Project Documentation - Spring Boot Application: Group 18

Talha Doğu Celal Kekeç Oğuzhan Ömer Taşkın

1 Introduction

This document provides an overview of the Java Spring Boot project including the architecture, technologies used, and setup instructions. The project exposes a REST API using OpenAPI specification and uses MariaDB for data persistence.

2 Project Overview

The backend for the game 'Catan'. The project provides fundamental authentication features, specifically registration, login, password resetting, and password encryption. There are two tables in the database. One is for players (users), and the other is for scores. There exists a many-to-one relationship from scores to players. More specifically, a player can have multiple scores, while a score can only belong to a single player.

3 Architecture

Describe the overall architecture and key components:

- Spring Boot - REST API, business logic
- MariaDB - Database
- OpenAPI - API specification
- Deployment architecture (e.g. Docker containers)

Include a diagram of the architecture.

4 Key Technologies

List the main technologies, frameworks, and tools used:

- Spring Boot

- Spring Data JPA
- OpenAPI/Swagger
- MariaDB
- Maven
- Docker

4.1 Software dependencies

The key dependencies used in the project are:

- **Spring Web** - For building RESTful web services. Provides RESTful routing and request handling.
- **Spring Data JPA** - Simplifies data access layer and integration with JPA.
- **MariaDB Driver** - JDBC driver for connecting to the MariaDB database.
- **SpringDoc OpenAPI** - For generating OpenAPI documentation for the REST APIs. Provides Swagger UI.
- **Lombok** - Utility library to reduce boilerplate code like getters/setters.
- **Spring Boot Test** - Helper libraries for testing Spring components.
- **Java Mail** - For sending emails like password reset. Provides utilities for SMTP mail sending.

5 REST API

There are 10 REST API endpoints, each serving a specific purpose:

- **/player:** Used to retrieve a player from the database using Session-key as a parameter.
- **/player/allPlayers:** Used to retrieve all players from the database.
- **/player/register:** Used to register a player to the database if there is no existing player with the same username or email address.
- **/player/login:** Used to log in a player if there exists a registered user with the same username and password. If user logs in successfully, the random session-key for the user is generated and assigned to him.
- **/player/reset-password:** Used when the user wants to change their password. By providing an email registered in the system as input, this function sends an email to the provided email address, and a reset-key is generated and set for the user with the same email address.

- **/player/change-password:** The second step of the password change involves the user checking their email to obtain the reset-key. They then provide this key, along with their new password, to this function. After executing this function, the password is changed, and the new password is set in the database for the user.
- **/score:** Used to save a score to the database by providing the username and score points as input.
- **/score/lastWeek:** Used to retrieve ordered scores from the last 7 days.
- **/score/lastMonth:** Used to retrieve ordered scores from the last 30 days.
- **/score/allTime:** Used to retrieve ordered scores from all time..

6 Database Design

The database schema can be outlined to show entities like:

- User
- Score

7 Testing

There are a total of 29 unit tests to assess the following functionalities:

- Overridden equalTo functions for the entities Player and Score.
- Testing JPA functions, including finding a Player by username, email, session-key, reset-key, and checking existence using username and email. Additionally, finding Scores after a specific date and checking the order of the scores are also tested.
- Testing service methods, including registration, login, email sending, password changing. Additionally, obtaining weekly, monthly and all times scoreboard with proper ordering is also tested.
- RestAPI tests that can be executed manually using SwaggerUI.