Qupids

Speed Dating

presented by

Group 6: Ade Aiho, Heta Hartzell, Mika Laakkonen, Jonne Roponen

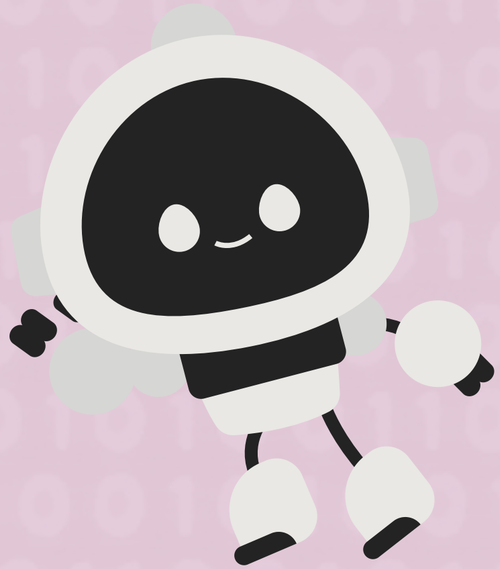
Introduction

In today’s world of online dating, speed dating has become a popular format for individuals seeking connections in person. Manual event management has many challenges like inefficient enrollment, matchmaking, and interest tracking. Without automation, delays, confusion, and missed matches are common. Our initial intent was to provide a solution for an event organizer with these issues.

Our project is a locally run mock implementation of an automated speed dating system. Designed for online hosted events, the system enables participant enrollment, interest selection, match & user management, and match display. This is done in a controlled offline environment.

The project serves as a proof of concept, showing the core functionalities and user interactions planned for the actual implementation. With automation we will create a more efficient and engaging experience for participants and event organizers.

Applied AI Tools



- **Trello**
  Link to board.
- **Github**
  Link to Github repo.
- **AI tools**
  GitHub Copilot, ChatGPT, Claude

- **AI tools**
  AI tools were mainly used to troubleshoot new techonologies, such as Jenkins and Docker, as using them with JavaFx testing frameworks proved to be challenging.

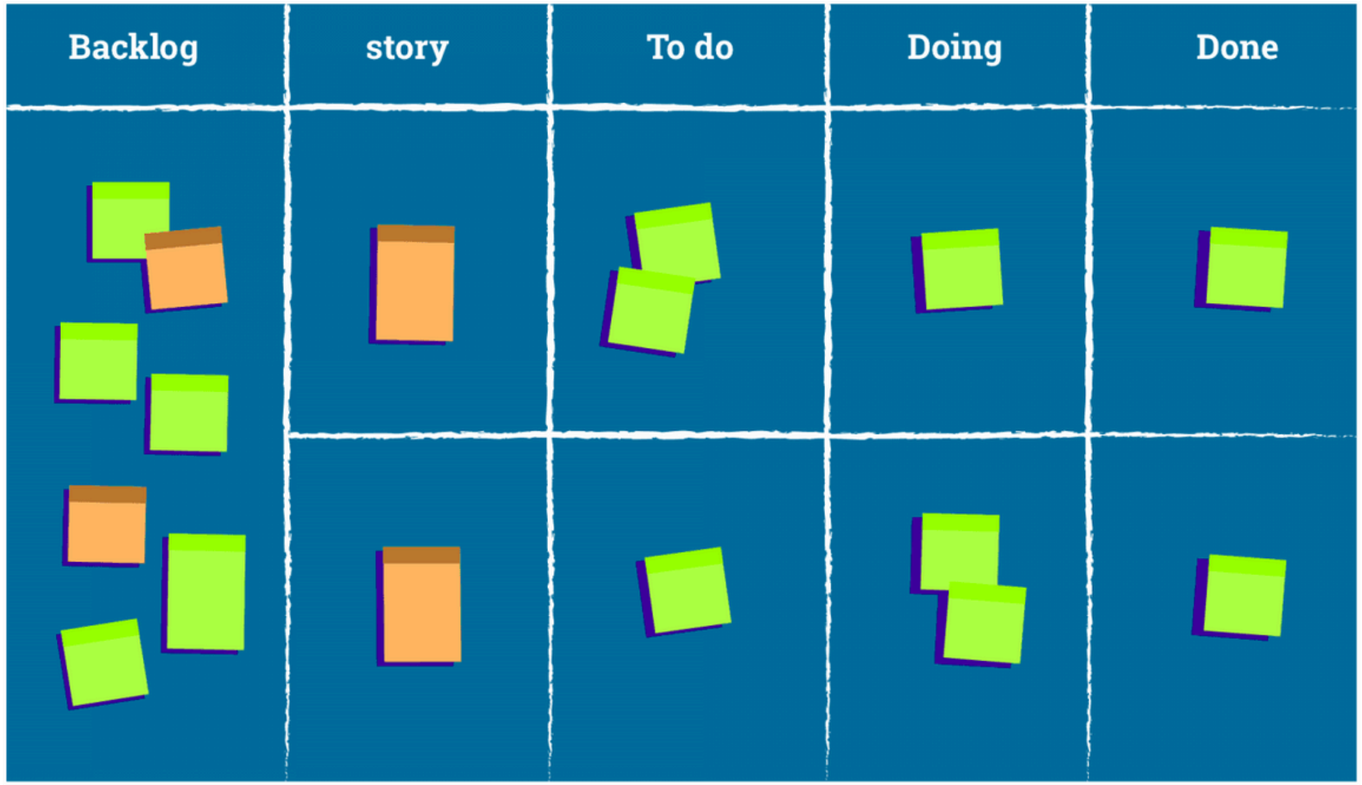
  They were also used to generate boilerplate code, fix bugs, and sometimes provide suggestions when working on a new, inexperienced area for the team.

- **Trello**

  Link to board.

- **Docs**

  Link to document.



Product Backlog

Sprint 1

Scrum Master

Heta Hartzell

Sprint Backlog

15 total tasks (https://trello.com/b/bqUCxp3k/sprint-1)

Done

15 tasks completed

Postpone

0 tasks postponed

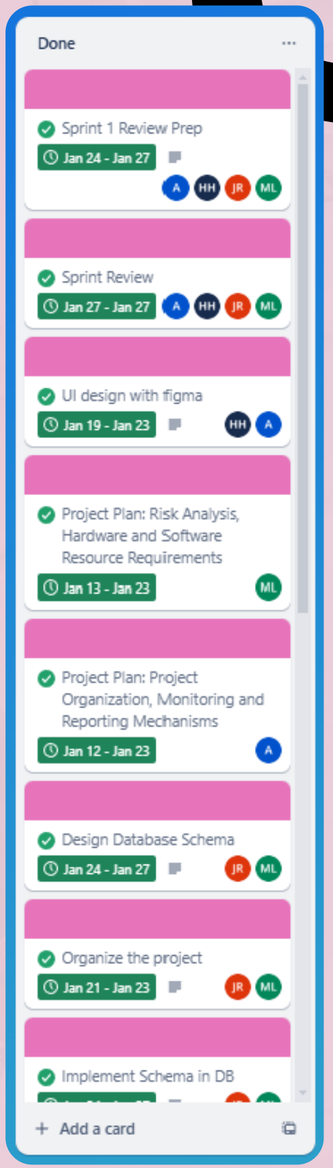
Testing

The mockup was tested using Figma's built-in tools to ensure a smooth user experience.

Release

The product release at the end of Sprint 1 included a clear plan for the next sprints, with a separate product backlog, scheduled tasks for each team member, and story points that reflected the workload fairly.

It also featured a Figma prototype with a tested and planned demonstration to show the application's functionality. The database was created with a schema that matched the project's needs.



Sprint 1

- **Sprint 1 Foundation**
  Core setup, database design, and foundational user interface

- **Project Ideation and Conceptualization**
  - Define project vision and make key user stories and project goals.
- **Organize the project**
  - Organize the project using Trello for task management.
- **Setup Github repository**
  - Create project structure with separate modules for UI, server-side, and backend.
- **Create a prototype**
  - Design a UI prototype for mobile application with Figma.
- **Database Design**
  - Design and create the database schema for:
    - Participants name, session number, phone, email.
    - Sessions session ID, session code, status, timers.
    - Interests participant preferences by number.
    - Matches post-session pairings.
  - Write SQL scripts for database initialization and sample data population.

Sprint 2

Scrum Master

Mika Laakkonen

Sprint Backlog

32 total tasks (https://trello.com/b/EpeuYIIL/sprint-2)

Done

26 tasks completed
7 user stories completed
completed story points 53/121

Postpone

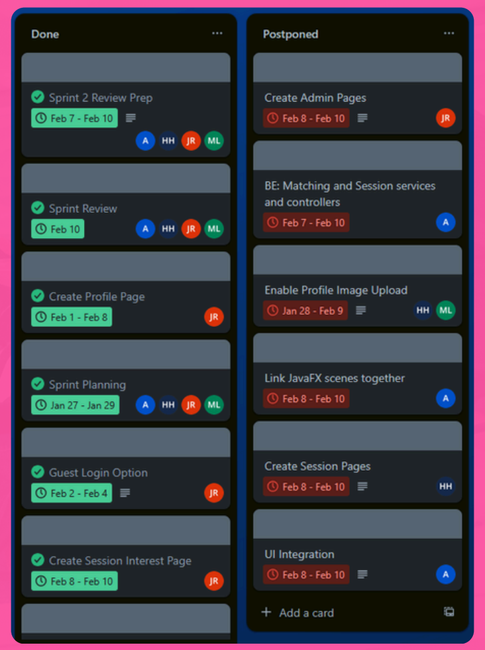
6 tasks postponed

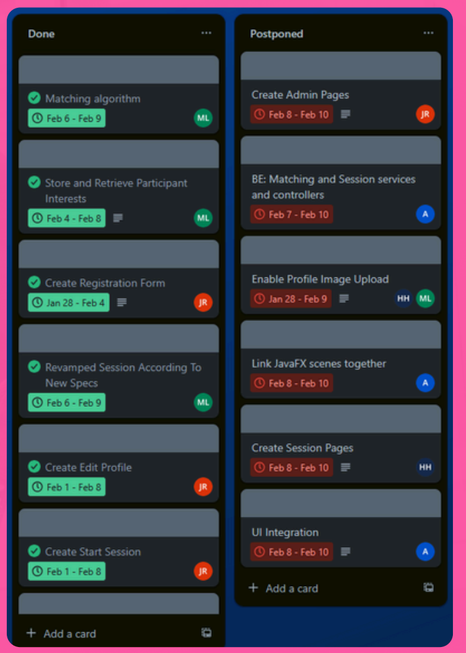
Testing

JUnit tests were conducted to test database connectivity, CRUD functions, and some business logic.

Release

The product release version at the end of sprint 2 had a functioning database, crud operations, and some javaFX components. It still lacked some important functions, but the BE was nearly fully functional and was able to be tested.





Sprint 2

- **Sprint 2 CRUD and Integration**
  Implementing core profile creation, interest selection, and session result display

- **Participant Features**
  - Develop a JavaFX form for participant profile creation name, phone number.
  - Enable profile image randomization from set pool of images.
  - Implement backend logic to store and retrieve participant profiles in the database.
  - Allow participants to modify profile details in the profile view.
- **Match Result Display**
  - Design a results screen that shows a participant's match based on predefined dummy data.
  - Implement the logic to retrieve and display the dummy match data.
- **Interest Management**
  - Develop an interest selection screen where participants choose interests at session start.
  - Implement following selection steps where users refine their preferences.
  - Store and retrieve participant interests in the database.
- **UI Integration**
  - Connect the JavaFX interface to the MariaDB database for profile and interest management.
- **Testing**
  - Write JUnit tests for interest selection and profile CRUD operations.

Sprint 3

Scrum Master

Jonne Roponen

Sprint Backlog

17 total tasks (https://trello.com/b/j538OX3h/sprint-3)

Done

16 tasks completed
7 user stories completed
completed story points 118/121

Postpone

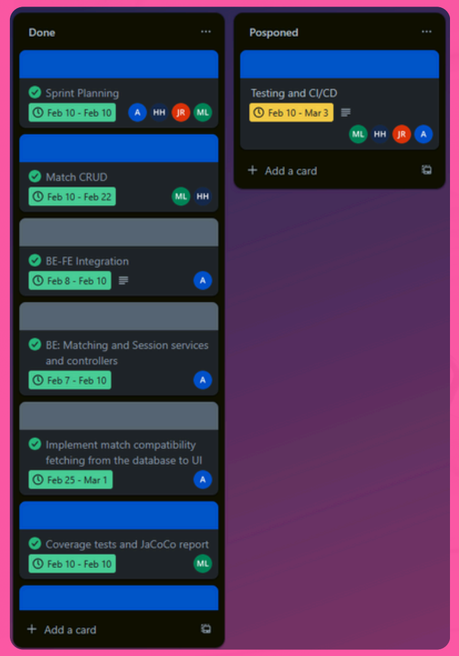
1 task postponed

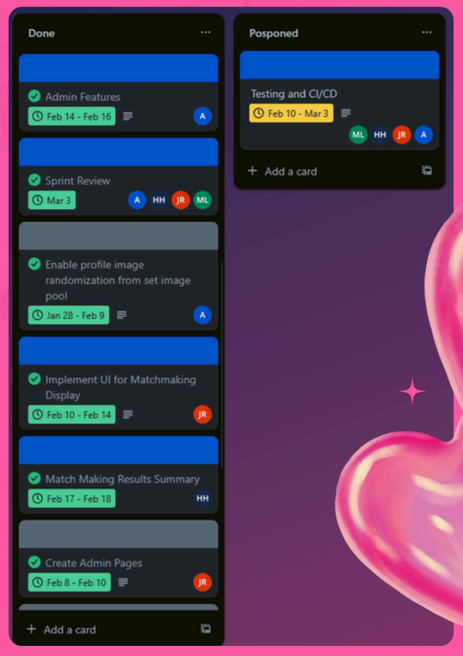
Testing

Unit testing was extended to cover matchmaking and javaFX functionality with Mockito. JaCoCo coverage test was conducted to assess the overall percentage of code tested.

Release

By the end of sprint 3. our product functionality was finished. All the planned features were realized and extensive testing was done to ensure the product worked as intended. Product is ready for deployement.





Sprint 3

- **Sprint 3 Advanced Features**
  Match display refinement, UI/UX improvements, and admin user/match view

- **Post-Session Features**
  - Display matchmaking results page.
  - Display matchmaking summary.
- **UI Enhancements**
  - Improve the overall UI/UX design for better usability.
  - Integrate a help section with instructions on app usage.
- **Connecting BE-FE**
  - Connecting features to communicate the frontend and backend functionality.
- **Admin Features**
  - SQL query to retrieve users and their matches from the database, possibility to remove users.
- **Final Features**
  - Implementing a random rotation of set images for user profiles for a more user friendly experience.
- **Testing and CI/CD**
  - Write JUnit tests for matchmaking.
  - Set up CI/CD pipelines for automated testing and deployment.
  - Coverage test JaCoCo

Sprint 4

Scrum Master

Ade Aiho

Sprint Backlog

9 total tasks (https://trello.com/b/oyO7xKtY/sprint-4)

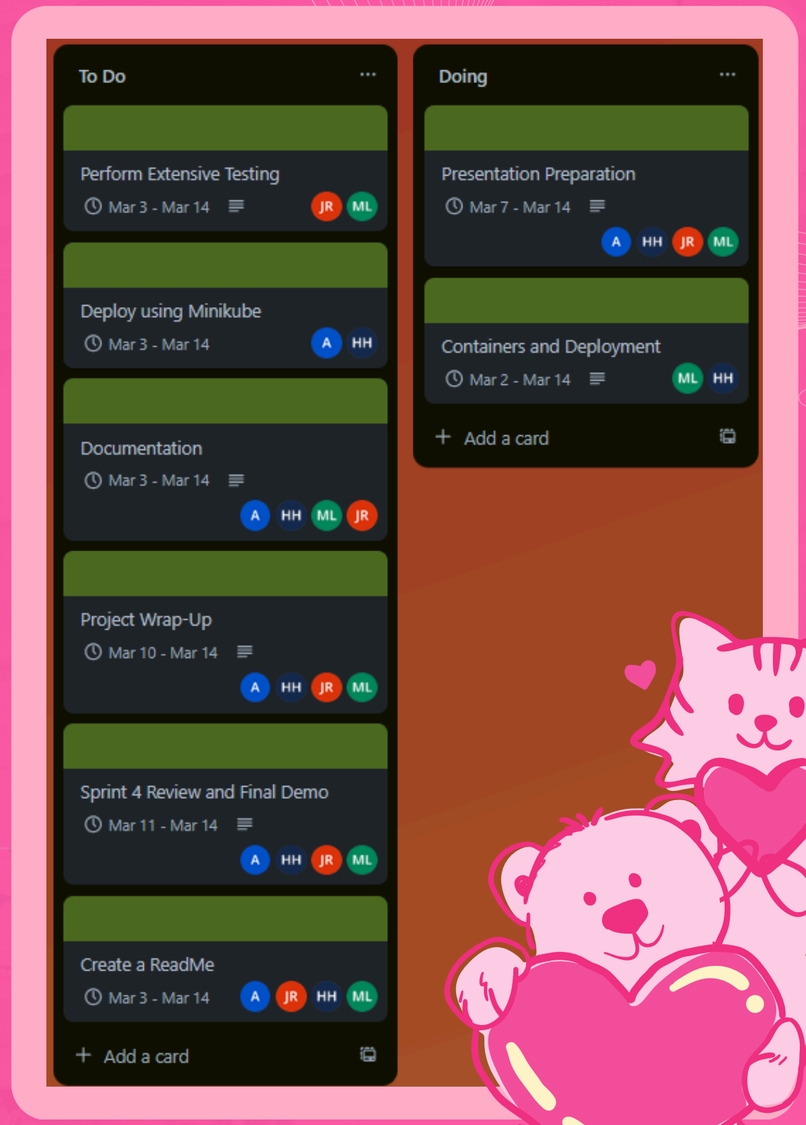
To be implemented

As the sprint is still ongoing, the team has focused the first week of it on preparing the presentation. Moving forward, the tasks to be implemented include:

- Documentation
- Final tests

Done

- Deployment



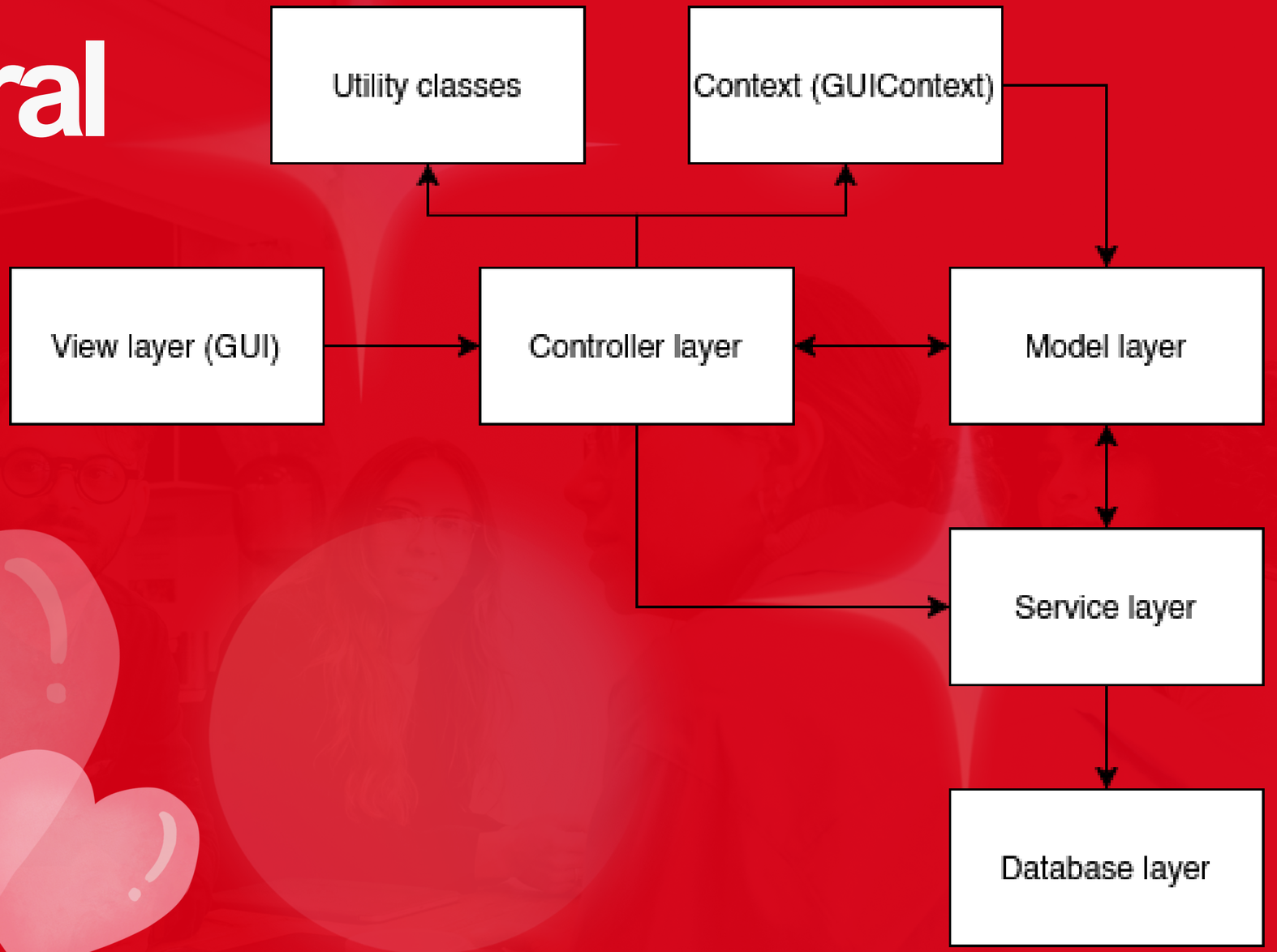
Sprint 4

- **Sprint 4 Final Features, Testing, Final Presentation, and Submission**
  System testing, and deployment

- **Extensive Testing**
  - Perform testing to ensure all features work seamlessly.
  - Write additional JUnit tests for edge cases.
- **Containers and Deployment**
  - Containerize the application using Docker for portability and easy deployment.
  - Test the deployment process in staging and production environments.
- **Presentation Preparation**
  - Create a comprehensive presentation showing all features.
  - Demonstrate how automation simplifies speed dating events.

- **Project Wrap-Up**
  - Submit complete documentation, including database schema, and testing reports.
- **Documentation**
  - Write ReadMe and do system design documentation.

Architectural Design



Applied Technologies

List of Used Technologies

- Java 21
- JavaFX 23.0.2 (GUI)
- Maven (Build tool)
- JUnit 5 (Unit testing)
- Jakarta Persistence API
- Hibernate (JPA ORM)
- VcXsrv
- MariaDB (Database)
- Mockito (Testing framework)
- TestFX (GUI testing)
- JaCoCo (Code coverage)
- Docker
- Jenkins

How the Technologies Were Used

This project uses Java 21 and JavaFX to create the user interface, with Maven for handling dependencies. Hibernate and Jakarta Persistence API manage the database connection to MariaDB, making it easier to store and retrieve data. JUnit 5 and Mockito are used for testing, while TestFX helps test the JavaFX interface. Docker is used to containerize the project, making setup easier, and Jenkins automates building and testing. VcXsv X Server allows users to run graphical applications from a Linux container on their Windows system

Demo

Demo

User Demo

Guest Demo

Admin Demo

Learning Achievements

CI/CD

The group learned how to build Maven applications on Jenkins using both a freestyle project and a pipeline script. The group also learned how to containerize applications on docker and push them to Docker hub, although getting a GUI to run visibly in a container was a struggle.

Testing

The group learned how to create mock objects with Mockito Framework to test certain functions without directly interacting with the database.

AGILE

The group's skills working with AGILE principles improved a lot during the project. All members gained experience working as SCRUM masters, writing sprint reports, and monitoring important SCRUM metrics like sprint velocity.

Plan for further development

Moving to Mobile

- Develop using React Native for Android & iOS.
- Ensure mobile responsiveness with flexible layout design.

Hosting & Deployment

- Frontend: Deploy React app via Vercel.
- Backend: Host Java backend on Render.
- Database: Move to Cloud-hosted or PostgreSQL.

Advanced Admin Dashboard

- Track participant engagement & session statistics.

Session & Organizer Features

- Participants can join sessions using an enrollment code.
- Organizers can create, manage, and track speed dating sessions.
- Session timers and automated progression handling.

Notification System

- SMS/email reminders for upcoming sessions.
- Instant notifications for new matches and session updates.

Two-Factor Authentication

- Secure login using OTP/authenticator apps.
- "Remember me" functionality for trusted devices.

Our Team

ADE AIHO

DEVELOPER

HETA HARTZELL

DEVELOPER

MIKA LAAKKONEN

DEVELOPER

JONNE ROPONEN

DEVELOPER

Thank you!