Internship Progress Report

Name: Saiprakash Bollam
Internship Role: Research Intern
Duration: 23rd October 2025 – 5th November 2025
Organization: Computer Science Department, Binghamton University
Supervisor: Zerksis Umrigar
Email: umrigar@binghamton.edu

---

1. Introduction

This week focused on upgrading and stabilizing the **Spring Boot test environment** following the framework migration to **Spring Boot 3.5.0** and **Spring Security 6.5.0**. The main objective was to ensure all unit and integration tests executed correctly after refactoring several deprecated annotations, reworking test configurations, and updating security components.
AI assistants such as GitHub Copilot and ChatGPT played a central role in diagnosing errors, suggesting replacements for deprecated methods, and generating optimized configurations for complex security scenarios.

---

2. Objectives

- Fix and refactor all broken test cases after the Spring Boot 3.5 upgrade.
- Replace deprecated `@MockBean` with `@SpyBean` and configure proper test beans.
- Validate compatibility between `SecurityConfig` and `TestSecurityConfig`.
- Update JPA, repository, and session management configurations for test stability.
- Integrate AI suggestions for debugging and code validation.

---

3. Key Tasks and Implementations

- Updated **SecurityTest**, **SessionManagementTest**, and **ApiControllerTest** to work with the latest Spring Security APIs.
- Reconfigured `TestSecurityConfig` to include modern lambda-based security filter chains.
- Fixed test context loading errors caused by conflicting `securityFilterChain` beans.
- Created a `TestRepositoryConfig` for proper JPA setup during test execution.
- Mocked external dependencies such as `JavaMailSender` and `Cloudinary` to avoid runtime injection errors.

- Used AI to identify deprecated APIs like `alwaysUse()` and replaced them with new `defaultSuccessUrl()` methods.
- Implemented custom `PasswordEncoder` beans and ensured consistent authentication providers across tests.

---

4. AI's Role in Debugging and Development

AI systems like GitHub Copilot and ChatGPT provided essential assistance by:

- Detecting deprecated or removed Spring Security methods.
- Suggesting new bean definitions and configuration methods.
- Automatically identifying circular dependency issues in test contexts.
- Generating clean test configuration templates (e.g., `TestSecurityConfig`, `TestApiConfig`).
- Proposing structured debugging sequences for Maven build and test failures.
- Providing deeper explanations for error patterns such as context-loading exceptions.

---

5. Challenges Faced

- Frequent **application context failures** due to duplicate bean definitions between main and test security configurations.
- Test classes failing with **404 and 401 errors** for public endpoints due to misconfigured anonymous access.
- **Dependency conflicts** between `DaoAuthenticationProvider` constructors across Spring Security versions.
- Handling **test user creation** for authentication-based tests without persisting to the production database.

- Integrating mock external services (like Cloudinary) in test mode.

---

6. Insights on AI Usage

Advantages:

- Accelerated error resolution by automatically suggesting the most probable cause of failures.
- Provided migration-safe code snippets for newer Spring Boot and Security versions.
- Significantly reduced manual debugging time by generating ready-to-test configurations.
- Enhanced understanding of modern testing architecture and Spring dependency injection.

Disadvantages:

- Occasionally produced **deprecated or version-incompatible code**, requiring manual verification.
- Some suggestions introduced circular bean references that needed human review.
- Context-specific understanding (especially across test profiles) was limited.

---

7. Outcomes and Learnings

- By the end of this week, most critical test cases were successfully migrated and executed with minimal errors.
- The project now uses a **robust, AI-verified test configuration structure** with clearly separated layers for main and test environments.
- The debugging sessions provided deep insight into **Spring Boot 3.5 testing architecture**, **security context initialization**, and **AI-assisted dependency resolution**.
- This phase improved both development efficiency and understanding of **AI-human collaboration in enterprise software maintenance**.