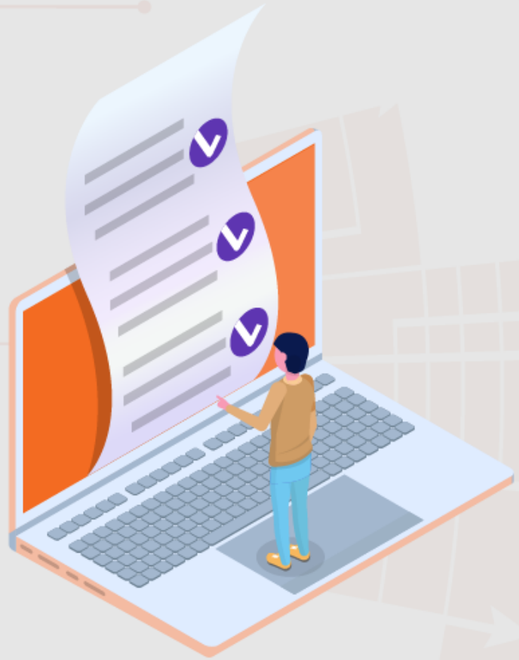
API Endpoints And Communication

Course-End Project

Objectives

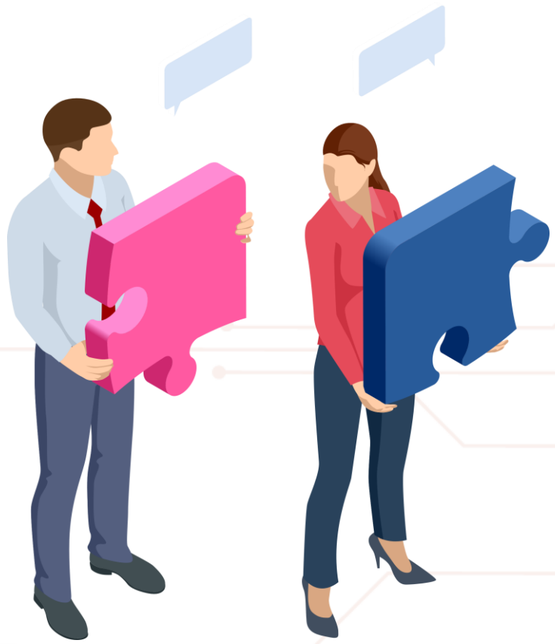
To build an application to handle data of bookings and passenger profiles for a travel company using microservices and API frameworks.



Prerequisites

- JUnit
- Spring
- SpringBoot
- WebServices
- MicroServices

Problem Statement and Motivation



Problem Statement:

This assignment is designed to help understand how to plan and develop the back end for a given problem. Further, to gain hands-on experience in designing the microservice architecture for the project and finally perform unit testing for the code.

Real-World Scenario:

George owns a Travel Company where he books cabs for his customers. To manage these bookings, he needs Travel Management Software. He meets Kia, who runs a Software Solution Company and tells her the requirement.

Kia aims to develop the Project using Spring and Spring Boot Framework. She needs to create microservices to provide a solution using REST architecture. Finally, perform the unit testing for various web methods.

Industry Relevance

Skills used in the project and their usage in the industry are given below:

Junit:

Popular functional testing tool for Java application.

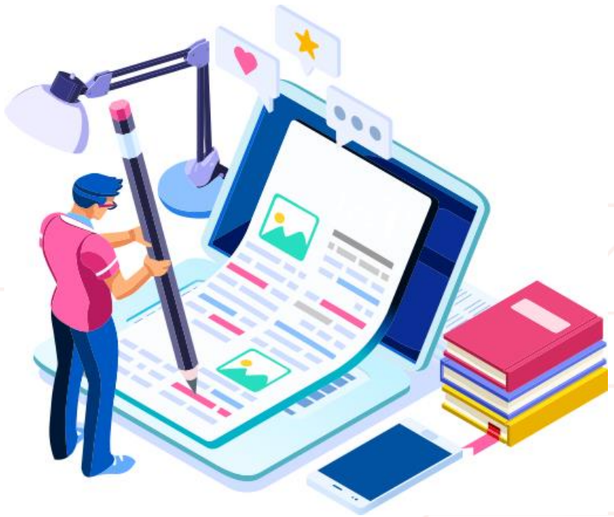
Spring and Springboot:

These are most popular and commonly used Java frameworks that help in making the class mapping and frontend to backend object conversion and data transfer.

Webservices and Microservices:

They attempt to address a single concern, such as a data search, logging function, or web service function. They help in implementing distributed computing in code.

Task (Activities)



1. Create a Spring Boot Application in Eclipse EE
2. Configure Spring Web dependency in the project
3. Develop HTML Web Pages for cab booking
4. Create a controller with annotations from Spring
5. Create a microservice to book the cab
6. Create a microservice to calculate the fare for cab
7. Write unit testcases to test the microservices
8. Build run and test the project on Postman
9. Test the project with front end web pages
10. Create an executable jar file using maven wrapper

Project Reference

Task 1 and 2:

Springboot - Lesson 1

Task 3:

Course 1: HTML

Task 4:

Spring – Lesson 3

Task 5 and 6:

Springboot – Lesson 3

Task 7:

Junit – Lesson 1

Task 8:

Springboot – Lesson 2

Submission Process

You will have to submit the project in 1 week.

It is recommended to work on the integrated labs as they have all the required tools available

Project can be submitted from the assessment tab followed by clicking on the **Submit** button.

Provide the documents mentioned below:

- Source Code in zip
- Database scripts to replicate your database settings
- Screenshots of the outputs

Reference Outputs

book-cab.html

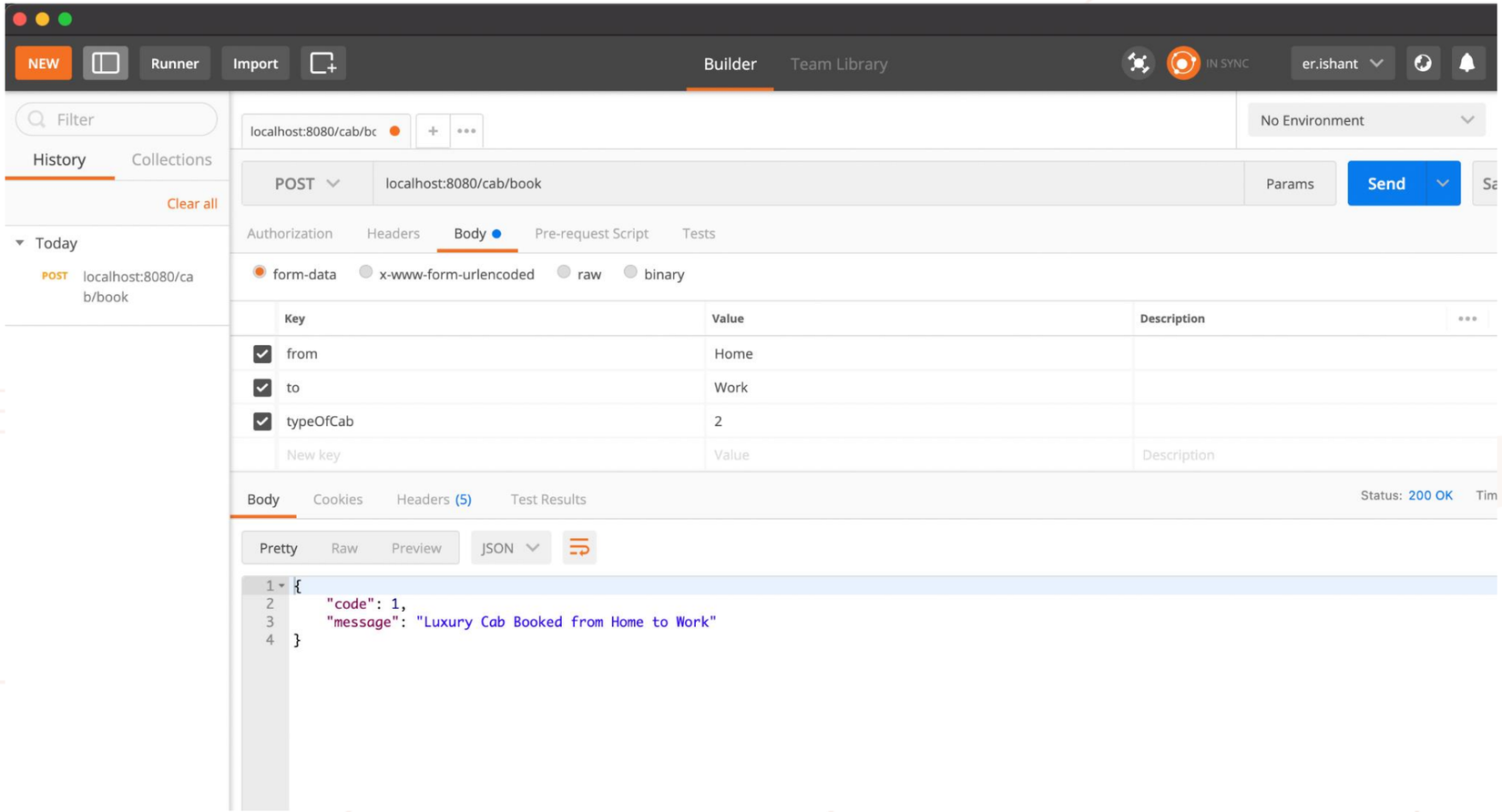
```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Book Cab</title>
<style>
.center {
  margin: auto;
  width: 60%;
  border: 3px solid #FFA500;
  padding: 10px;
}
</style>
</head>
<body>
<div class="center">
<h3>Book A Cab</h3>
<form action="/cab/book" method="post">
  <label for="from">From Location:</label><br>
  <input type="text" id="from" name="from" placeholder="Home"><br>
  <label for="to">To Location:</label><br>
  <input type="text" id="to" name="to" placeholder="Work"><br><br>
  <label for="typeOfCab">Type of Cab:</label><br>
  <input type="text" id="typeOfCab" name="typeOfCab" placeholder="2"><br><br>
  <input type="submit" value="Submit">
</form>
</div>
</body>
</html>
```

Reference Outputs

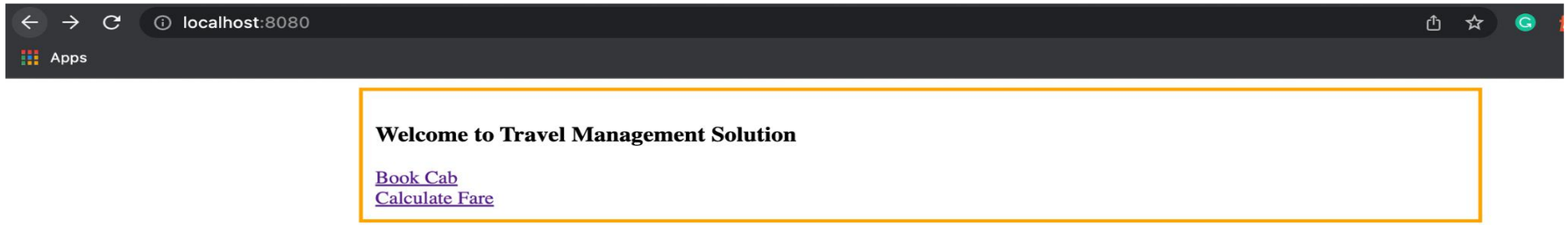
SpringBootAssignmentSolutionApplicationTests.java

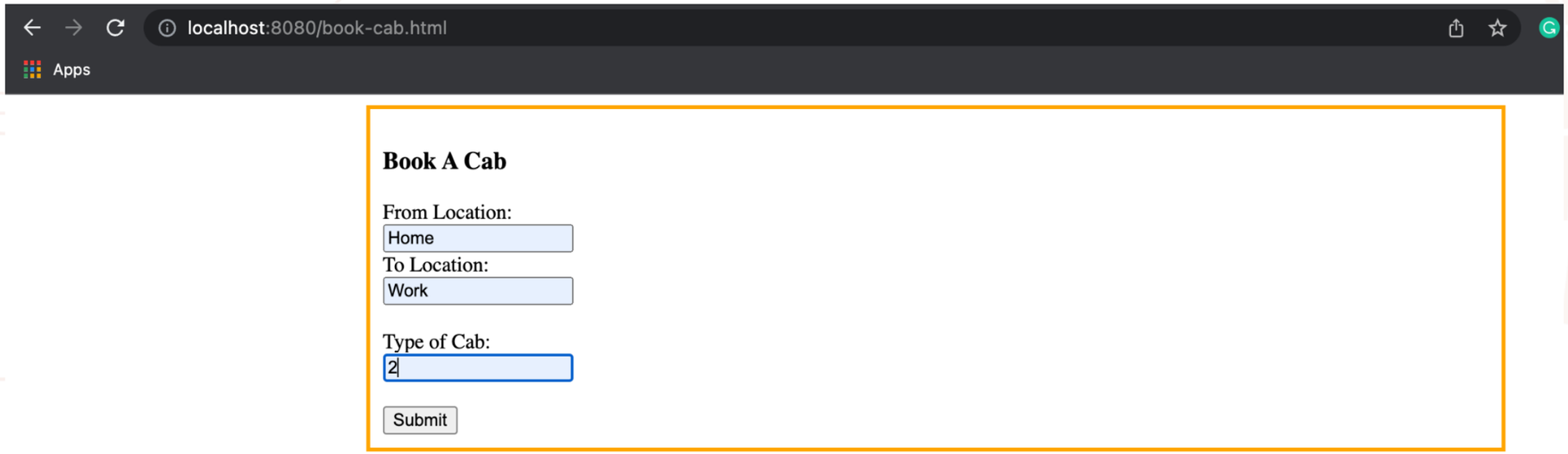
```java
package com.travel.george;

import static org.assertj.core.api.Assertions.assertThat;

import org.junit.jupiter.api.Test;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;

import com.travel.george.controller.CabBookingController;

@SpringBootTest
class SpringBootAssignmentSolutionApplicationTests {

    @Autowired
    private CabBookingController controller;

    @Test
    public void contextLoads() throws Exception {
        assertThat(controller).isNotNull();
    }

}

```

Reference Outputs



Reference Outputs





Thank you