Group 6 Sprint 6 Planning

Scrum Master

- Armas Nevolainen

Deliverables

- **Code Review Report:** code review will analyze the StudyShelf codebase using static analysis tools and will include visualizations of key metrics and specific code examples requiring attention.
- **GitHub Codebase:** Based on the review of the codebase, code will be refactored according to Java coding standards and best practices. Redundant or duplicated code will be removed and complex methods will be broken down in manageable units. Also test coverage will be expanded.
- **Acceptance Test plan:** The acceptance test plan will validate that the StudyShelf application meets all functional and non-functional requirements. Each test case will include detailed steps, expected results, actual results, and pass/fail status.
- **Localized version of database**

Tasks

Task 1: Statistical Code Review

Duration: 6 h
Team Members: Santtu Saaranen
Description: Conduct a static code analysis to identify quality issues, potential bugs, and optimization opportunities.

Task 2: Code Clean-Up

Duration: 6 h
Team Members: Santtu Saaranen + Armas Nevolainen
Description: Implement code improvements based on the findings from the statistical code review, focusing on readability, maintainability, and adherence to Java best practices.

Task 3: Acceptance Test Planning

Duration: 10 h
Team Members: Jiayue Zheng + Armas Nevolainen
Description: Develop a comprehensive acceptance test plan that validates all core functionalities of the StudyShelf application.

Task 4: Database Localization Implementation

Duration: 20h
Team Members: Jiayue Zheng + Armas Nevolainen
Description: Implement support for multiple languages in the database to enable localized content storage and retrieval.

User stories we have aimed to implement

- As a developer, I want to identify code quality issues using static analysis tools, so that I can understand which parts of the codebase need improvement.
- As a developer, I want to have metrics about code complexity and duplication, so that I can make informed decisions about refactoring.
- As a developer, I want to refactor complex methods into smaller, more manageable units, so that the code is easier to understand and maintain.
- As a developer, I want to eliminate duplicate code, so that future changes can be made more consistently.
- As a developer, I want refactored code to maintain existing functionality, so that users don't experience new bugs.
- As a developer, I want a complete set of acceptance criteria for each feature, so that I can verify the application meets user requirements.
- As a teacher user, I want to ensure that all course management features work correctly, so that I can effectively organize educational materials
- As a student user, I want to verify that all study material interactions function properly, so that I can access and engage with educational content.
- As an international user, I want to see content in my preferred language.