

Developer Test Task

Junior Java developer, winter 2018-19

And the challenge begins!

Make sure you follow the assignment and the tech guide that follows. Everything you do must be documented to make the scoring easier (for example comments.txt). Solve as much as you can – a partial solution is better than no solution! In addition to making your code work, make it pretty as well. Feel free to even improve existing code.

Good luck! 😊





1. The backstory

Once upon a time, a group of good friends started a cake club. Though baking was fun, the club had trouble finding suitable times for meetings that would suit everyone. They decided to step into the 21st century and create a webapp that could be used on the go and would take into account everyone's swamped schedules. This is where one of the friends – the only one with coding experience – decided to take up the challenge. Sadly, he was unable to get the whole thing done and stumbled into development hell. Though the architecture set, he had a really hard time completing the different features critical for the cake club's needs. Can you help him and other cake clubbers out?

2. The assignment

2.1. Preface

The application is based on managing two types of calendars, the shared cake club calendar and the personal calendar. The shared calendar includes both shared events and personal ones. All calendar events should have life cycle, in which they can be created, updated, read and deleted. Your goal is to improve the application and implement new features. The main goal is to have an application that helps the cake club plan regular meeting to fit its members' everyday schedules.

2.2. The assignment

First, take a look around in the application and code. Once familiarized, could you please help them build an actually useful webapp? Make sure to look for clues in the code to get more information about where things went wrong and where the previous developer got stuck. The members of cake club are extremely grateful for any kind of help you can provide.

These are the main requirements for making the cake club webapp usable:

1. **Update the Profile page so the user can also save and update skill levels: speed, strength, stamina and cooking.**
 - 1.1. Skill levels are numeric (0-100) and cannot be negative.



- 1.2. Every user has 100 points that can be distributed among different skills.
- 1.3. Make sure to show how many points have been used in total. (Example: speed, 10; strength, 25; stamina, 30; and cooking, 15. Combined: 80).
- 1.4. Make sure to validate your request (i.e., user cannot save combined points of over 100).
- 1.5. Don't forget about data validation and error handling.
- 1.6. Bonus: Sliders are preferred over number inputs.
2. **The members of cake club would like to choose their own profile pictures.**
 - 2.1. User can choose from among four profile pictures (located in webapp/content/images).
 - 2.2. On "Save" image, location is saved into database.
3. **The members of cake club cannot currently filter the group calendar by participation count.**
 - 3.1. The filter should be placed above the calendar.
 - 3.2. Should be a numeric field, where the user can fill in the "number of participants" field.
 - 3.3. When "Search" is clicked, all events with that given number of participants should be displayed in the calendar.
 - 3.4. Bonus: Add a multi-select filter, for querying by selected users.
4. **The most important feature for the users is the ability to create a group event.**
 - 4.1. In the group calendar, implement the "Add event" button, which takes the user to a new add-event view. (Hint: Create this component using: ng generate component.)
 - 4.2. The view contains a form with the following fields: event title, description and date period (period is the minimum and maximum date range where the event has to take place).
 - 4.3. When "Save" is clicked in the form:
 - 4.3.1. The *perfect date** is calculated in the given period.
**Perfect date* – a date where every user can participate or when the highest number of users can participate.
 - 4.3.2. If the perfect date is found, one event is created and fields saved into jhi_calendar_event and event participants are saved into jhi_user_event.
 - 4.3.3. If the perfect event is not found (i.e., no participants in the given time period), an error is shown.
 - 4.4. Bonus: Perfect date calculation event and "Save" event are separated into two steps.
 - 4.4.1. First step queries the all the suitable dates, for the user to choose.
 - 4.4.2. Second step saves the event using the suitable date picked by the user.
 - 4.5. Don't forget about data validation and error handling.



5. Users can delete an event.
 - 5.1. "Delete" button is added to event view and delete feature implemented.
 - 5.1.1. Make sure that only the creator of the event can delete it.
 - 5.2. Bonus: Logical delete is used over physical one and related queries modified.
6. Update Angular to the latest version.
 - 6.1. Check if the Angular used in the project is the latest, if not update it.

3. Set up

All you need is Java 8 installed and some IDE available to simplify coding. Gradle is used as the build system and Gradle Wrapper is provided. This means you don't even have to install Gradle and can run all commands with either 'gradlew.bat' or 'gradlew' bash script.

Try running 'gradlew.bat tasks' or './gradlew tasks' in the current directory from your command line.


4. Wrap up

Once you have completed your development, run 'gradlew.bat createSubmissionZip' in the root directory, where the current README.pdf resides. Your results will be packed to assignment_submit.zip under build directory. This is the file you need to upload as well.

5. Tech guide

5.1. Gradle


Remember that Gradle commands can be executed in several ways:

1. When you have Gradle installed globally: `gradle <command>` 
2. When working on Windows command line or PowerShell: `gradlew.bat <command>`
3. When working on MacOS, Linux or Windows 10 Shell: `./gradlew <command>` In the following sections we use in commands **gradle**, but you can any of the previously mentioned above that suit your system (we recommend `./gradlew` though) .



5.2. Building the project

This application will consist of multiple modules. To build the application:

1. Install node.js (latest) from <https://nodejs.org/en/download/> 
 - 1.1. Node version: ≤ 8.9
 - 1.2. Npm version: $\leq 6.4.1$
2. Go to your terminal and execute a few commands in the assignment directory:
`npm -v`
3. Install project dependencies (might show some warnings, that's OK): `npm install`
4. Gradle installs project dependencies (might show some warnings, that's OK):
`gradle build`

5.3. Starting the application

5.3.1. Backend

To start the backend part of the application in development mode:

1. Gradle will start the application (command will block for the duration of the application run): `gradle bootRun`
2. Wait for confirmation message indicating the application has been successfully started.

5.3.2. Frontend

To start the frontend part of the application in development mode:

1. Node will start the application (command will block for the duration of the application run): `npm start`
2. After the last command, the application should open in your browser (or check the console).

5.3.3. Database

Database layer management:

1. To view the database, go to <http://localhost:8080/h2-console/>.
2. To reset the database, delete the target folder and restart the application.



5.4. Suggestions

OK, now you've got all the info you need to fix the cake club's scheduling problems. Have fun while you devise a solution!

Some suggestions that might help you out.

- The application is based on the <https://www.jhipster.tech/> architecture.
- New to angular? Learn more from the Angular Tour of Heroes: <https://angular.io/tutorial>
- At Nortal, we practice the clean code principle. Learn more: <https://medium.com/mindorks/how-to-write-clean-code-lessons-learnt-from-the-clean-code-robert-c-martin-9ffc7aef870c> and <https://cleancoders.com/>.
- How to write great unit tests: <https://stackoverflow.com/questions/3258733/new-to-unit-testing-how-to-write-great-tests>.
- How to write Angular without writing boilerplate and use industry standards: <https://cli.angular.io/>.
- Easy object validation with Spring: <https://spring.io/guides/gs/validating-form-input/>. Introduction to Typescript: <https://www.typescriptlang.org/docs/handbook/basic-types.html>.