Internship Progress Report

Name: Saiprakash Bollam

Internship Role: Research Intern

Duration: 20th August 2025 - 2nd September 2025

Organization: Computer Science Department, Binghamton University

Supervisor: Zerksis Umrigar

Email: umrigar@binghamton.edu

1. Introduction

This week concentrated on the implementation of automated test cases for the Smart Contact Manager project. Having previously completed feature enhancements such as Forgot Password, Relationship Attributes, Favorites, and QR Code functionalities, the focus shifted towards ensuring reliability, stability, and robustness of the system. AI tools were leveraged to design, write, and refine both unit and integration tests, while also suggesting coverage improvements.

2. Objective

The primary objectives were:

- Develop comprehensive unit test cases for services, controllers, and repository layers.

- Implement integration tests to validate multi-layer workflows.

- Ensure high code coverage and identify potential weak spots in logic.

- Use AI tools to generate and optimize test cases while reducing manual effort.

- Improve debugging workflows when tests failed, ensuring meaningful error resolution.

3. Features and Functionality Developed

- Created JUnit test cases for core services such as ContactService, UserService, and PasswordResetService.

- Implemented Mockito-based tests for service dependencies and repository interactions.

- Built integration tests for workflows like Forgot Password and QR Code scanning.

- Configured Spring Boot Test environment with in-memory H2 database for realistic test scenarios.

- Added assertions to validate entity persistence, relationship field updates, and favorite contacts logic.

- Automated validation of error handling paths, ensuring robustness against invalid inputs.

- Improved Maven build configuration for test execution and reporting.

4. AI's Role in Development

AI played an important role across multiple aspects of this week's work:

- Test Design: Suggested appropriate structures for JUnit and Mockito tests, including edge case handling.

- Code Generation: Provided boilerplate test methods that reduced repetitive coding effort.

- Debugging: Helped interpret cryptic stack traces when tests failed, pointing out misconfigured beans or missing annotations.

- Optimization: Suggested parameterized tests and reusability improvements in test suites.

- Coverage Analysis: Recommended strategies to increase coverage, especially for exception handling and invalid input scenarios.

Overall, AI significantly accelerated the testing process and deepened understanding of systematic validation in Spring Boot projects.

5. Challenges Faced

- Initial difficulty in configuring Spring Boot Test with H2 database.

- Mocking repository interactions correctly to avoid false positives.

- Thymeleaf-based tests for form submissions required additional setup.

- Verbose AI-generated tests sometimes needed trimming and alignment with project-specific naming conventions.

- Ensuring coverage did not compromise clarity and maintainability of tests.

6. Insights on AI Usage – Pros and Cons

Pros:

- Accelerated the process of writing structured and maintainable tests.

- Improved ability to debug failing tests with AI's explanations.

- Expanded understanding of test-driven development principles.

- Suggested innovative improvements like parameterized tests.

Cons:

- AI-generated tests occasionally duplicated existing logic or were unnecessarily verbose.

- Needed careful refinement to align with coding style and project context.

- Could not anticipate all real-world integration errors (e.g., environment-specific H2 issues).

7. Outcome and Learning

The testing phase was highly successful, with most core features now covered by automated unit and integration tests. This ensures higher reliability of the system and faster debugging of regressions in the future. AI support was invaluable in overcoming steep learning curves in JUnit, Mockito, and Spring Boot Test. The process reinforced the importance of rigorous testing as a cornerstone of production-ready software development.