AUTOAMTION MAJOR PROJECTS

In-House QA Automation Tool

Problem Statement: Legacy automation suite was slow, flaky, hard to scale, and not data-driven, resulting in poor regression time.

Tech Stack: Java, Selenium, Maven, BrowserStack, PL/SQL, Oracle, C#, .Net window service, Cloud Machines [ATS]

Interface: Web & API

Role & Contributions:

- Migrated legacy scripts to a robust Selenium-based hybrid framework.
- Implemented keyword + data-driven testing methodology.
- Enabled parallel UI test execution on a single machine.
- Integrated REST API calls for transaction flow testing.
- Managed dynamic pop-ups during validation & release phases.
- Created Excel-based scenario-driven execution.
- Supported Left-to-Right (L2R) functional execution.

- Accelerated releases and boosted stability by streamlining QA cycles and reducing defects.
- Delivered reusable automation that cut costs and increased stakeholder trust in deployments.

Mobile Automation Framework

Problem Statement: Mobile test coverage was poor, lacking backend validation, limited to local runs, and UI flakiness caused false failures.

Tech: Java, Appium, RestAssured, Zephyr, Headspin, TestNG, Maven

Interface: Mobile App (Android & iOS) & API

Role & Contributions:

- Combined UI + API automation to improve test reliability.
- Enabled parallel mobile UI and API executions.
- Embedded API calls in UI flows to simulate real user actions.
- Implemented Headspin device farm for real-device testing.
- Integrated framework with Jira Zephyr for traceability.
- Improved deployment quality by integrating tests in CI/CD pipelines.

- Boosted release confidence by improving mobile test reliability and expanding device coverage.
- Enabled scalable real-device testing for faster releases and consistent user experience.

E2E & Integration Automation Framework

Problem Statement: Lack of automation in integration test cases involving S3, DynamoDB, and downstream services.

Tech: Java, RestAssured, AWS S3, DynamoDB

Interface: API

Role & Contributions:

- Designed and developed automation test strategy and flow.
- Automated complex test flows like ASIN cloning across environments.
- Integrated validations across DDB audit logs and S3 Feed Stores.
- Created negative testing scenarios for robust coverage.
- Built downstream data verification logic for regression integrity.

- Automated backend workflows to boost integration reliability and ensure data consistency.
- Accelerated regressions and reduced risk with deep AWS validations and fewer manual efforts.

Shadow Verification Framework

Problem Statement: New app release introduced changes in data structure and submission, but lacked a reliable regression testing solution.

Tech: Python, Request.io, RDS

Interface: API

Role & Contributions:

- Designed a Shadow testing framework for parallel prod and new service comparisons.
- Validated real-time data between existing and new services using production traffic.
- Enabled regression coverage for scenarios with minimal effort.

- Mitigated release risk by validating new services against live production behavior.
- Ensured backward compatibility and faster releases with early defect detection.

Desktop Application Automation

Problem Statement: Legacy test suite had high failure rates and corrupted SIT database during executions.

Tech: VBScript, UFT, PS/SQL

Interface: Desktop

Role & Contributions:

- Refactored and stabilized existing UFT automation suite.
- Fixed test data dependencies and script logic errors.
- Introduced DB-level test data initialization and cleanup.
- Prevented SIT DB damage by reading test data from DB dynamically.
- Developed DB-based reporting for execution traceability.

- Stabilized legacy automation to cut false failures and eliminate SIT DB corruption.
- Improved regression safety with reliable tests and DB-driven reporting for better traceability.