

Task 1

Create a full stack online metro ticket booking system.

Tanvi V. Chaudhari [LinkedIn](#)
B.Tech [GitHub](#)
CSE Branch [Email](#)
IIT Gandhinagar

Requirements

1. Java SDK Version 17
2. Spring Boot Initializr
3. IntelliJ Community 2024

Steps to Follow

1. Download dependencies from Spring Boot Initializr, mentioned in pom.xml

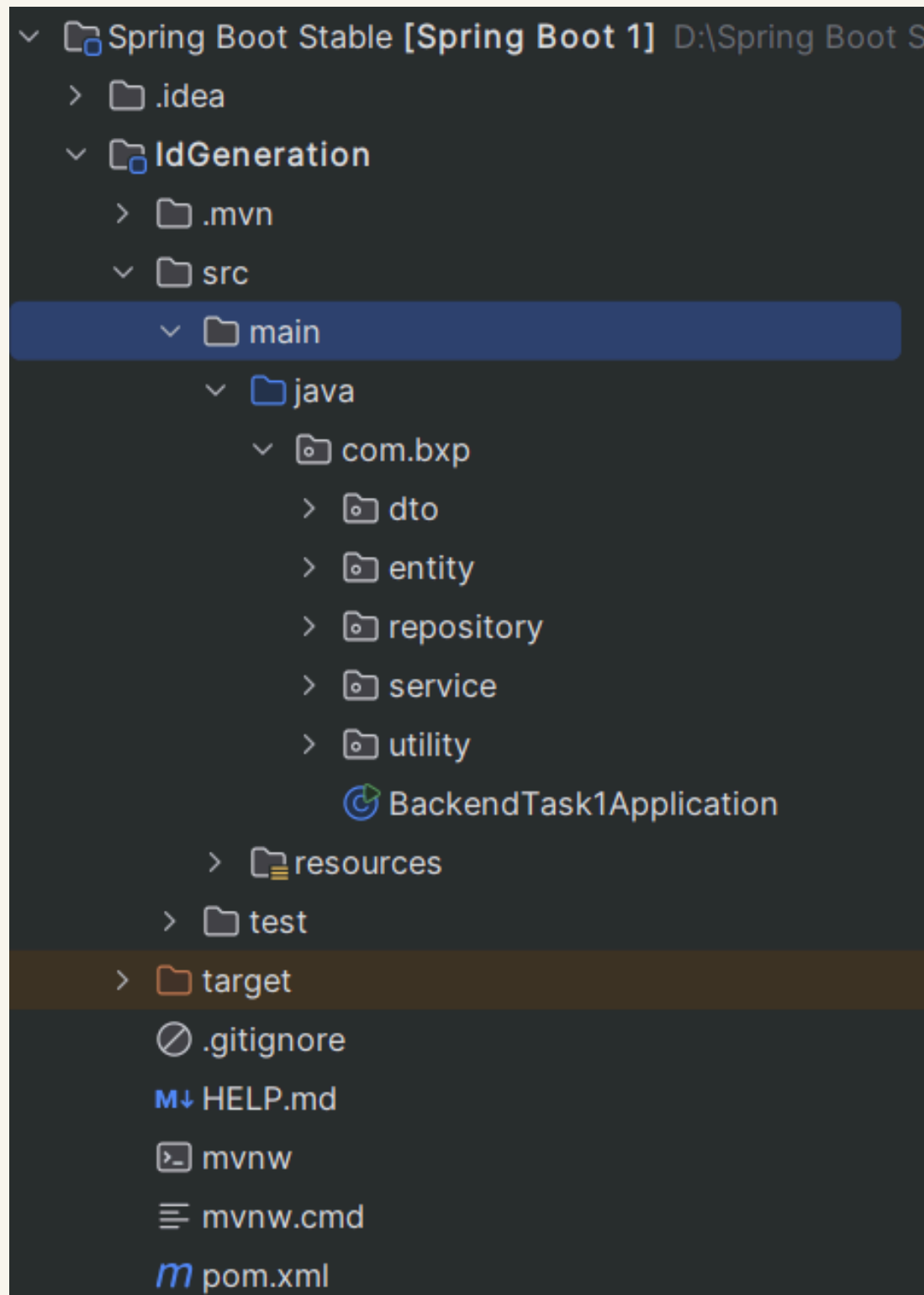
The screenshot shows the Spring Boot Initializr web application interface. The browser address bar displays `https://start.spring.io`. The interface is divided into several sections:

- Project:** Includes radio buttons for `Gradle - Groovy`, `Gradle - Kotlin`, and `Maven` (selected).
- Language:** Includes radio buttons for `Java` (selected), `Kotlin`, and `Groovy`.
- Spring Boot:** Includes radio buttons for `3.4.0 (SNAPSHOT)`, `3.4.0 (M2)`, `3.3.4 (SNAPSHOT)`, `3.3.3`, `3.2.10 (SNAPSHOT)`, and `3.2.9` (selected).
- Project Metadata:** Includes input fields for `Group` (com.bxp), `Artifact` (IdGeneration), `Name` (Backend_Task1), `Description` (Task1: Metro System Id Generation for bxp in SpringBoot), and `Package name` (com.bxp.IdGeneration).
- Packaging:** Includes radio buttons for `Jar` (selected) and `War`.
- Java:** Includes radio buttons for `22`, `21`, and `17` (selected).

On the right side, there are several dependency cards:

- Lombok** (DEVELOPER TOOLS): Java annotation library which helps to reduce boilerplate code.
- MySQL Driver** (SQL): MySQL JDBC driver.
- Spring Data JPA** (SQL): Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.
- Spring Boot DevTools** (DEVELOPER TOOLS): Provides fast application restarts, LiveReload, and configurations for enhanced development experience.
- Spring Configuration Processor** (DEVELOPER TOOLS): Generate metadata for developers to offer contextual help and "code completion" when working with custom configuration keys (ex.application.properties/.yml files).
- Spring REST Docs** (TESTING): Document RESTful services by combining hand-written with Asciidoctor and auto-generated snippets produced with Spring MVC Test.

At the bottom, there are three buttons: **GENERATE** (CTRL + ↵), **EXPLORE** (CTRL + SPACE), and **SHARE...**



2. Open the installed spring boot folder after extracting on IntelliJ Community 2024 named Spring Boot Stable using Maven build.

3. Here, file and directory order for this project.

4. Connect ot MySQL Workbench and write sql code in the file “database.sql”.

The screenshot displays the MySQL Workbench interface. The left sidebar shows the 'SCHEMAS' panel with a search filter and a list of databases: oceo, pdc, and sys. The main editor window is titled 'SQL File 4*' and contains the following SQL code:

```
1 • drop database if exists metroSystem_db;
2
3 • create database metroSystem_db;
4
5 • use metroSystem_db;
6
7 • create table customer (
8     customer_id int auto_increment ,
9     email_id varchar(30),
10    name varchar(30),
11    creation_time time,
12    constraint ps_customer_id_pk primary key ( customer_id )
13 );
```

Below the code editor, the 'Result Grid' shows the execution results of the SQL statements. The grid has columns for customer_id, email_id, name, and creation_time. The results are as follows:

customer_id	email_id	name	creation_time
552101	random1@example.com	Alice Johnson	09:12:34
552102	random2@example.com	Bob Smith	10:25:47
552103	random3@example.com	Charlie Brown	11:30:59
NULL	NULL	NULL	NULL

The 'Output' panel at the bottom shows the 'Action Output' for the executed statements:

#	Time	Action	Message	Duration / Fetch
✓ 37	15:38:53	INSERT INTO customer (customer_id, email_id, name, creation_time) VALUES (552101, 'random1@exa...	1 row(s) affected	0.000 sec
✓ 38	15:38:53	INSERT INTO customer (customer_id, email_id, name, creation_time) VALUES (552102, 'random2@exa...	1 row(s) affected	0.016 sec
✓ 39	15:38:53	INSERT INTO customer (customer_id, email_id, name, creation_time) VALUES (552103, 'random3@exa...	1 row(s) affected	0.000 sec
✓ 40	15:38:53	commit	0 row(s) affected	0.000 sec
✓ 41	15:38:53	select * from customer LIMIT 0, 1000	3 row(s) returned	0.000 sec / 0.000 sec

The status bar at the bottom indicates 'Query Completed'.

My Changes in the Source
Code...Soon!

Github

1. Please find the github repository, [here](https://github.com/Tanvi-Chaudhari/Fetching-Emails). (Link -> <https://github.com/Tanvi-Chaudhari/Fetching-Emails>)
2. I have provided requirements, steps to follow, github link, and resources in this presentation.

Resources

1. <https://github.com/Arpitbrur/Online-train-ticket-booking-Spring-Boot-project/tree/main>
 2. Guides: The following guides illustrate how to use some features concretely:
 3. [Accessing Data with JPA](#)
 4. [Accessing data with MySQL](#)
 5. [Building a RESTful Web Service](#)
 6. [Serving Web Content with Spring MVC](#)
 7. [Building REST services with Spring](#)
- 