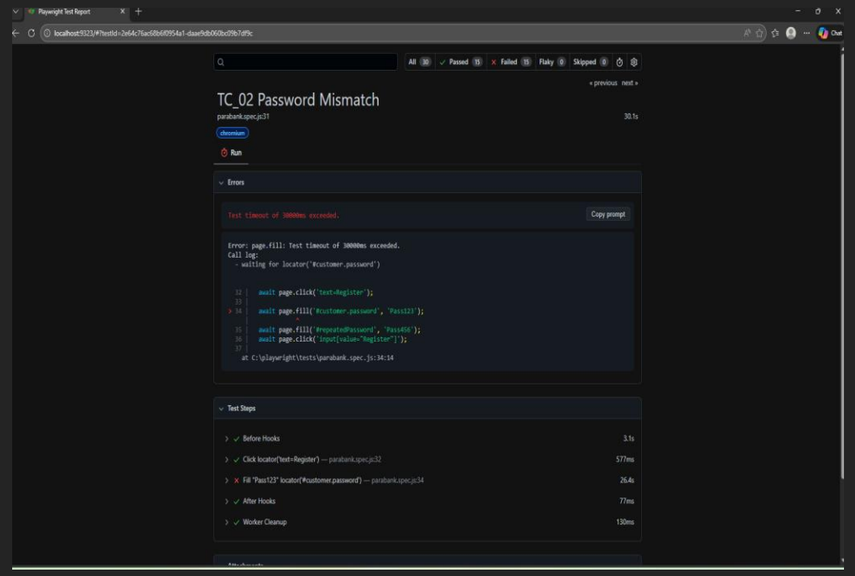
Automation Framework: Setup & configured Playwright framework in VS Code for automated testing.

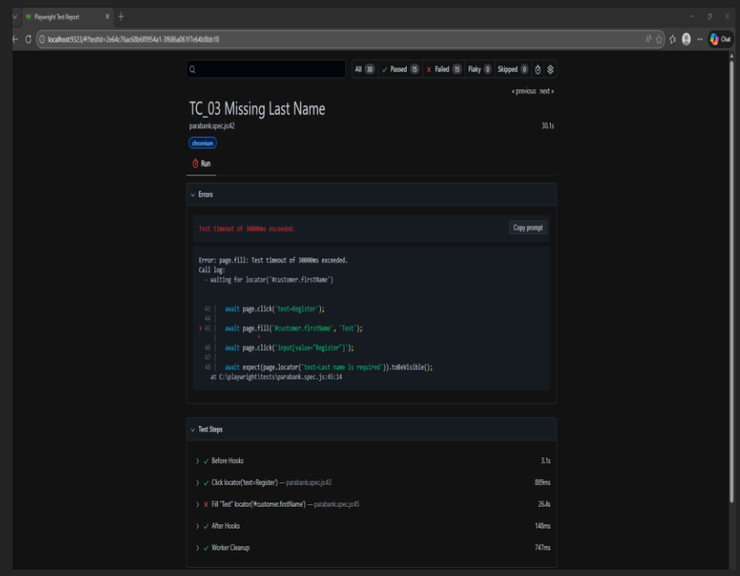
Navigation links (TC_22)

- 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.





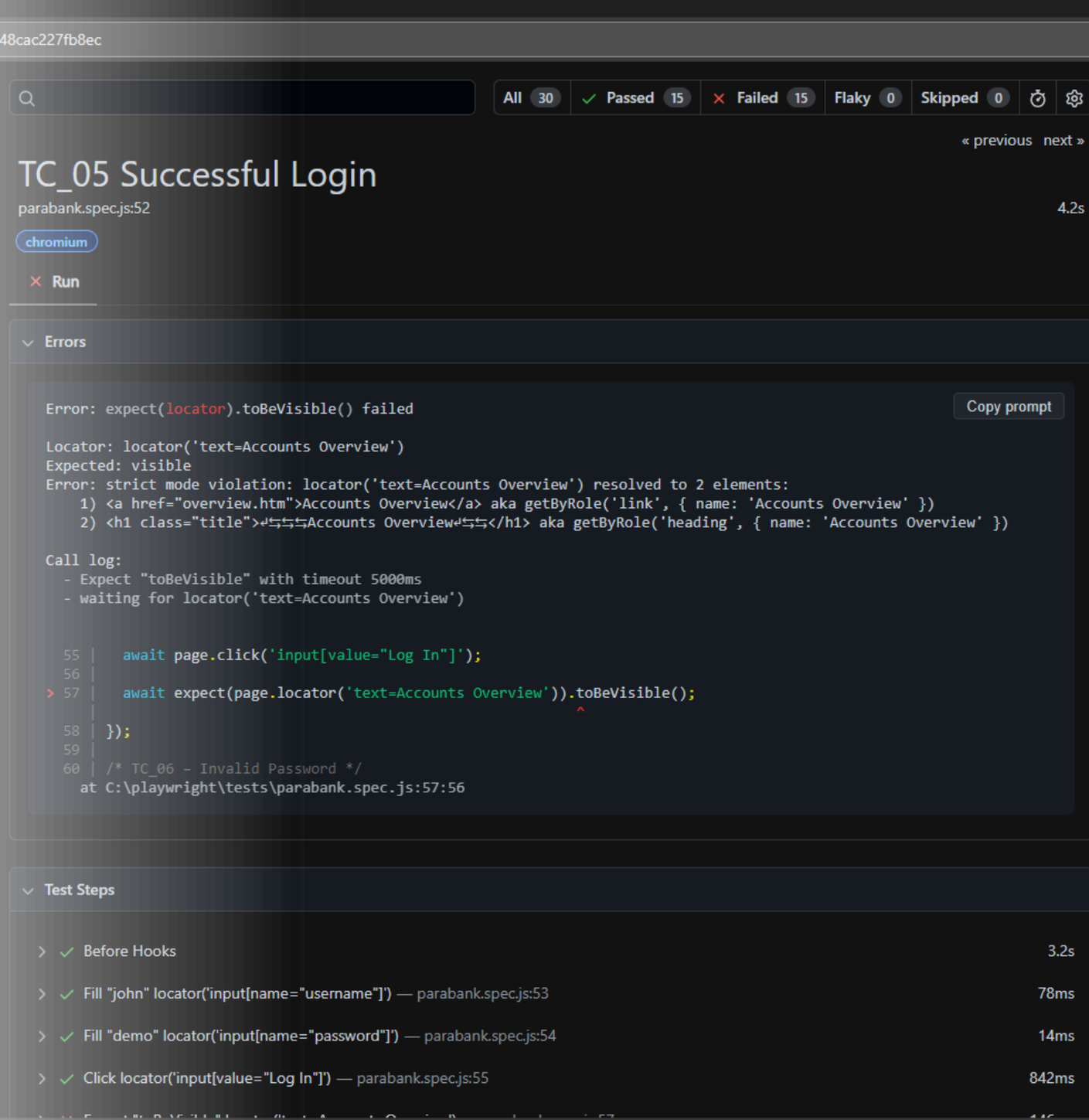




- **TC_01, 02, 03 (Registration): All failed due to Timeouts (30s).**
- **Finding: The Registration page is unstable; automation cannot reliably find the input fields.**

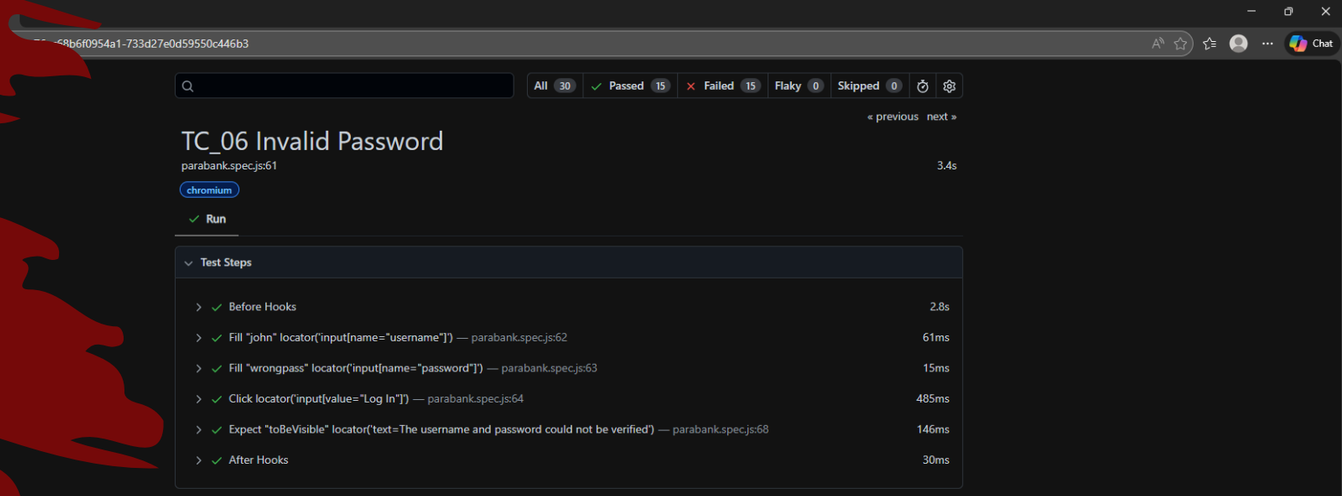
Successful Login (TC_05):

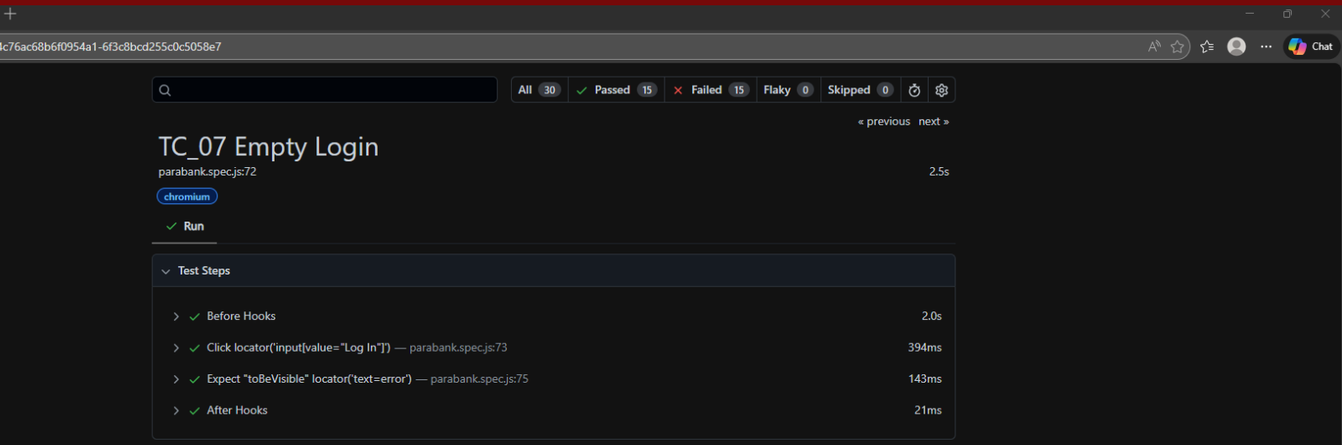
- Failed due to a "Strict Mode Violation".
- Playwright found two elements called "Accounts Overview," causing the script to stop because it didn't know which one to click.

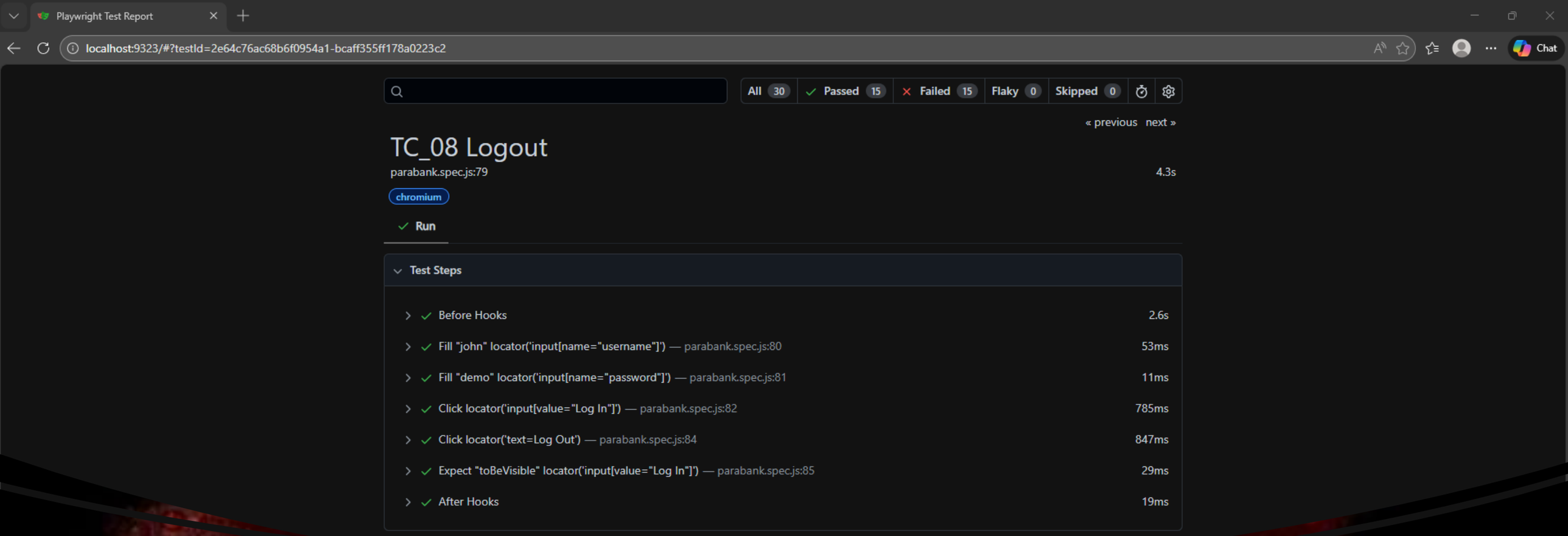


Negative Login Tests (TC_06, TC_07):

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







Logout Functionality (TC_08):

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

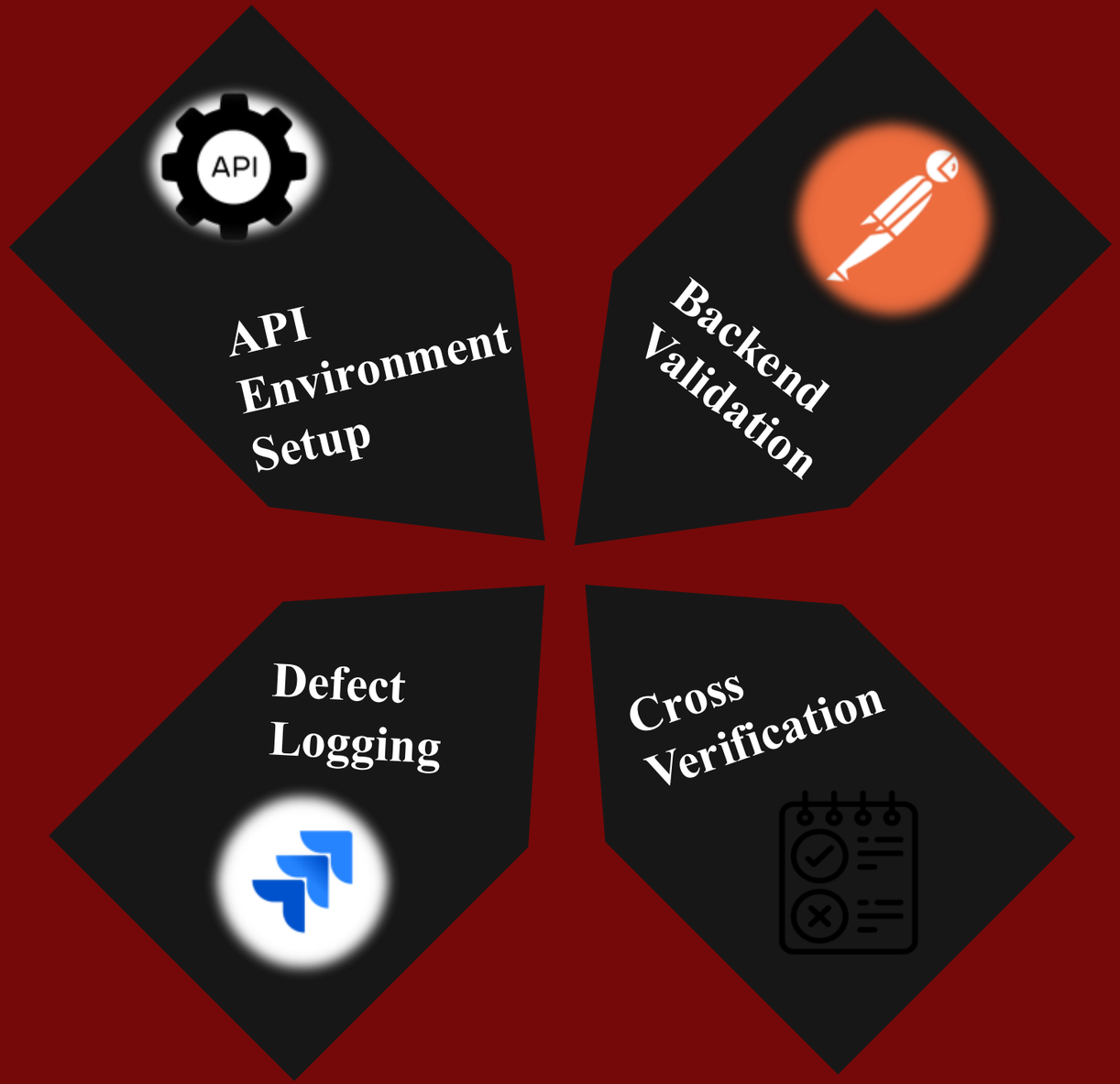
Automation Testi

Automation Conclusion

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

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)

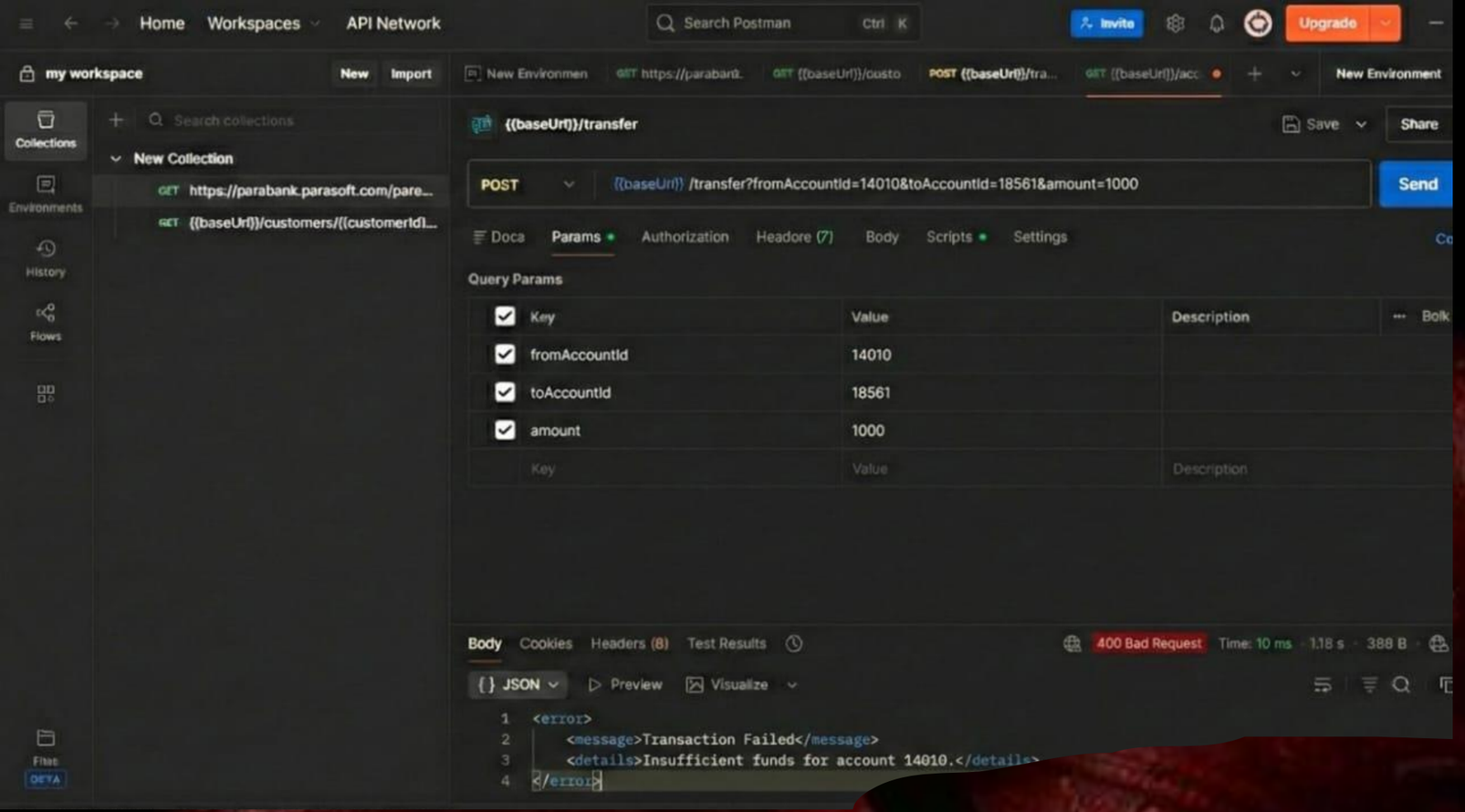
- Request Testing: Sent POST and GET requests to verify that the server processes data correctly.
- Response Analysis: Checked for Status Codes (like 200 OK for success or 401 Unauthorized for failed logins).
- Data Integrity: Confirmed that the backend data (like account balances) matched what was shown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

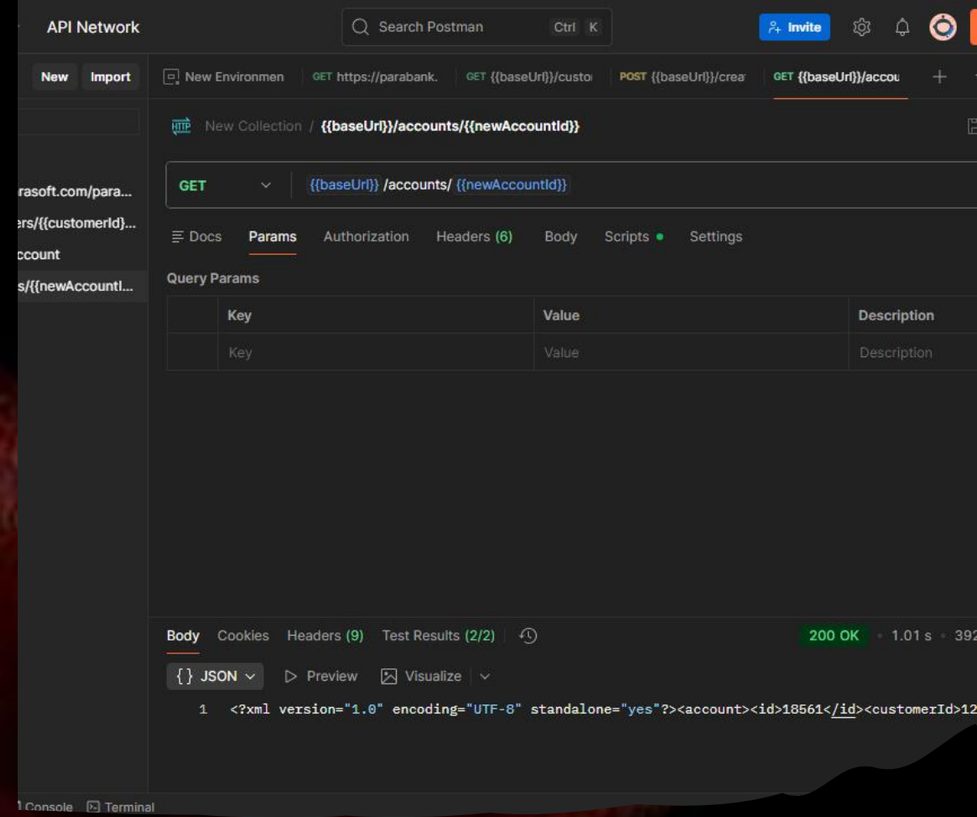
Defect Logging (Jira)

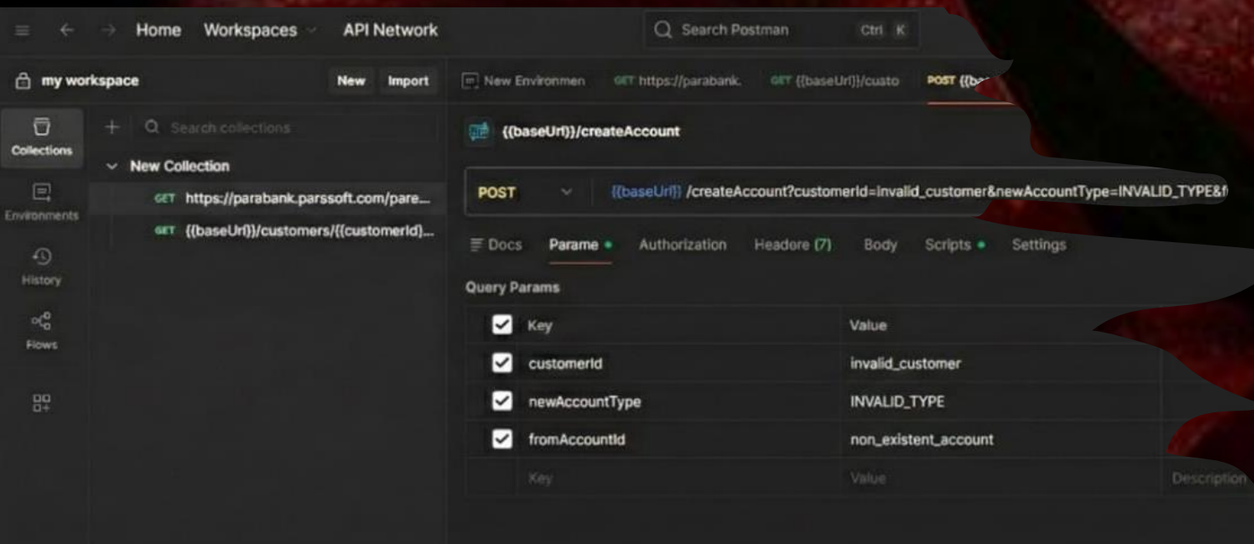
- Defect Manager by logging all 7 identified bugs into Jira.
- Evidence Collection: Attached screenshots
- Severity Rating: Assigned priority levels (Critical, High, Medium) to help the "developers" know which bugs to fix first.

Cross-Verification

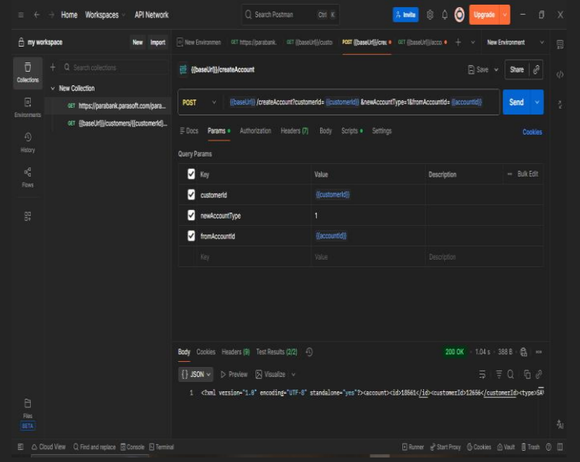
- Manual & Automation Support: Verified the bugs found at the API level to see if the issue was in the UI or the database.
- RTM Mapping: Linked all API test results to the Requirements Traceability Matrix to ensure full backend coverage.

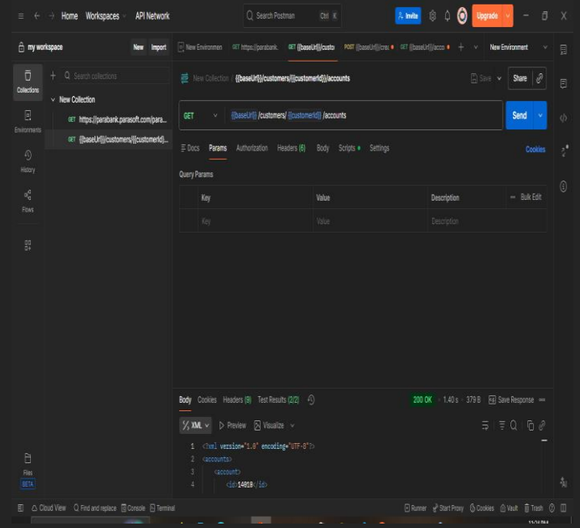






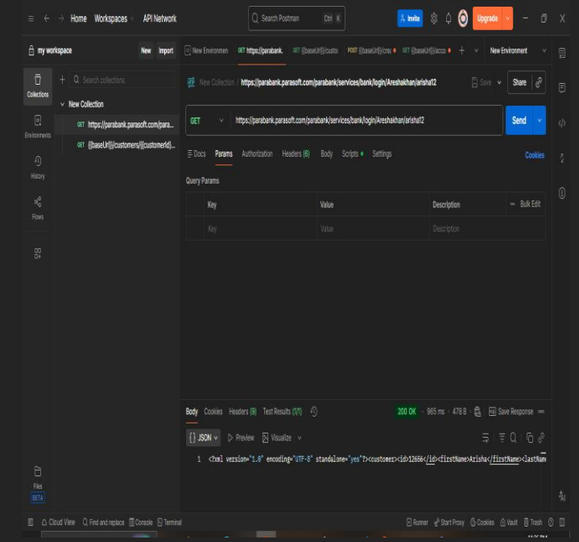
- **General Access: Passed; the main index page returned a 200 OK status.**

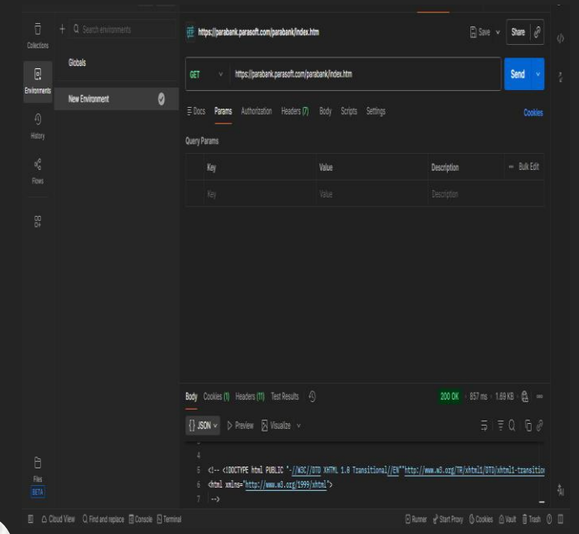


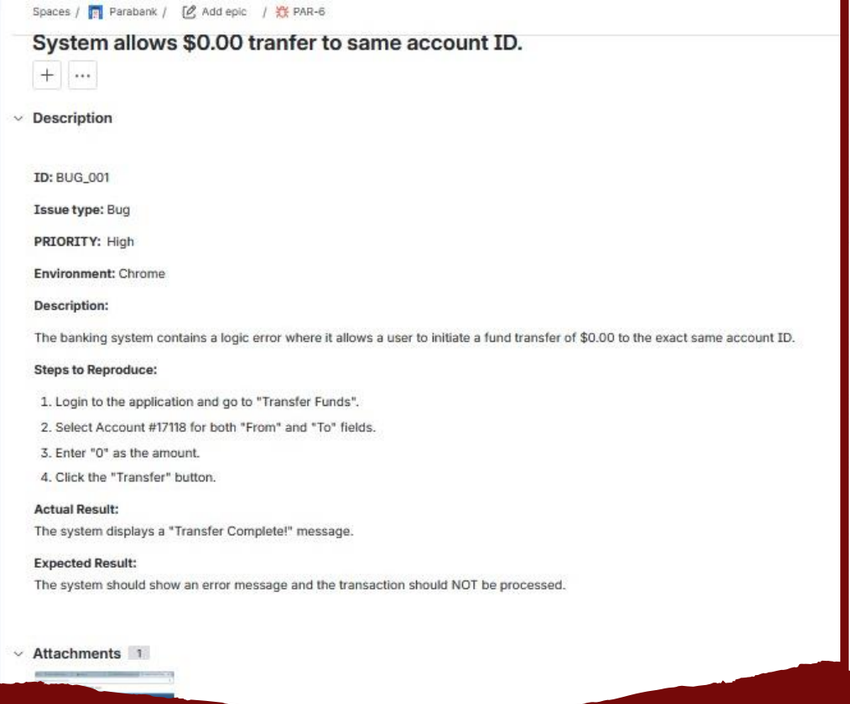


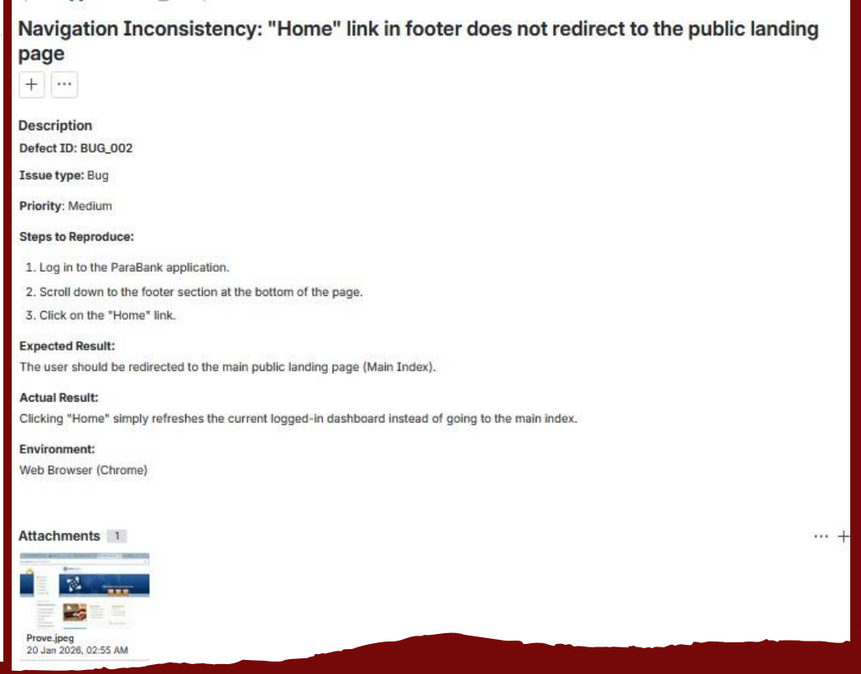
- **Login & Account Creation:**
  Passed; backend successfully processed login for user and created new account IDs.

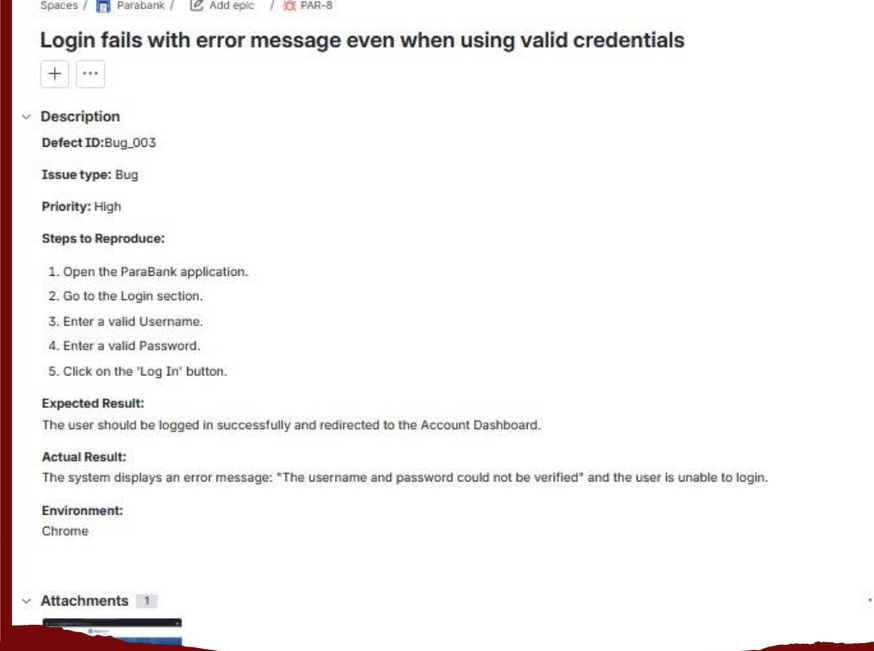
- **Negative Testing:**
  Passed; the API correctly returned 400 Bad Request for invalid customer IDs or insufficient funds.





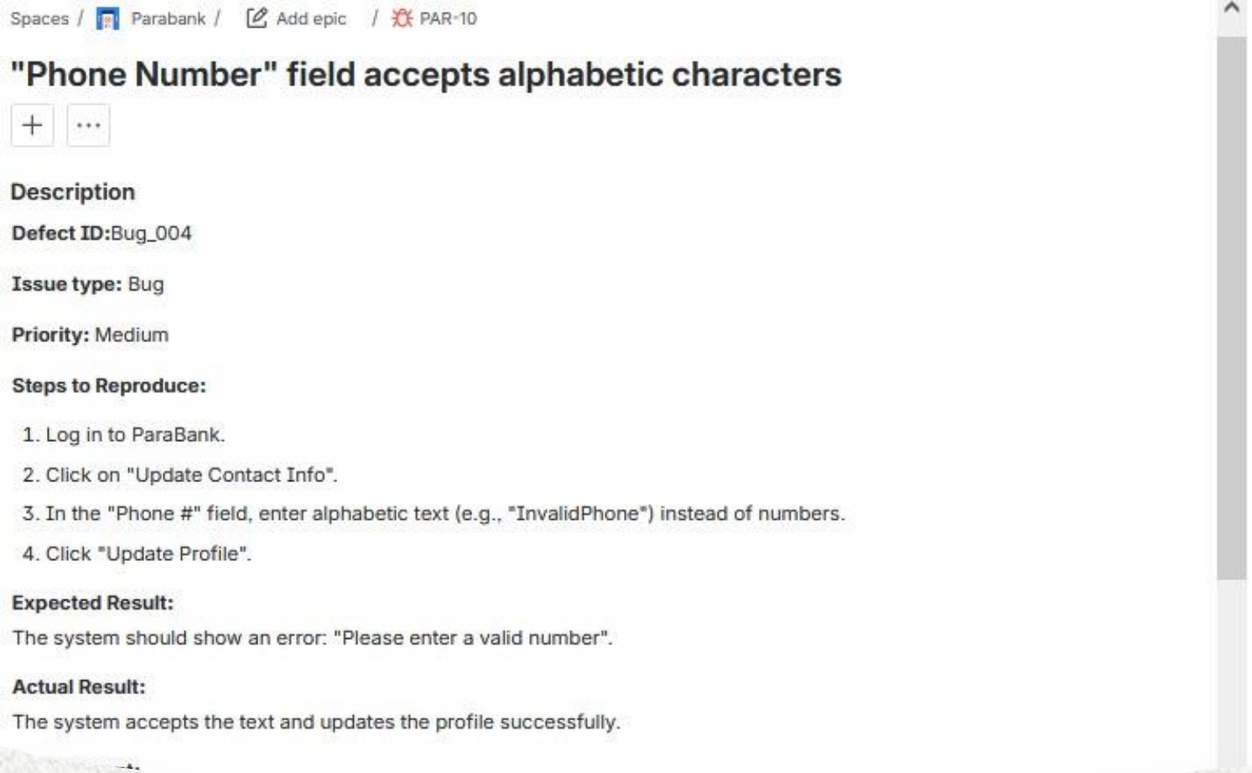


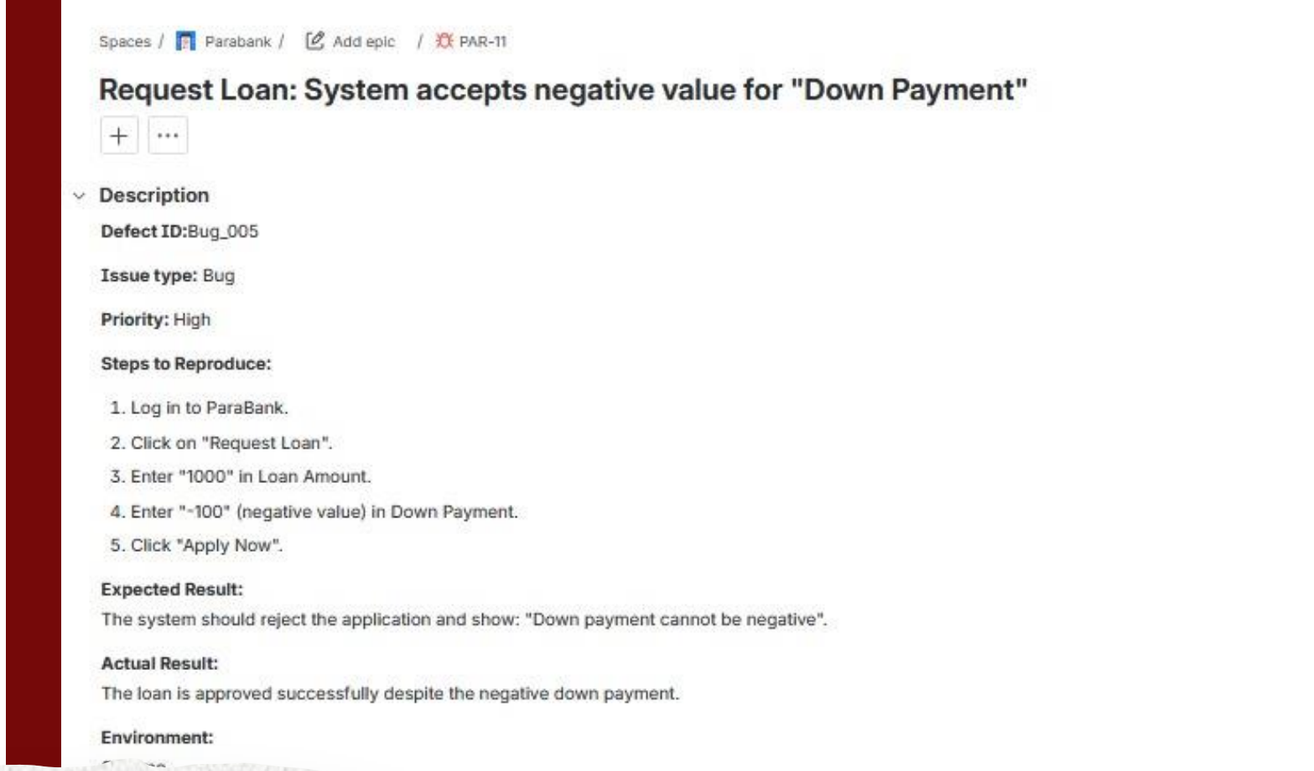




Jira Issue Summary

- **BUG_001: Invalid Fund Transfer**
- **BUG_002: Navigation Inconsistency**
- **BUG_003: Login Instability**





Jira Issue Summary

- **BUG_004: Validation Failure**
- **BUG_005: Loan Process Error**


Thank YOU!