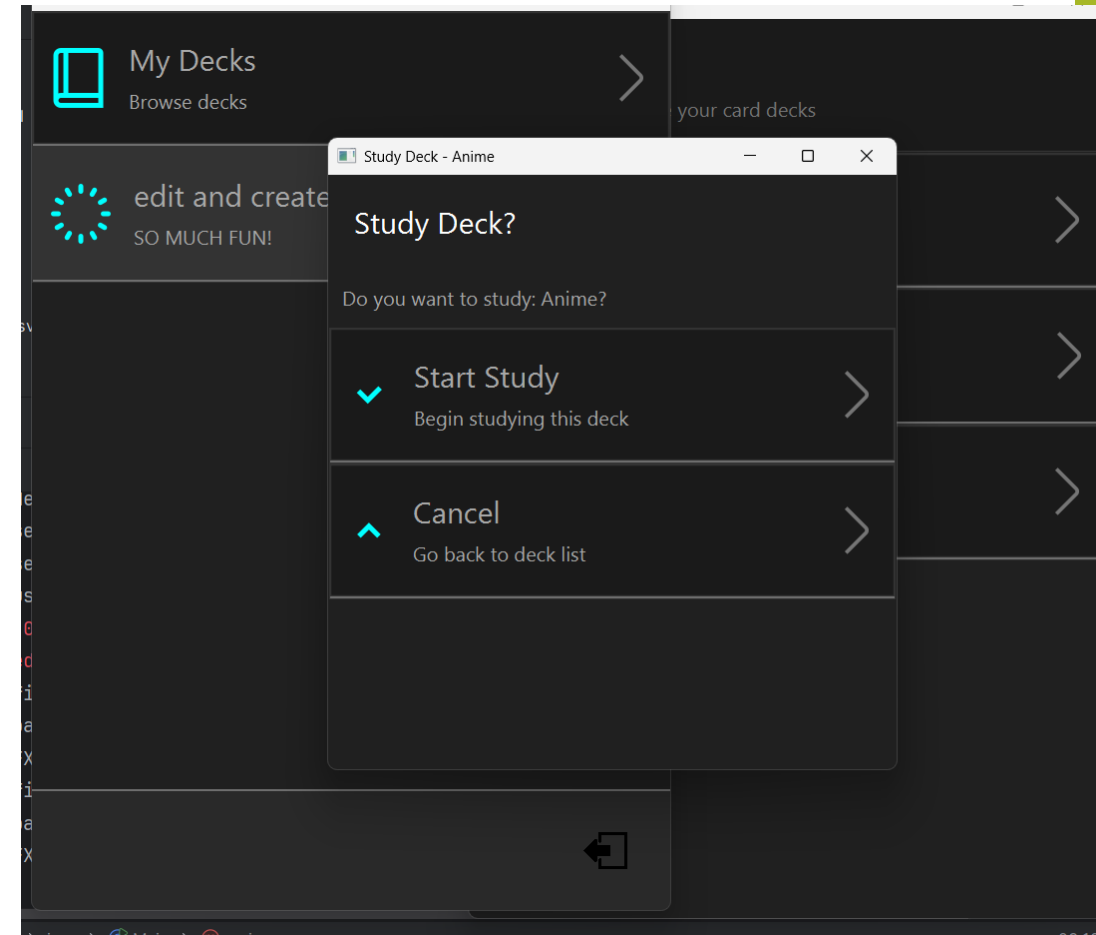# MEMORYCARD GAME

Group 2
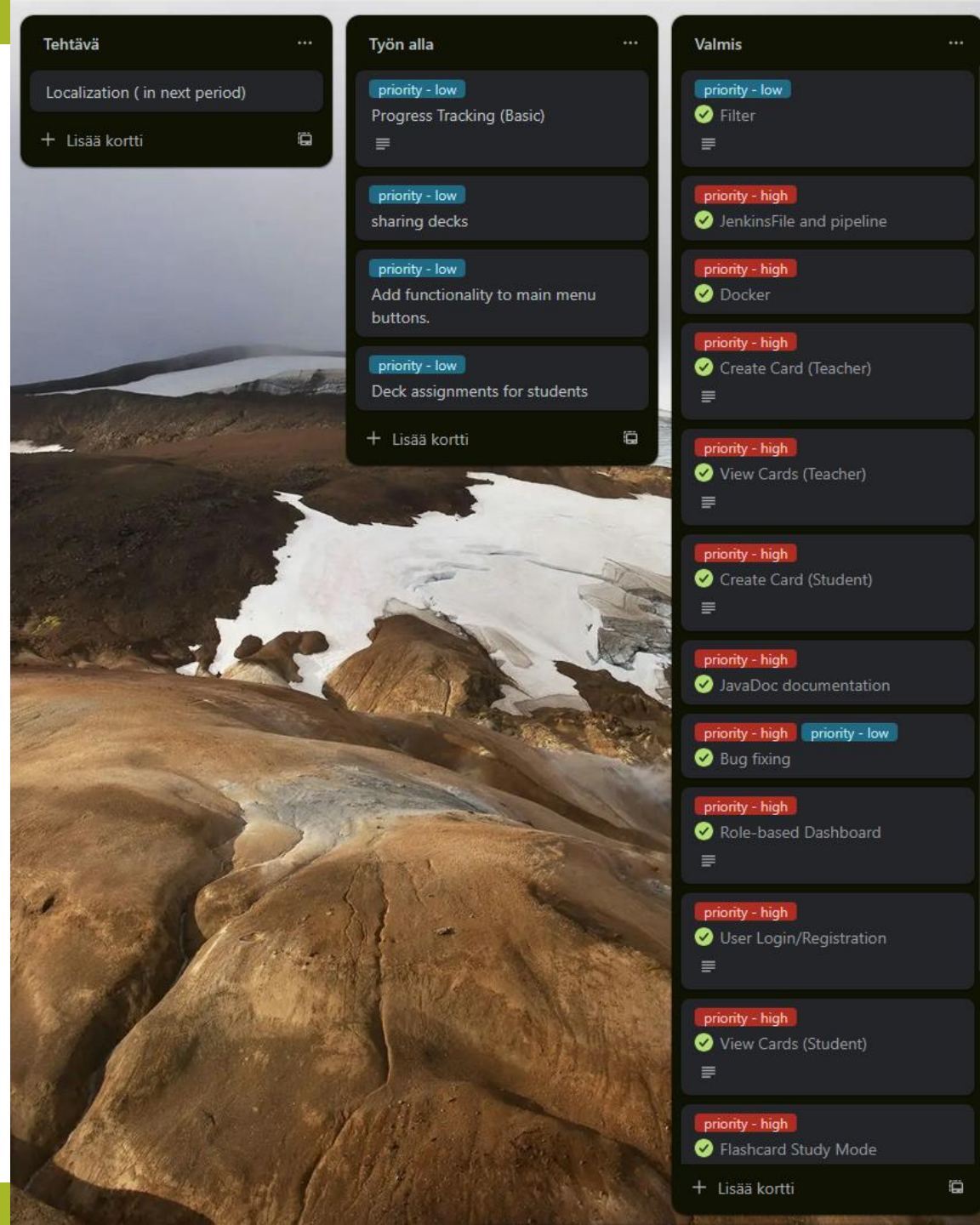
Axel Nokireki ,Vladimir Karpenko ,Georgii Afanasev ,Patrik Skogberg

Introduction

- Use of AI: only in code comments (Github copilot)
- GitHub link
- Trello link

Product Backlog

Tehtävä

Localization (in next period)

+ Lisää kortti

Työn alla

priority - low
Progress Tracking (Basic)

priority - low
sharing decks

priority - low
Add functionality to main menu buttons.

priority - low
Deck assignments for students

+ Lisää kortti

Valmis

priority - low
✓ Filter

priority - high
✓ JenkinsFile and pipeline

priority - high
✓ Docker

priority - high
✓ Create Card (Teacher)

priority - high
✓ View Cards (Teacher)

priority - high
✓ Create Card (Student)

priority - high
✓ JavaDoc documentation

priority - high   priority - low
✓ Bug fixing

priority - high
✓ Role-based Dashboard

priority - high
✓ User Login/Registration

priority - high
✓ View Cards (Student)
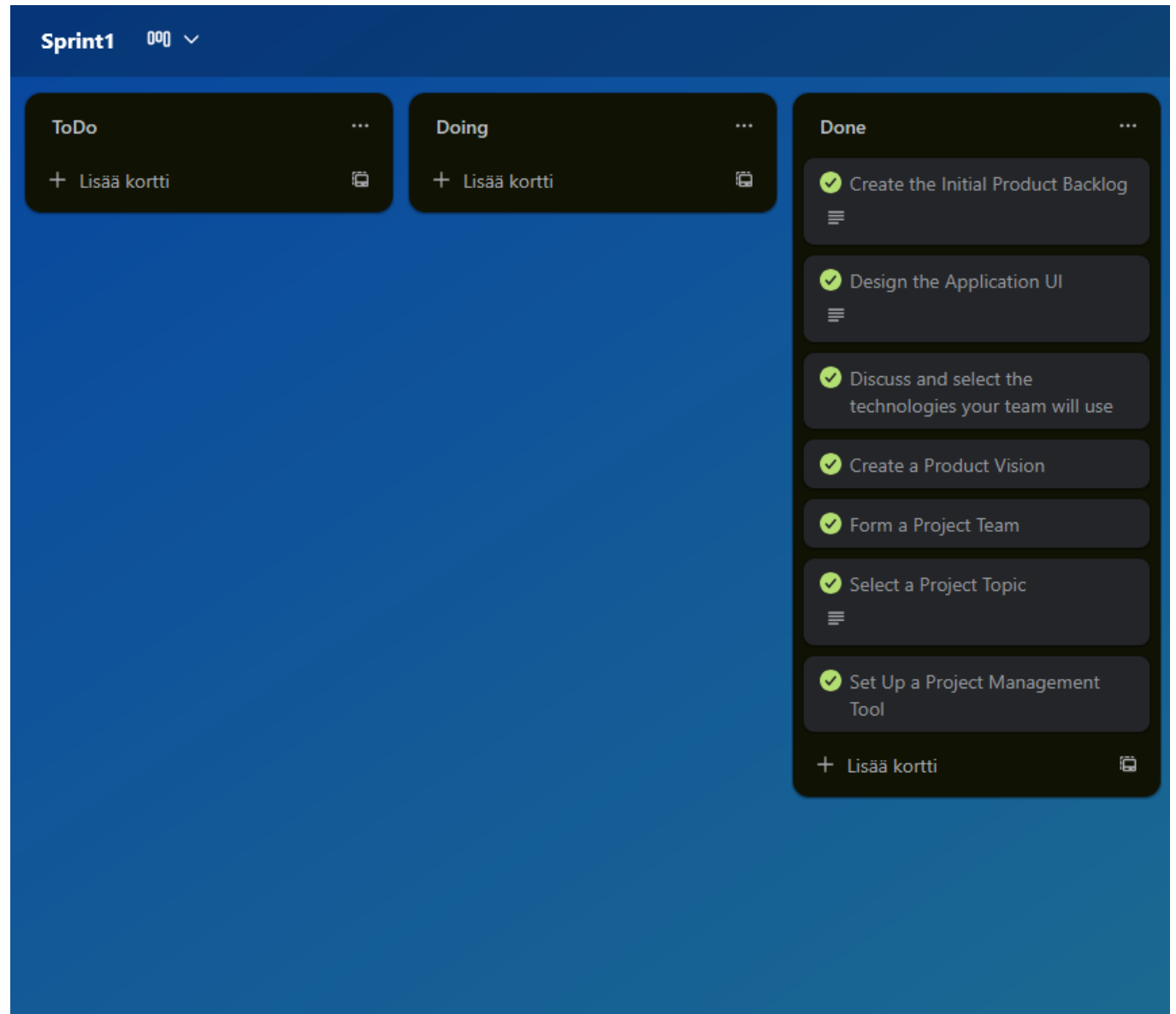
priority - high
✓ Flashcard Study Mode

+ Lisää kortti

Sprint 1

- Sprint Master: Axel
- Sprint Backlog -->
- Done = All of it
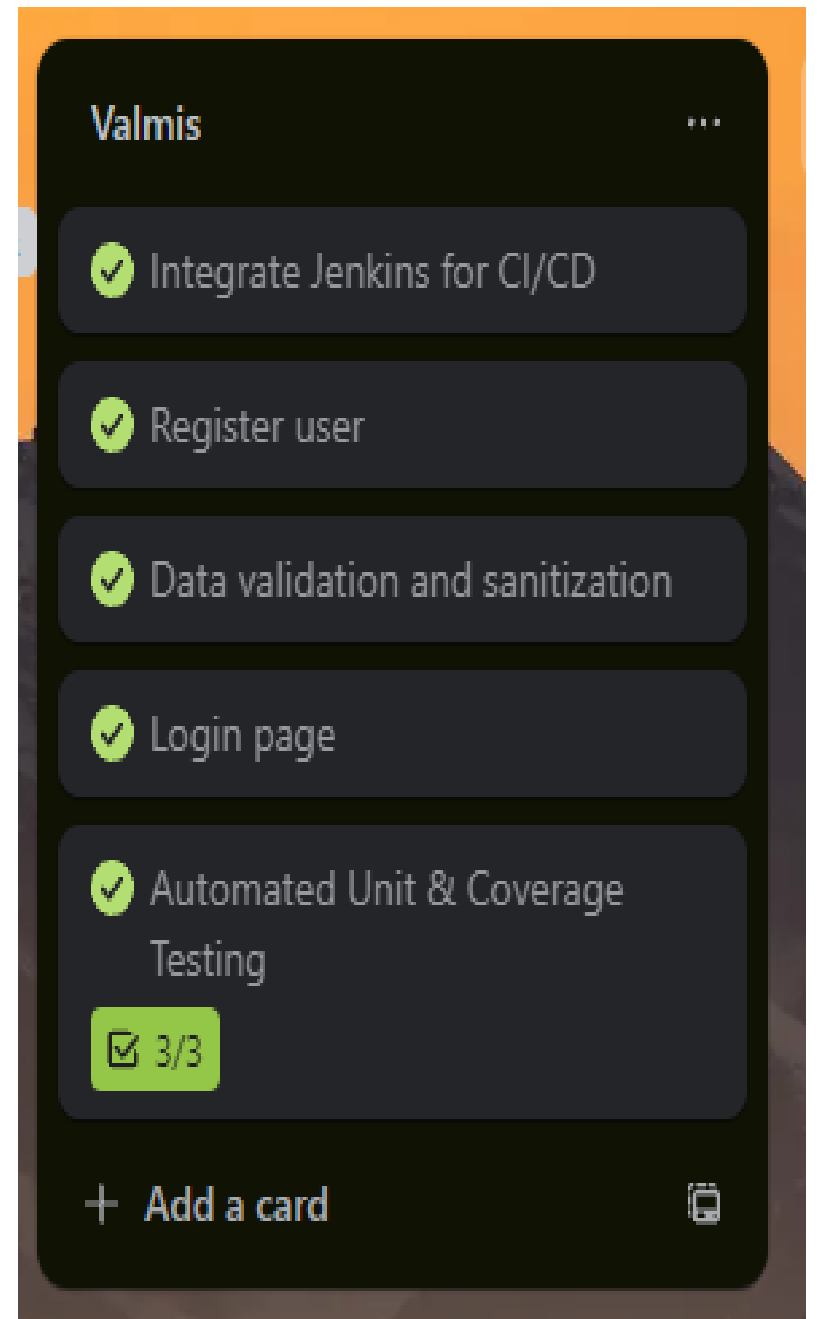- Postpone = Nothing
- Testing

Sprint 2

Sprint Master - Vladimir

The database was implemented, and both unit test coverage and code coverage configuration were successfully set up. The first version of the user interface was developed, and team responsibilities were discussed and defined. Additionally, the project was prepared for the Sprint 2 Review. Maven was also integrated into the project to improve build management and automation.

Sprint 3

- Sprint Master – Patrik
- Sprint Backlog
- Done – Jenkins, User registration and login, bcrypt
- Postpone

- Testing – Junit

Valmis ...

- ✓ Integrate Jenkins for CI/CD
- ✓ Register user
- ✓ Data validation and sanitization
- ✓ Login page
- ✓ Automated Unit & Coverage Testing
  - ☑ 3/3

+ Add a card

Sprint 4

- Sprint Master – Georgii
- Sprint Backlog
- Done – the most part of our objectives
- Release – current version of app
- Postpone - search filters (low priority) , statistics and progress

- Testing – Junit , TestFx for Ui
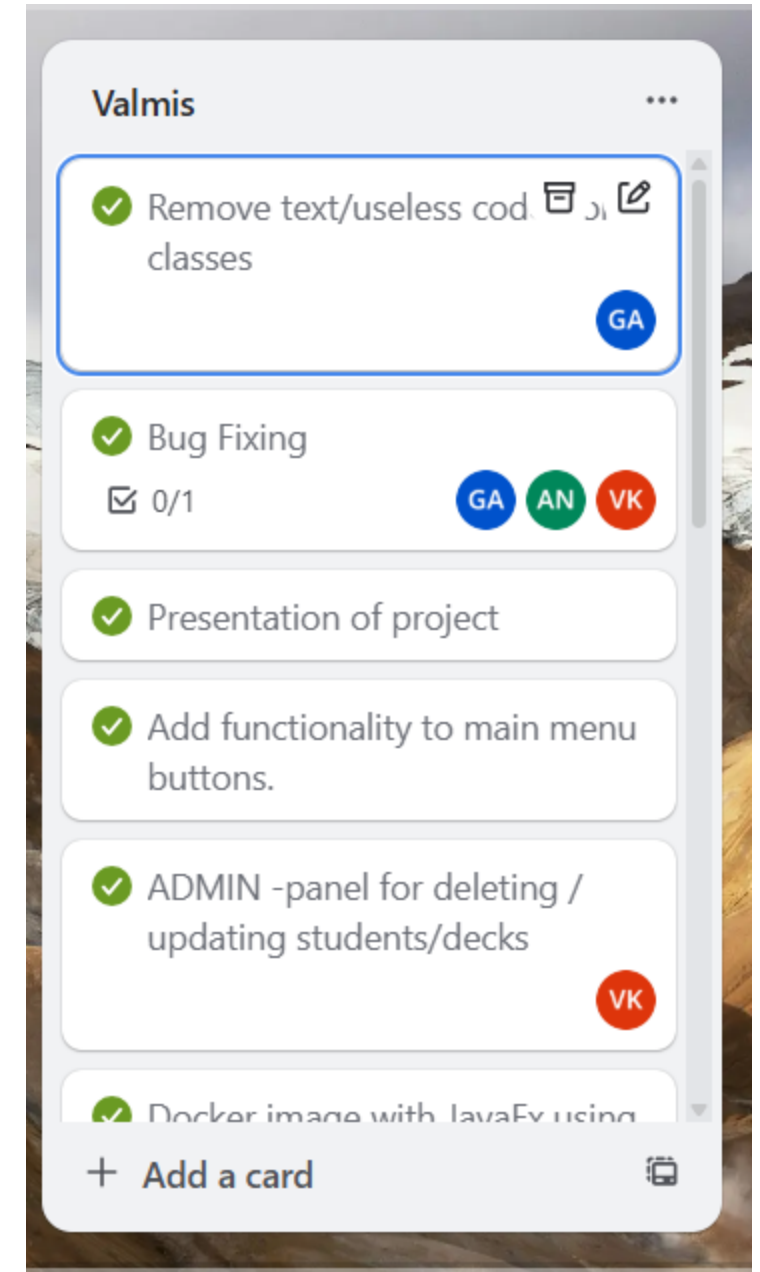
Architectural Design

- Components in Java are used to create modular, reusable, and maintainable pieces of code that can be easily integrated into XML-based configurations. By defining components separately, we can simplify complex structures, improve code organization, and reduce duplication.

- MVC model is used

Applied Technologies

- **Backend:** Java 17+, Maven

- **Frontend/GUI:** JavaFX

- **Database:** MariaDB

- **Testing:** JUnit 5

- **DevOps:** Jenkins, Docker, GitHub, Jacoco

DEMO TIME 🔥 !!!

Learning achievements

- Learning Software Development Processes and methodology

- Using of new technologies – Docker , Jenkins , Pipelines

- Gaining more experience in Java and its tools

- Teamwork

- Testing key features of our app

- Getting mvp version of app (Minimum Viable Product)

Plan for further development

- Adding features and functionality

- Fixing bugs

- Continious testing , checking and development

- Adding localization

- Adding supporting for different platforms

- Adding orcesterization with kubernetes