

## Liv2Train: Problem Statement

Assume that you are working at a startup Liv2Train. You have to make an MVP for a registry for Govt funded Training Centers with the following minimum requirements. In the MVP, you have to create a Spring project with support for two APIs, which are explained below.

### First API

- Create a POST api to create and save a new training center with the following fields.
  - CenterName (text)\* (less than 40 characters)
  - CenterCode (text)\* (exactly 12 character alphanumeric)
  - Address (object)\*
    - Detailed Address
    - City
    - State
    - Pincode
  - Student Capacity (number)
  - Courses Offered (List<text>)
  - CreatedOn (Epoch time generated by System, not by user)
  - ContactEmail (text) (If present, email should be validated)
  - ContactPhone (text)\* (Phone number validation )
- Api should accept data in json.
- Validations for mandatory fields should be done, along with phone, email and size validations wherever mentioned in the aforementioned fields should be done
- As mentioned above with the fields, createdOn field should be populated from the server based on the server timestamp, and the user's input if present for this field should be discarded.
- On validation failure, an appropriate error message should be shown. This should be handled by Spring ExceptionHandler.
- On success, the api should return the newly saved TrainingCenter information in json format

### Second API

- Create a GET api to get list of all stored training centers information.
- Api should return the result in json format.
- In the absence of any training centers, we need to return an empty list in response.

## **Evaluation Criteria**

- The most important thing is to make the APIs functional. Api to create a new Training center should save it in the DB. Any subsequent api call to get a list of saved training centers should retrieve all the training centers saved till now.
- Everyone loves if things work smoothly. Try to ensure that APIs are performant.
- Validation should work perfectly and show a proper message.
- Validation should happen through annotations. Please refrain from adding conditional statements for validation.
- You are free to choose any database for persisting training center information. However, the use of an ORM (Object Relational Mapper) or ODM (Object Document Mapper) is preferred for database interaction.
- Use of Spring MVC ExceptionHandler is a bonus
- The ability to filter data in the list api is something that will be a cherry on top.
- We will also have a look at the following for evaluation
  - The project structure & architecture
  - Good coding practices, consistent coding style & formatting
  - Naming & Naming conventions
  - Good use of comments

## **Submission Guidelines**

- Please give us detailed setup instructions in a readme file.
- Email a zipped dump of your project or GitHub link with the title "Backend\_Liv2Train\_YourName".

Good luck! Remember to approach the problem in parts, and reach out to us in case of any questions.