1. Project Overview

Project Title:

Quizzy Cards

Problem Summary:

Students often find it boring and struggle to study using traditional teaching methods, and many flashcard apps are too limited, not engaging enough or behind paywalls. Teachers may also have trouble finding a solution for these students to make studying something rewarding and motivating.

Intended Audience:

Students who want to create and study flashcards and quizzes. Teachers who want to upload study sets, create quizzes, and monitor student progress.

Main Features:

- Students can create and study custom flashcards and quizzes
- Teachers can upload and share flashcard sets with their students
- Progress tracking and stats for students, and analytics for teachers

2. Project Objectives

- Authentication for teachers and students.
- Flashcard creation for both sides - teachers and students (flashcard has to be in a set).
- Creation and deletion of flashcard sets, for example Finnish language set 1.
- When the set is done - the summary (percentage of how many correct).
- The teacher can see the students' statistics (percentage of correct and incorrect answers).
- The teacher can see the percentage of flashcards, for example Finnish language set 1, card 4 - 80 % of the time students get it wrong.

- The teacher can share sets with everyone having an account (they’re public to everyone using the app).
- People using the app without an account won’t have their statistics saved in the database (but they can see their own statistics after they’ve done the set).

3. Scope and Deliverables

In scope:

- Flashcard creation and study for students
- Quizzes from flashcard sets
- Teacher material upload
- User authentication
- Progress tracking
- JavaFX GUI and MariaDB integration
- Basic testing

Out of scope:

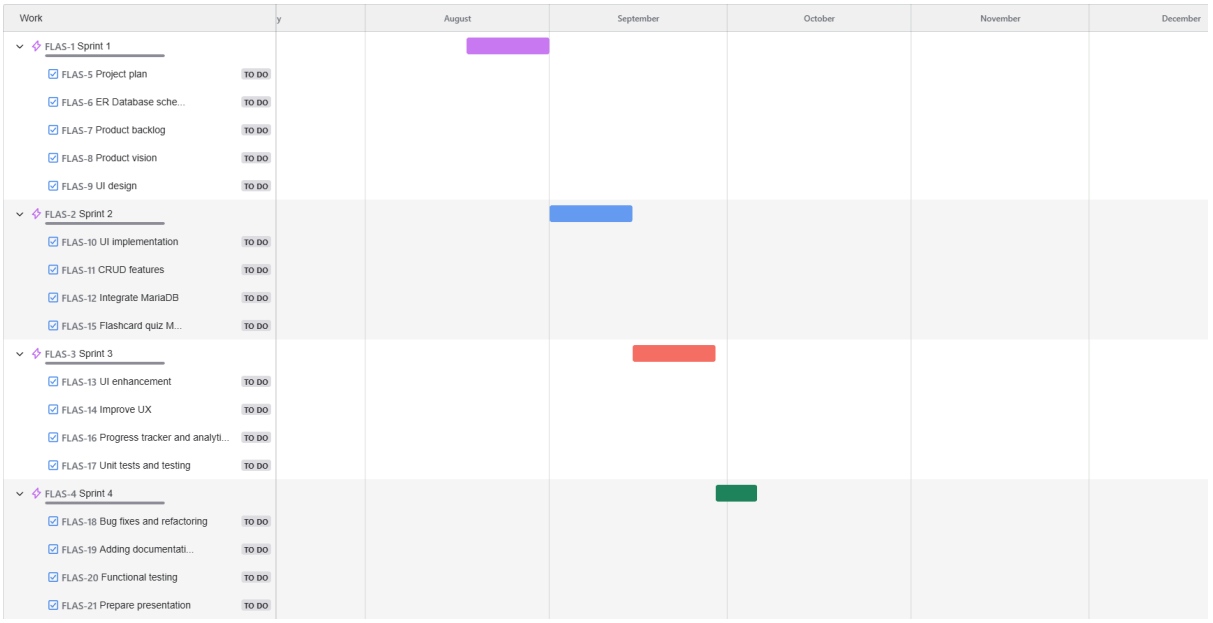
- Mobile app
- Web application
- More complex management features (assignments etc)

Key deliverables:

- Functional JavaFX application
- Documentation
- Presentation
- Sprint reviews

4. Project Timeline

This section describes the planned project timeline. Each sprint has specific goals and deliverables. See picture below.



Sprint 1: Planning

Duration: 21.08 - 31.08

Objectives:

- Create project plan
- Create ER diagram for the database
- Create the initial product backlog
- Discuss and define the product vision and features
- Begin UI design using mockups

Sprint 2: Start of development

Duration: 01.09 - 14.09

Objectives:

- Develop core user interface using JavaFX
- Implement CRUD functionality
- Integrate MariaDB with the application
- Develop the initial minimum viable product (MVP) of the application
- Ensure data persistence with the database

Sprint 3: Continue development + testing

Duration: 15.09 - 28.09

Objectives:

- Improve and polish UI/UX
- Implement progress tracking for students
- Add basic analytics for teachers
- Unit testing with JUnit
- Manual UI and application testing

Sprint 4: Finalization

Duration: 29.09 - 05.10

Objectives:

- Final bug fixes and performance improvements
- Refactor code, if necessary
- Complete all project documentation
- Final application testing
- Prepare the project presentation

5. Resource Allocation

This section describes the roles and responsibilities of each team member, and the resources required to complete the project.

- Team Members and Roles:
  - Otto Pärnänen: Developer, Scrum Master.
  - Jarmo Illikainen: Developer, UI and UX
  - Aaron Lyhtinen: Developer, Database and Testing
- Software, Hardware, and Tools:
  - Java
  - MariaDB
  - JavaFX and Scenebuilder
  - JUnit
  - Trello
  - IntelliJ IDEA
  - Github
  - Laptop/PCs
- External Resources or Support:
  - https://github.com/ADirin/OTP1_LectureMaterial
  - JavaFX tutorials
  - Stack Overflow

6. Risk Management

This section describes the potential risks that may affect the project timeline, quality or completion of the project.

Description	Likelihood	Impact	Mitigation
Database connection failures	Medium	High	Early DB testing, exception handling
GUI responsiveness issues	Medium	Medium	Break features into small components
Merge conflicts in git	Low	Medium	Smart branching, frequent commits
Scope creep	Medium	High	Stick to MVP features, prioritize backlog realistically
Difficulty implementing animations	Medium	Low	Use basic JavaFX transitions first, refactor later if needed
Bugs in progress tracking logic and analytics	Medium	High	Write unit tests, test edge cases early

7. Testing and Quality Assurance

This section describes the testing strategies that will be used for this project to ensure a stable and functional application.

Types of testing:

- Unit testing for individual classes and methods
- Testing interactions between components (Database queries, UI bindings)
- Testing the application manually for core user interactions (flashcard flipping, navigation, creating flashcards, progress tracking)

Success Criteria:

- No critical bugs
- All features functional

- Flashcards and quizzes work with correct data
- Clean UI and valid progress stats
- Application handles invalid input or missing data correctly

Tools:

- JUnit

8. Documentation and Reporting

This section describes the documentation that will be produced throughout the project, and how progress will be tracked via both project management and informal communication channels.

Planned documentation:

- Technical documentation (architecture, DB schema, code comments)
- Setup guide in readme
- Simple usage guide for users of the application in readme
- Final presentation

Progress reporting:

- Trello board updates
- Sprint review reports
- Discord updates for informal discussions and daily updates to ensure team is aware of what is happening regarding the project