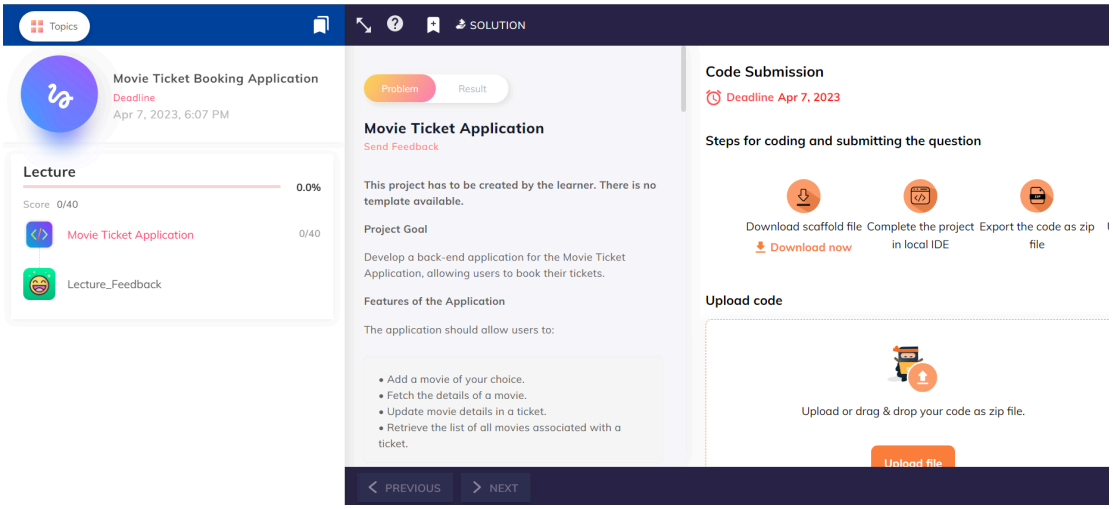
How To Attempt Mini Project?

This document aims to help learners attempt mini-projects. Learners are advised to read the document before attempting any mini-projects. To assist learners, a step-by-step guide is provided.

Guide on Attempting Mini Project:

1. Below is a mini-project from the course, and we will see how to attempt it.

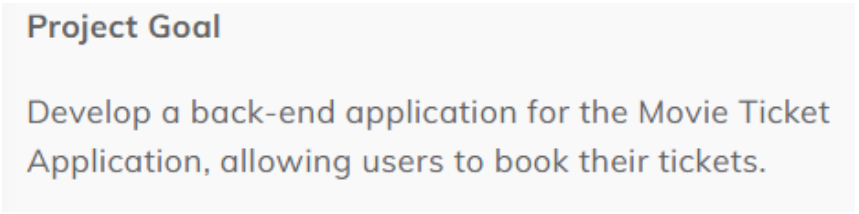


2. Mini-projects are like coding problems but with a few differences.

3. For this project, the version for Springboot should be **3.0.0,** and for Java, it should be **17**.

4. It consists of the following segments as mentioned below:

    - **Project Goal: It includes the primary motive of what we will do.**



    - **Features of the Application: It gives a general description of the features the application will have**

Features of the Application

The application should allow users to:

- Add a movie of your choice.
- Fetch the details of a movie.
- Update movie details in a ticket.
- Retrieve the list of all movies associated with a ticket.

- **Steps: To solve a mini project successfully, we have broken them down into smaller steps, and you must complete them accordingly.**

Steps:

Use the following guidelines and hints to build the project.

1. Create a Model class named Movie having the following attributes:

- String movie name
- String movie director
- long movie rating
- String movie language

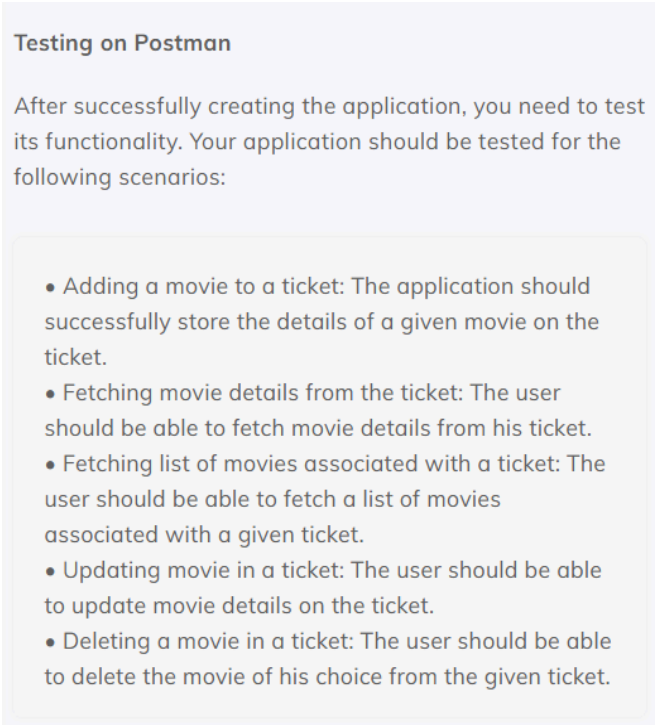
- **End Points To Be Created: It contains the information regarding the APIs you are supposed to design.**

End Points To Be Created

Movie Booking Endpoints:

- GET /ticket/movies: Retrieve the list of all movies associated with a ticket.
- GET /ticket/movie/{id} : Retrieve a movie based on the given id.
- POST /ticket/movie: Adds a movie in a ticket (Body: Movie movie, BindingResult bindingResult).
- PUT /ticket/update/{id}: Updates a movie in a ticket.
- DELETE /ticket/movie/{id}: Deletes a specific movie from the given ticket

- **Testing on Postman: This section briefs about testing your application through Postman.**

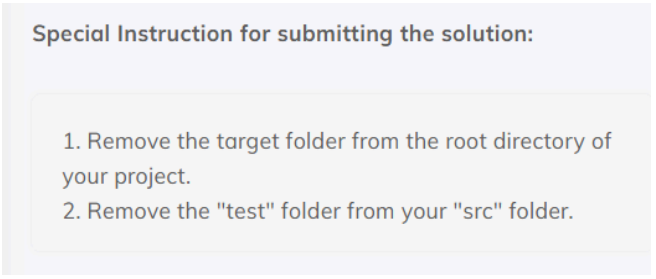


Testing on Postman

After successfully creating the application, you need to test its functionality. Your application should be tested for the following scenarios:

- Adding a movie to a ticket: The application should successfully store the details of a given movie on the ticket.
- Fetching movie details from the ticket: The user should be able to fetch movie details from his ticket.
- Fetching list of movies associated with a ticket: The user should be able to fetch a list of movies associated with a given ticket.
- Updating movie in a ticket: The user should be able to update movie details on the ticket.
- Deleting a movie in a ticket: The user should be able to delete the movie of his choice from the given ticket.

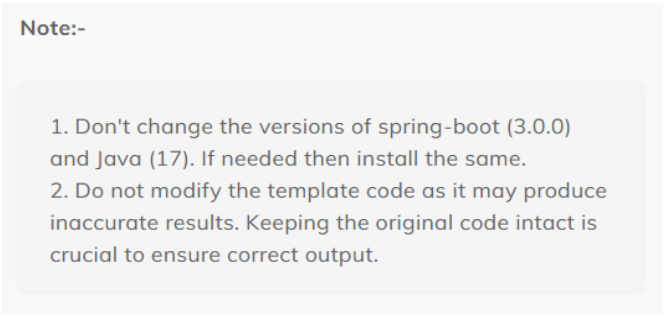
5. Special Instructions have guidelines for submitting a solution that needs to be followed before submission.



Special Instruction for submitting the solution:

1. Remove the target folder from the root directory of your project.
2. Remove the "test" folder from your "src" folder.

6. The Note section contains the do's and don'ts, which are the basic requirements for the project.



Note:-

1. Don't change the versions of spring-boot (3.0.0) and Java (17). If needed then install the same.
2. Do not modify the template code as it may produce inaccurate results. Keeping the original code intact is crucial to ensure correct output.