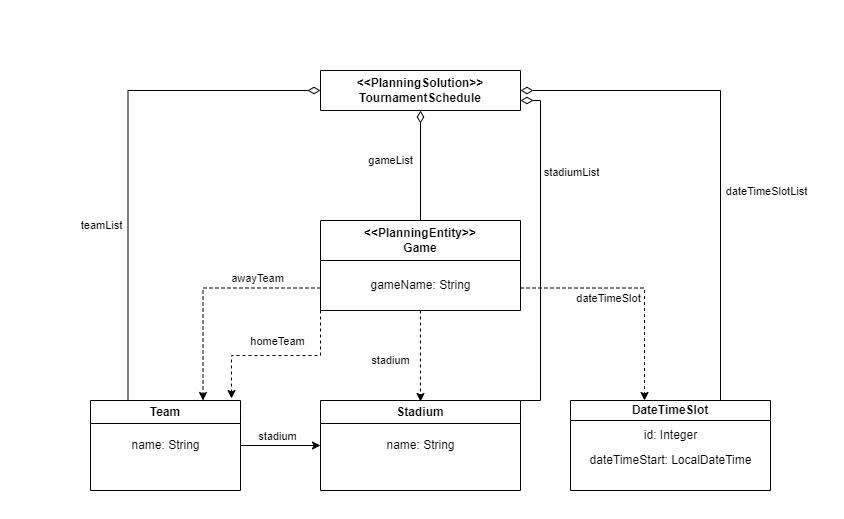
Aleksejs Jeļisejevs, aj17052, 21.05.2023.

**Task:**

Create an optimization service, that makes games calendar for football tournament. As input there is a list of stadiums, list of teams with their names and stadium name, where they play home games (one stadium can belong to many teams), and list of available Date/time slots. Every team pair plays against each other twice: once as home team, once as away team. Games are always played on home team stadium.

**Domain model:**

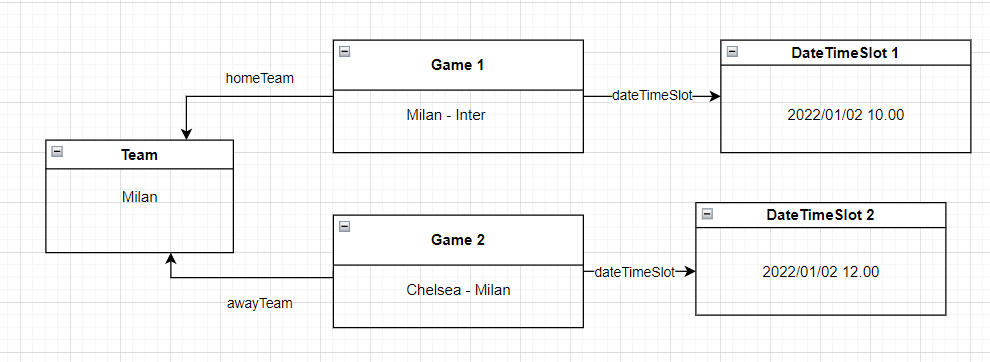


At the very beginning, in input .json file we provide list of stadiums, list of teams and corresponding stadium, all the available time slots, and we specify for every game home team, away team, stadium (equal to home team stadium) and dateTimeSlotId = 1 (it is work around to serialization bug).

**Surrounding function**: Default function, provided by OptaPlanner

**Cost functions:**

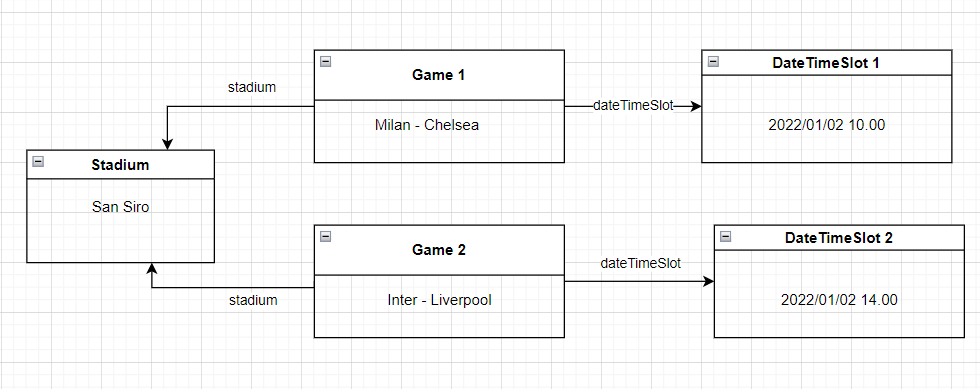
Hard – No more than 1 game per day for each team. Cost = 1 hard.



*Game1.homeTeam == Game2.awayTeam &&*

*Game1.dateTimeSlot.getDate() == Game2.dateTimeSlot.getDate()*

Hard – No more than 1 game per day on each stadium. Cost = 1 hard.



*Game1.stadium == Game2.stadium &&*

*Game1.dateTimeSlot.getDate() == Game2.dateTimeSlot.getDate()*

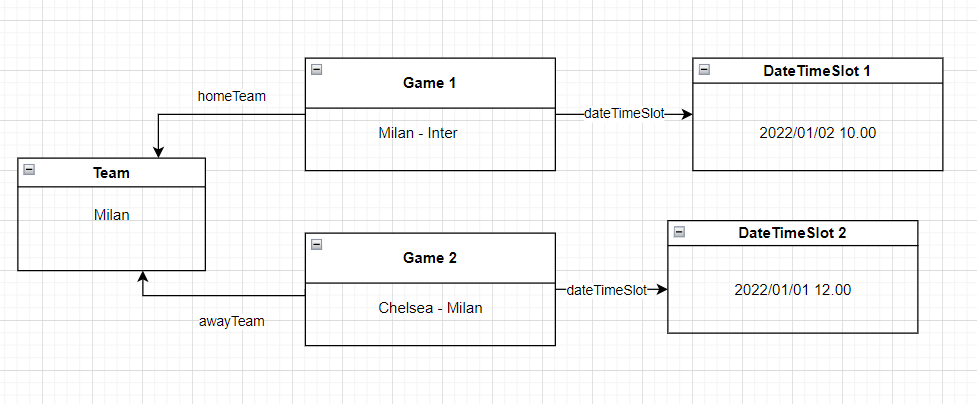
Soft – Games should be scheduled on weekend and Fridays. Cost = 2 soft.

Diagram

Description automatically generated

*Game1.dateTimeSlot.getDayOfWeek not in [“Friday”, “Saturday”, “Sunday”]*

Soft – Minimum 2 free days between games for each team. Cost = 2 soft.



*Game1.homeTeam == Game2.awayTeam &&*

*Abs(Game1.dateTimeSlot.getDate() - Game2.dateTimeSlot.getDate()).DAYS < 3*

Soft – On workdays games starts not earlier than 18.00. Cost = 1 soft for each hour before 18.00.

Diagram

Description automatically generated

*Game1.dateTimeSlot.getDayOfWeek not in [“Saturday”, “Sunday”]*

*&& Game1.dateTimeSlot.getHour() < 18*

Soft – On weekend games starts not earlier than 12.00. Cost = 1 soft for each hour before 12.00.

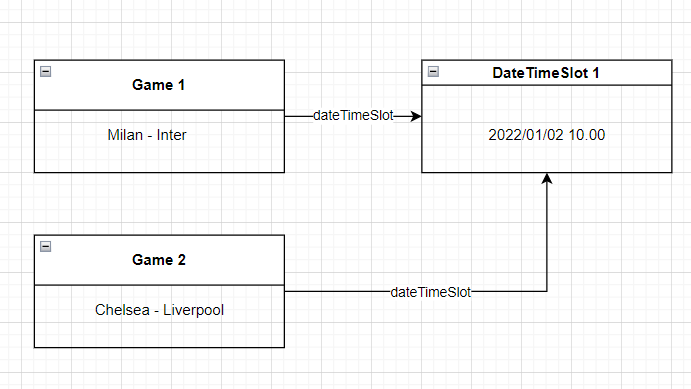
Diagram

Description automatically generated

*Game1.dateTimeSlot.getDayOfWeek in [“Saturday”, “Sunday”]*

*&& Game1.dateTimeSlot.getHour() < 12*

Hard - No more than one game for each time slot. Cost = 1 hard.



Game1.dateTimeSlot == Game2.dateTimeSlot

**Architecture of solution:**  
  
Solution uses all the same technologies and architectures as were shown during lectures. That is JAVA with Maven, Optaplanner and SpringBoot. Thymeleaf was used for UI.

Problems in JSON format can be uploaded using Postman ‘/optimizer/solve2’ request.

Then solutions appear in UI using ‘/optimizer/list’ path.

**Testing:**

I’ve created 4 JSON files with problems of different size. Problem generator’s code is also provided in App.java class. Short description of testing data:

problem.json:

4 teams, 3 stadiums, 14 days for tournament. Every day has time slots on even hours starting from 8.00 till 22.00. Teams Inter and Milan share same stadium.

Problem1.json:

6 teams, 5 stadiums, 28 days for tournament. Every day has time slots on even hours starting from 8.00 till 22.00. Teams Inter and Milan share same stadium.

problem.json:

8 teams, 7 stadiums, 61 days for tournament. Every day has time slots on even hours starting from 8.00 till 22.00. Teams Inter and Milan share same stadium.

problem.json:

8 teams, 7 stadiums, 92 days for tournament. Every day has time slots on even hours starting from 8.00 till 22.00. Teams Inter and Milan share same stadium.

Optaplanner benchmarking was used for testing. TABU\_SEARCH shown the best performance and because of that, in solverConfig I choose it as localSearchType. I also set-up

unimprovedSecondsSpentLimit = 30. Results of last benchmarking are stored in local>benchmarkingReport>2023-01-21\_112913.

**GitHub link:** <https://github.com/SmallUnicorn/TournamentScheduler>

**DockerHub link**: <https://hub.docker.com/r/smallunicorn/tournament_scheduler>