

Rock-Scissors-Paper Exercise

Ciklum Challenge

Amine Kerzazi Seddini

13/09/21

Index

1.Introduction.....	3
1.1Overview.....	3
1.2The application we are building.....	3
1.3Used technologies.....	4
1.4Steps.....	4
2.Backend project.....	4
3.Class and sequence diagrams.....	7
4. Testing.....	8
5.Frontend.....	9
6.Steps to run the application locally.....	11

1. Introduction

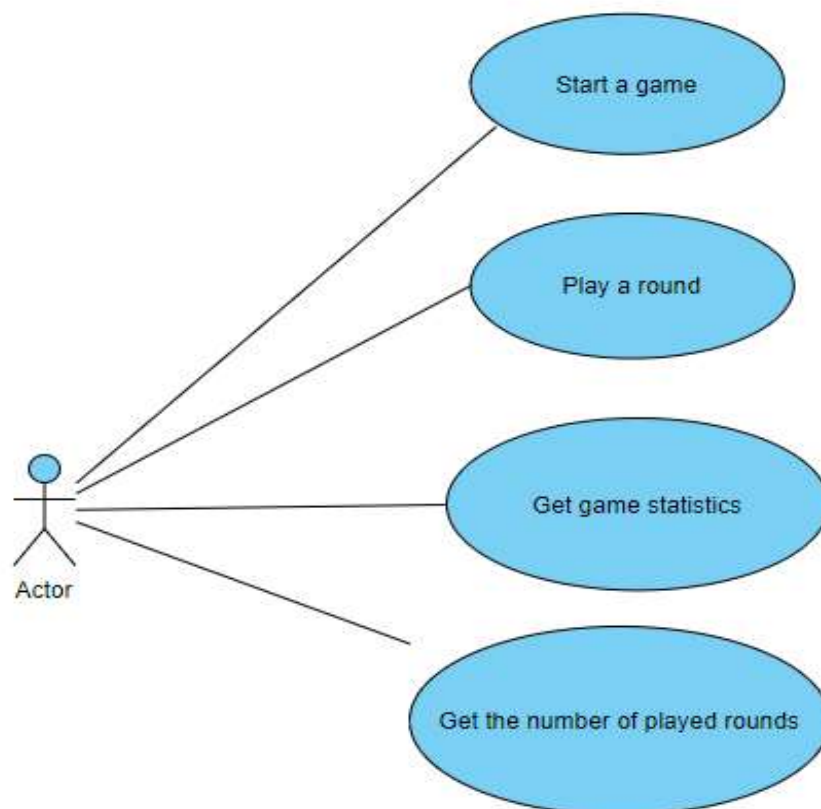
1.1 Overview

This document is a guide on how we developed the application required for the Rock-Scissors-Paper exercise.

1.2 The application we are building

The application we are building is a web application that provides a basic GUI for a human to play several rounds of the game rock-scissors-paper. The application will allow the user to start the game, play rounds and get game statistics.

Find below a use case diagram illustrating the use cases we will implement.



1.3 Used technologies

For the development of this application, we made use of the following technologies:

- Maven 3.8.1
- Java SDK 1.8
- Groovy 2.4.21
- Spring framework 5.3.9
- Angular: Angular CLI: 12.2.0, Node: 14.17.4, Package Manager: npm 6.14.14
- IDE: We developed the project under IntelliJ IDE for the backend part and with Visual Studio Code for Frontend part.

1.4 Steps

To develop this application, we follow these steps:

- Setting up the project for the backend part
- Adding Spring Maven dependencies
- Creation of the package structure
- Implementing of Units tests
- Implementation of the business logic (the model, the controller, the service)
- Testing the API Rest with postman.
- Setting up the project for the Frontend part
 - Implement the components, the model, the service.
 - Connect the Angular app to our API rest
- Running both part (BE and FE) all together and start user testing.

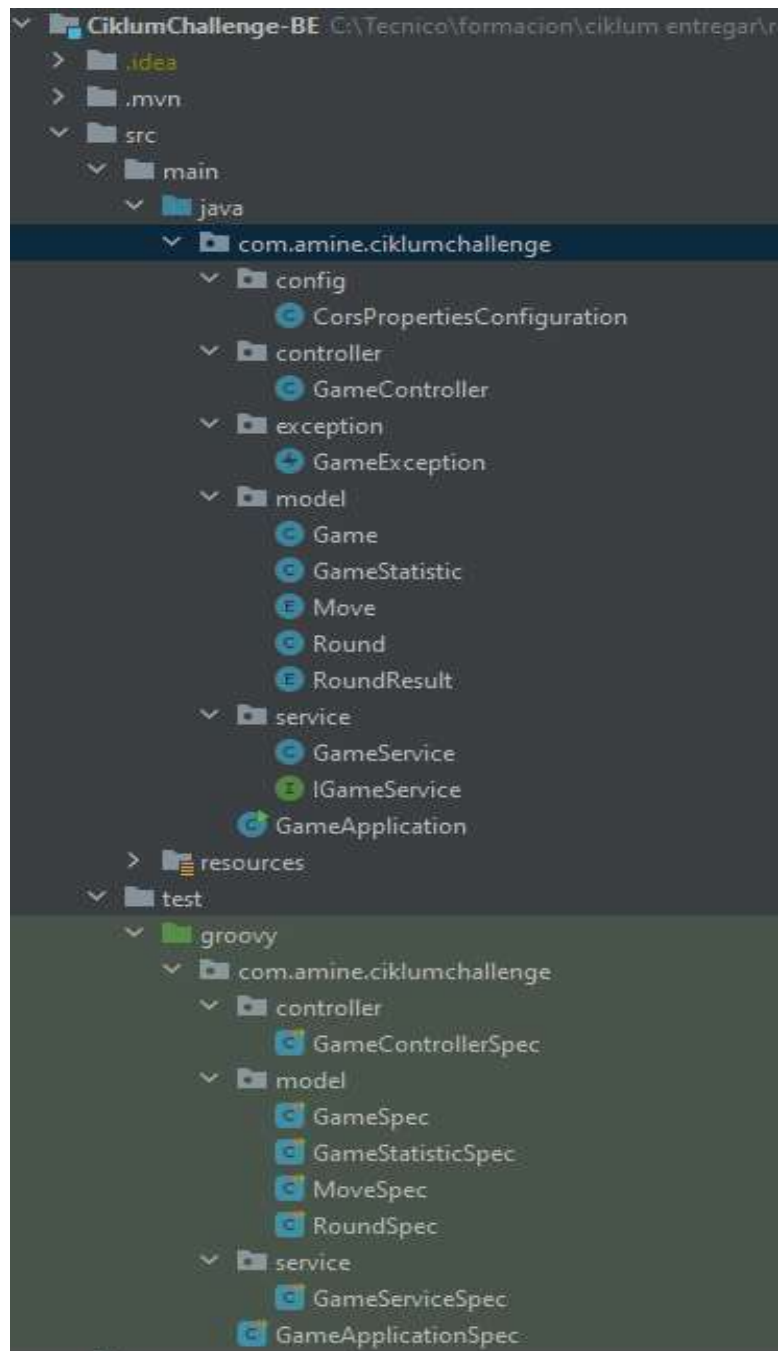
2. Backend project

To set up the project, we create a Maven project and we create the following package structure:

- **com.amine.ciklumchallenge.controller:** This package will contain Spring Controller classes for the game application.
- **com.amine.ciklumchallenge.model:** This package will contain the object model for the game application. The model will be a simple POJO class with needed methods and attributes.
- **com.amine.ciklumchallenge.service:** This package will contain the code which implements the behaviour of the game.
- **com.amine.ciklumchallenge.exception:** This package will contain our specific defined exceptions.
- **com.amine.ciklumchallenge.config:** This package will contain the needed classes for game configuration.

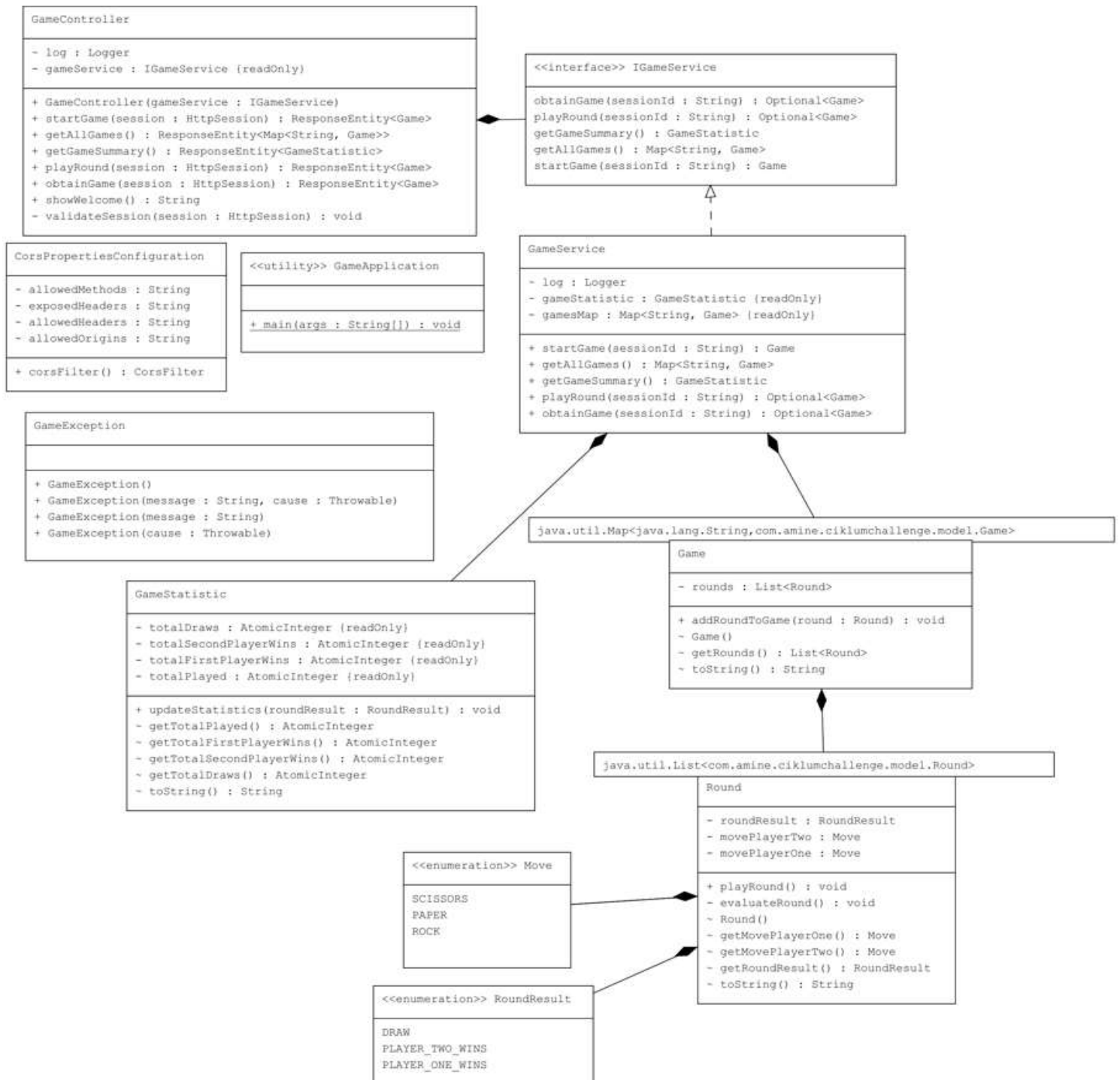
We create the same structure under `src/main/test/groovy` folder to implement the corresponding tests.

Then, we had a Maven-based project, which looks like:

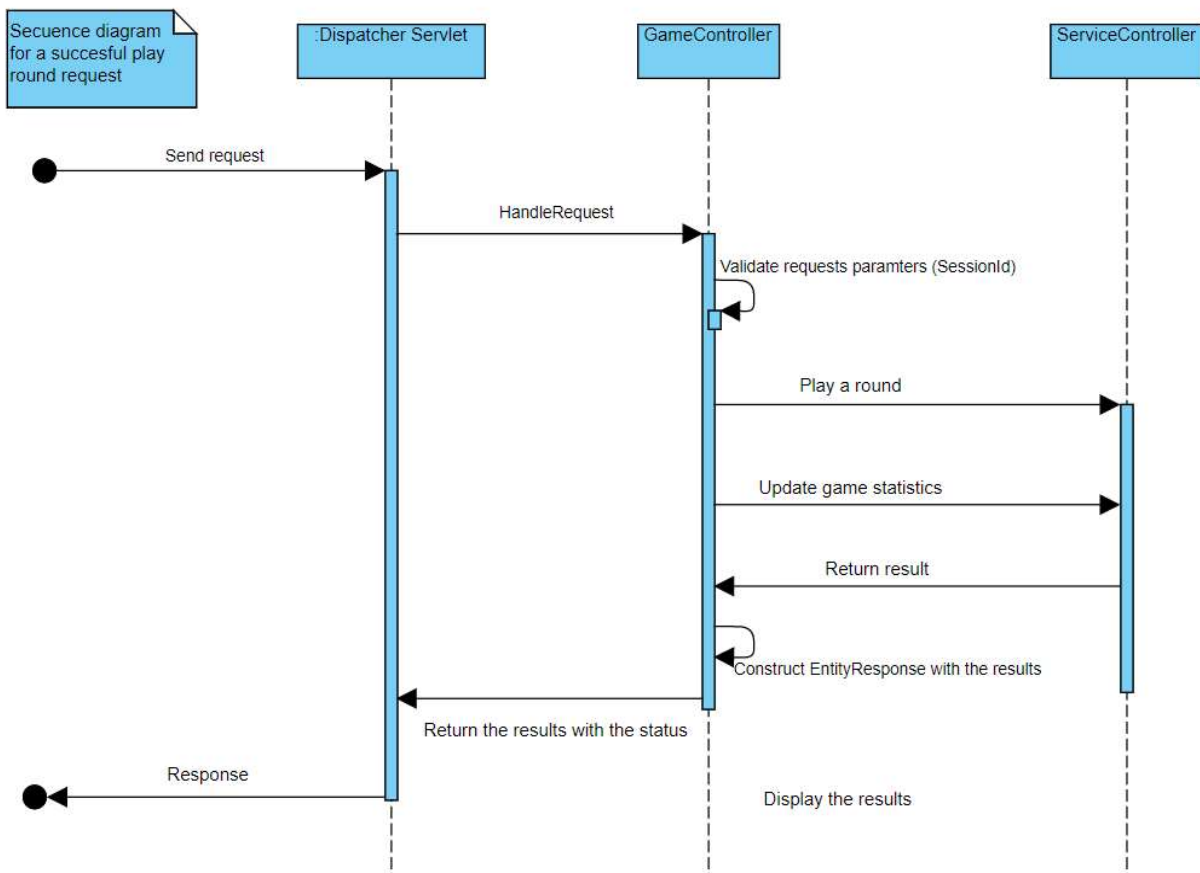


3. Class and sequence diagrams

Here is a class diagram to highlight the existing relations between the classes already shown in section2' picture:



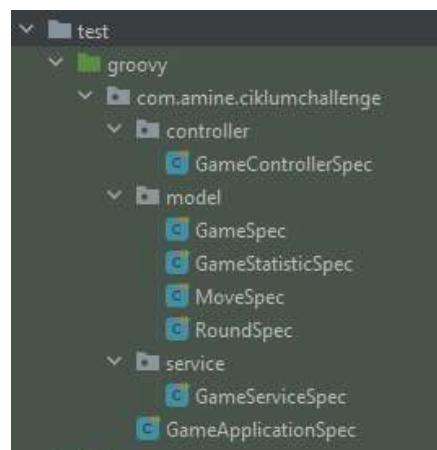
Here's a simplified sequence diagram of how our system can work .



4. Testing

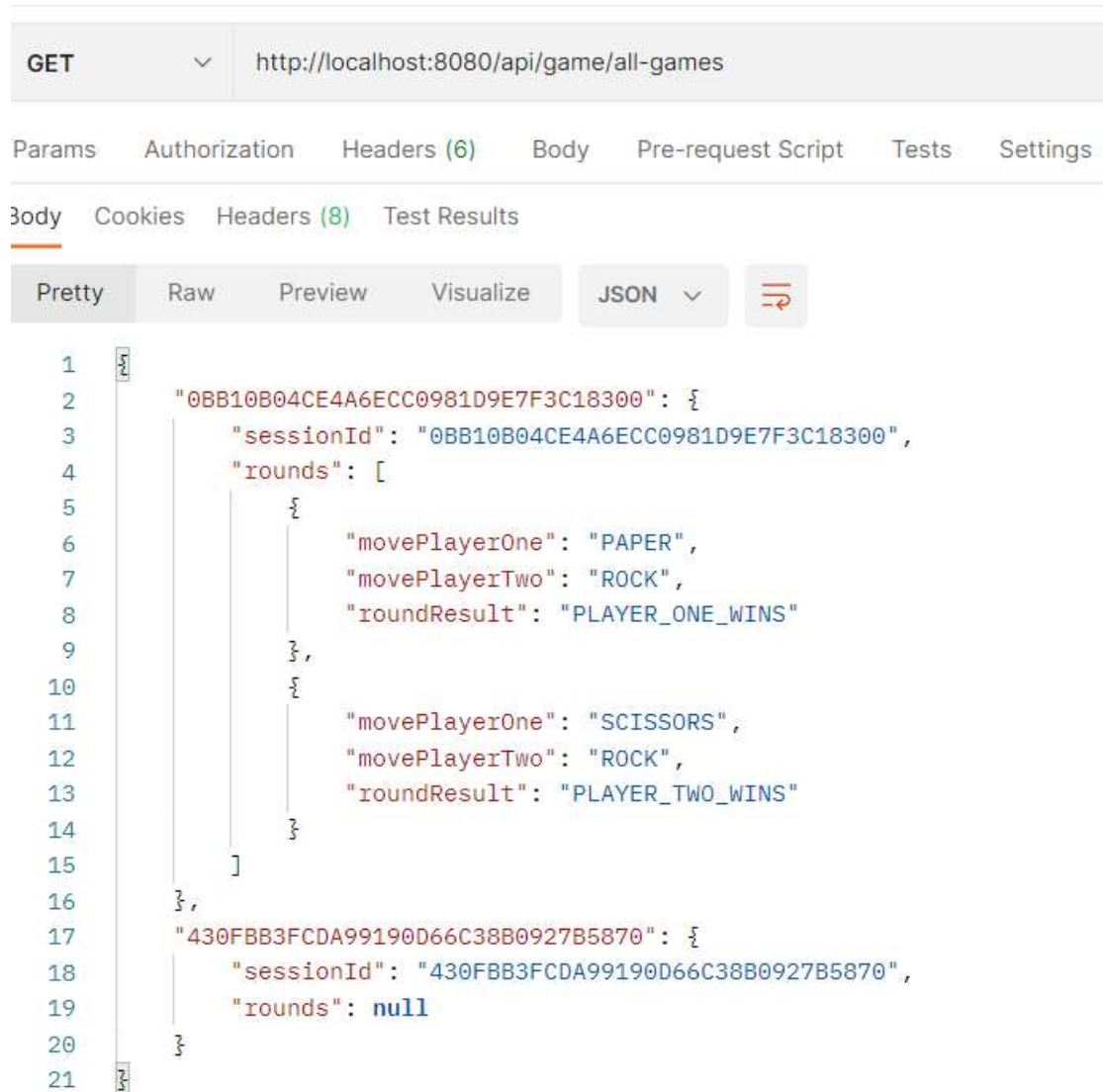
The application had been tested by:

- **Unit tests** for all the clases previosuly implemented using groovy.



- **Postman** checking the answer for each one of the methods like the one shown in the picture below

<http://localhost:8080/all-games>



5. Frontend

The frontend part had been implemented using Angular.



As requested by the exercise, we have two views: (**/game-playing** and **/game-summary**)

- **/game-playing:** This view is the responsible to start a game, play round and show the total of played rounds for the current game. This view looks like:

Ciklum Challenge App

Game playing
Game summary

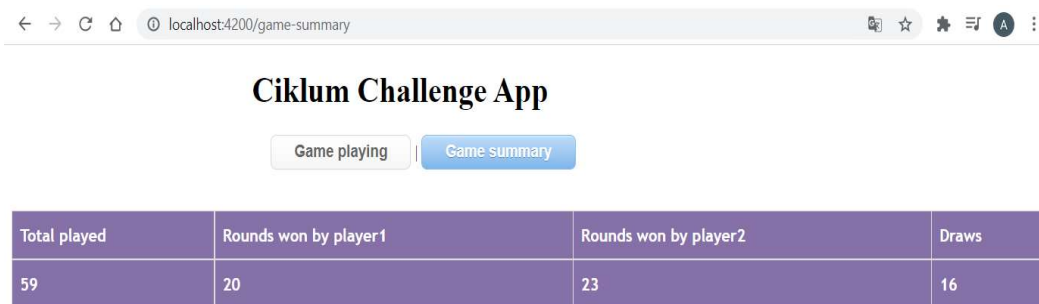
Use the buttons to restart a game or to play a round

Restart game
Play a round

Number of played rounds:5

Round	Player1 move	Player2 move	Round result
Round_1	PAPER	ROCK	PLAYER_ONE_WINS
Round_2	SCISSORS	ROCK	PLAYER_TWO_WINS
Round_3	SCISSORS	ROCK	PLAYER_TWO_WINS
Round_4	PAPER	ROCK	PLAYER_ONE_WINS
Round_5	ROCK	ROCK	DRAW

- **/game-summary:** This view is the responsible of showing the global games summary. This view looks like:



Total played	Rounds won by player1	Rounds won by player2	Draws
59	20	23	16

6. Steps to run the application locally

In this case we have two parts:

- Backend part:
 - if using an Ide (Eclipse or IntelliJ), just run the **GameApplication** class located in **com.amine.ciklumchallenge.controller**
 - If you prefer launch it without opening any Ide, just run the comand **mvnw spring-boot:run** from a comand line (from the project folder).
- Frontend part: As said in the exercise "Keep in mind this is a BE position, so good FE skills are good to have but not a requirement. (It just need to work)"
 - If you already have Angular installed, you only need to run the commands:
 - **ng update @angular/cli @angular/core** to perform a basic update to the current stable release of the core framework and CLI.
 - **npm update** from a comand line (from the project folder).
 - **ng serve** from a comand line (from the project folder).
 - If Angular is not installed, you should proceed to install it following these steps:
 - Install Node.js
 - Install Angular CLI (npm install -g @angular/cli)
 - Check if it's already installed (ng -version)
 - Run the command npm update from the project folder.
 - launch the Frontend project with the command (ng serve) from the project folder.

- Running

Once both parts (backend and frontend) are running, open a browser with <http://localhost:4200> and start playing the rock-scissors-paper game.

Notes: Just to take into consideration:

- I've used the following versions for Frontend part: Angular CLI: 12.2.0, Node: 14.17.4, Package Manager: npm 6.14.14. If you run an old version, it is possible that you need to update to a newer one. You can perform a basic update to the current stable release of the core framework and CLI using the command: **ng update @angular/cli @angular/core**
- Angular project is launched by default in <http://localhost:4200>
- Angular project is considering that Backend project will be launched in <http://localhost:8080>. If backend is launched in other url/port, please don't forget to change the url in the file `environment.ts` located in `project_path/src/environments`
- Backend project is expecting the angular project to be launched in <http://localhost:4200>. The backend project is configured to allow only requests coming from that url. If you prefer to launch the Frontend from other url, please don't forget to add the url to the configuration file `application.properties` to the variable `cors.allowed.origins` (currently it is set like this: `cors.allowed.origins=http://localhost:4200`)