1 Backend & full-stack developers Exercise : TheChampion

Who is TheChampion ?

As an initiative a tennis-table league has to be design for team members inside the digital factory, allowing all staff to participate and randomly manages the matches of all participants and compose a league of 3 matches a day and maximum 12 participants, consuming a modern API for basic features. This is planned to be API first approach, of well-tested functions and enabling agility of later modifications.

Technology constraints

- At the digital factory, we currently have a microservices architecture with services mostly written in Spring Boot.
- We would like you to develop a single service using Spring Boot with either Java or Kotlin that exposes several REST endpoints (see next page)
- You can use whatever database and data access method; the application should create the required database structure.
- You should include a README file that has instructions for us to get the solution running on our machines.

What are we looking to test?

- The overall software architecture of the application
- The structure and **quality of the code itself**
- The use of well-known patterns for REST and Spring development
- Your ability to model the problem domain (data models and APIs)
- **Full-Stack developers/leads (only),** overall Angular project structure and code-quality.
- Bonus points if you include unit tests in your solution.

What are we *not* looking to test?

- We don’t expect you to implement authentication or authorization

External Resources

- Building a RESTful service in Spring Boot
- Testing in Spring Boot

2 Backend & full-stack developers Exercise : TheChampion

Your APIs should provide the following functionality to downstream consumers of the API (in a **RESTful** way):

Participants

- Submit a participant request
- Get a list of all participants
- Group randomly participants into (n) groups

Matches

- Get list of all automatically created the first-round matches
- Update match winner and results
- Close round

The League | BONUS

- Submit a request of new match (players and time)
- Submit league champion

TIP #1: focus on defining a very clear data model. You should include the SQL create table statements etc. in your code base.

TIP #2: invest time in building some 'seed data' that covers all the use cases you want to demonstrate.

ONLY FULL-STACK DEVELOPER/LEADER

TIP #3: build web frontend pages, for some of the journeys as APIs described using Angular.

BONUS POINTS:

Create a new model for 'Congratulation Mail' to be sent to the champion.