Assignment 05

Design and deploy Leave Management System using Spring Web

# Step 1: Create Project Using Spring Initializr

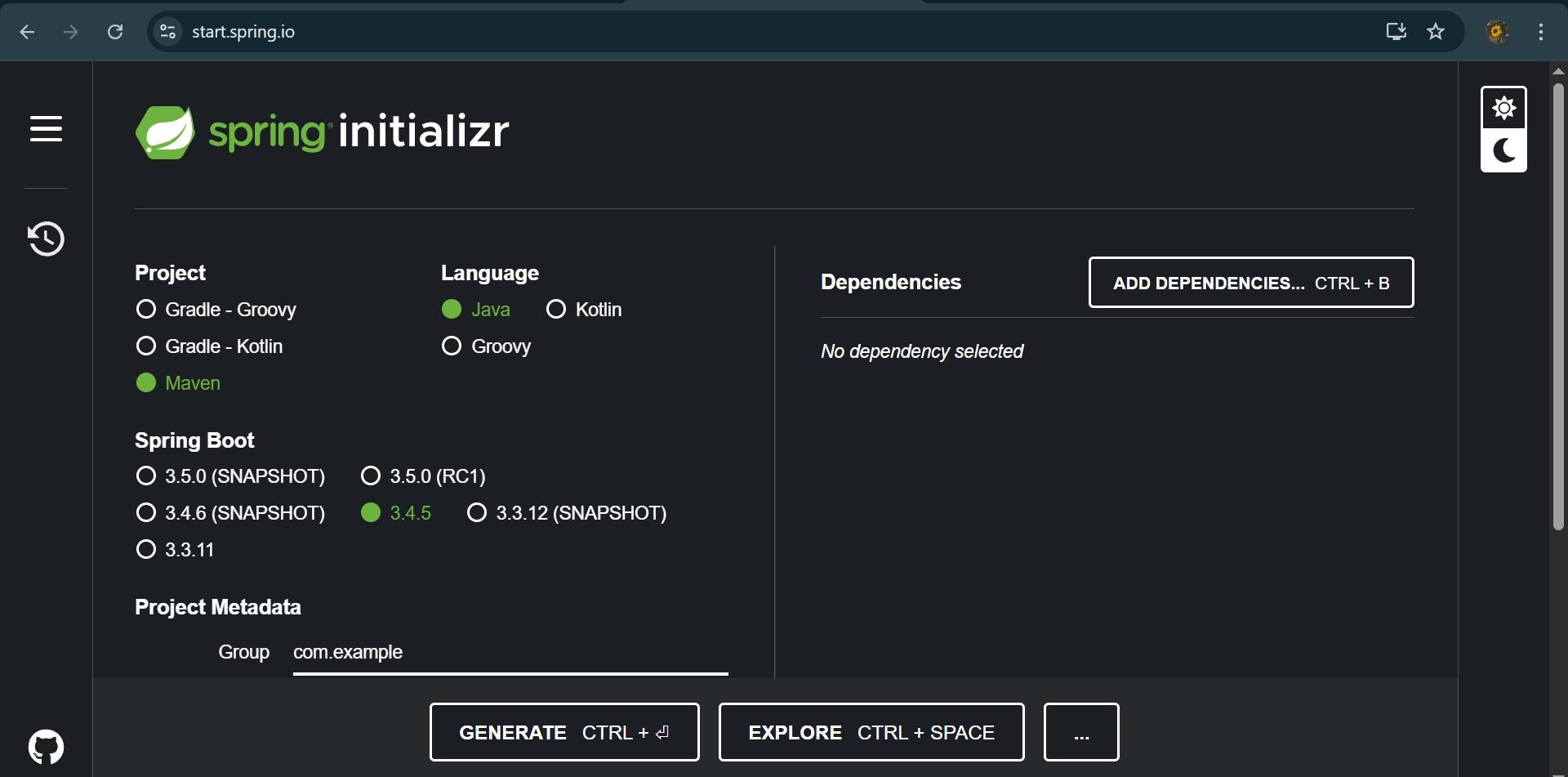
Go to [https://start.spring.io](https://start.spring.io/)

Choose:

* **Project**: Maven
* **Language**: Java
* **Spring Boot Version**: Latest stable
* **Group**: com.example
* Artifact: bussystem
* Name: BusManagementSystem
* **Packaging**: Jar
* **Java**: 17 or above

# Dependencies:

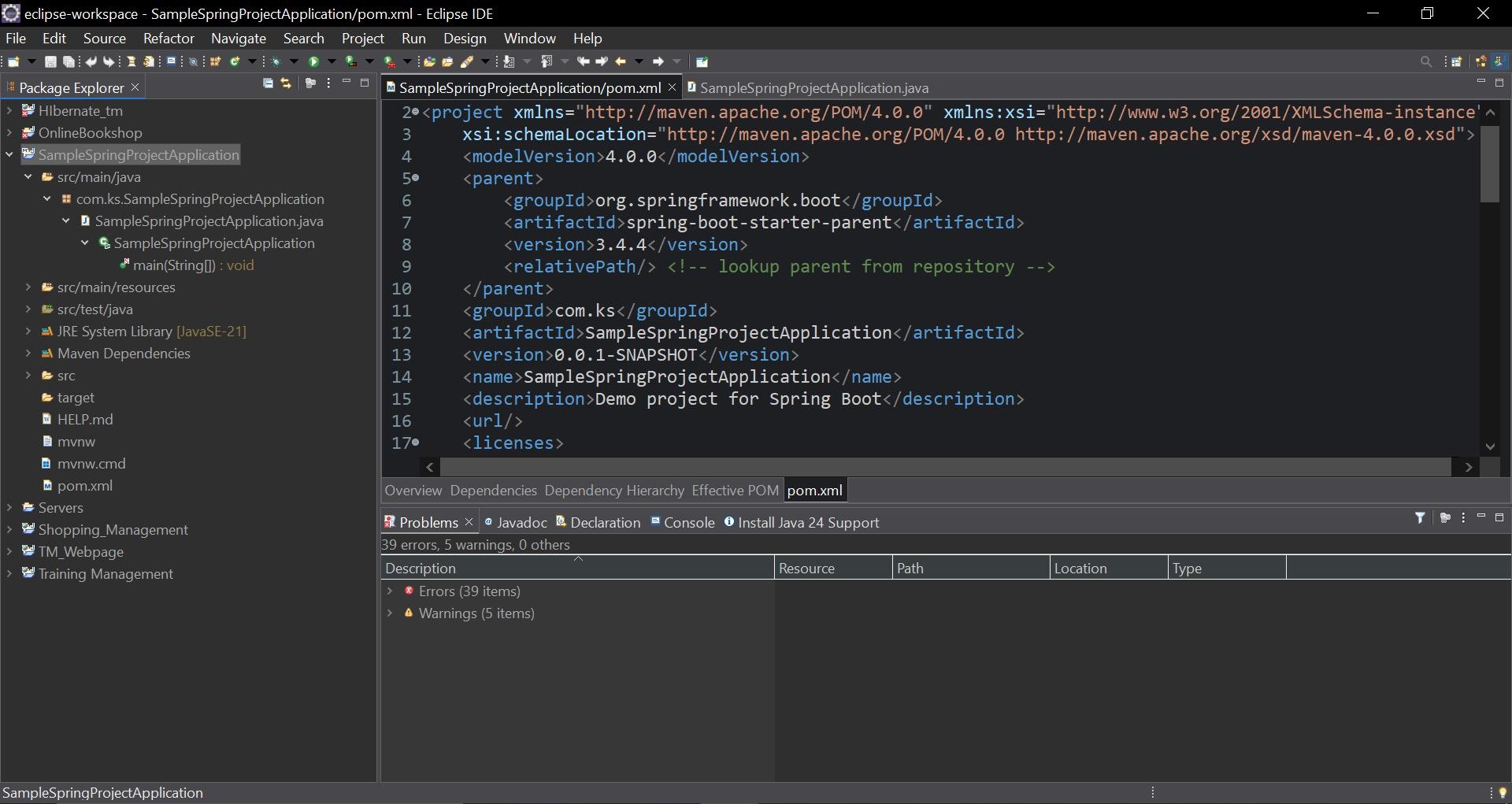
* + Spring Web
  + Spring Data JPA
  + MySQL Driver
  + Spring Boot DevTools (optional for hot reload)



Click on **Generate** → Extract the ZIP file.

# Step 2: Import the Project

Open IDE → File → Import → Maven → Existing Maven Projects → Select extracted folder → Finish



# Step 3: Configure pom.xml dependencies (already included if you used Spring Initializr)

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

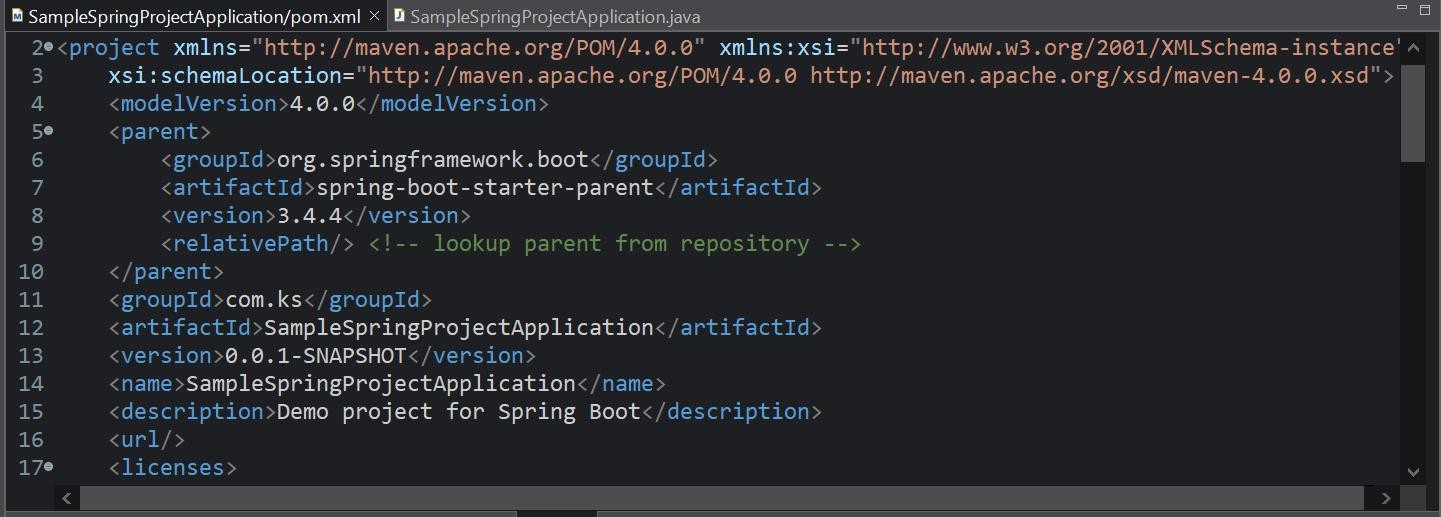
<dependency>

<groupId>com.mysql</groupId>

<artifactId>mysql-connector-j</artifactId>

<scope>runtime</scope>

</dependency>



# Step 4: Create Model (Entity) Class — Book.java

@Entity

public class Book { @Id

@GeneratedValue(strategy = GenerationType.IDENTITY) private Long id;

private String title; private String author; private String isbn;

private int quantity;

// Getters & Setters

}

# Step 5: Create Repository — BookRepository.java

@Repository

public interface BookRepository extends JpaRepository<Book, Long> {

}

# Step 6: Create Service Layer — BookService.java

@Service

public class BookService { @Autowired

private BookRepository bookRepo;

public List<Book> getAllBooks() { return bookRepo.findAll();

}

public Book addBook(Book book) {

return bookRepo.save(book);

}

public void deleteBook(Long id) { bookRepo.deleteById(id);

}

}

# Step 7: Create Controller — BookController.java

@RestController @RequestMapping("/books") public class BookController {

@Autowired

private BookService bookService;

@GetMapping

public List<Book> getBooks() { return bookService.getAllBooks();

}

@PostMapping

public Book addBook(@RequestBody Book book) { return bookService.addBook(book);

}

@DeleteMapping("/{id}")

public void deleteBook(@PathVariable Long id) { bookService.deleteBook(id);

}

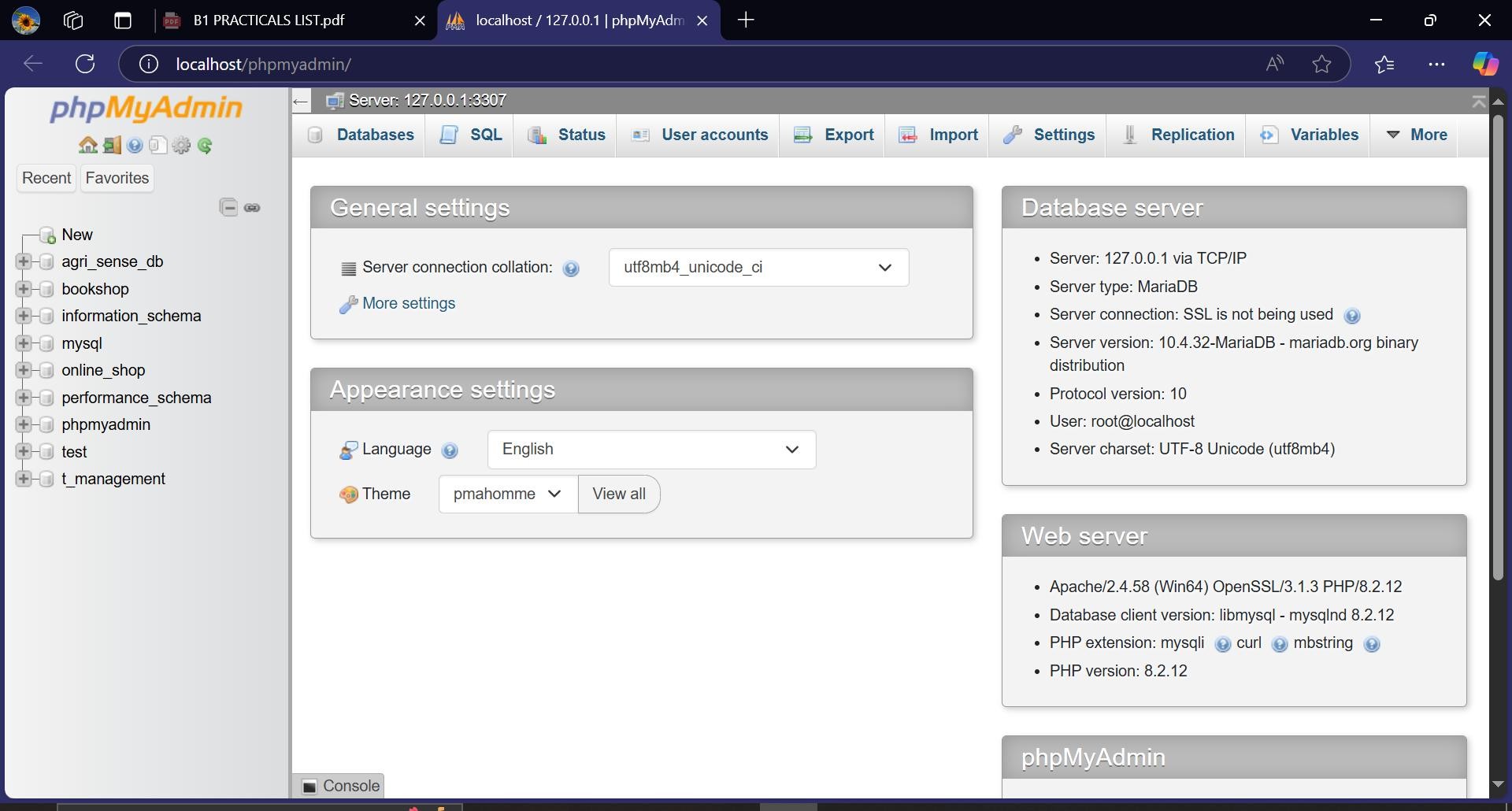
}

Step 8: Configure MySQL Connection Add to application.properties

spring.datasource.url=jdbc:mysql://localhost:3306/busdb spring.datasource.username=root spring.datasource.password=your\_password spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

Create a database busdb in MySQL before running.



# Step 9: Run the Application

Right click on BusManagementSystemApplication.java → Run as Java Application

**TEST CASES:**

**Test Case ID**

**Test Scenario Input Expected Output Result**

TC01 Get all books GET /books Returns list of all books

(JSON)

Pass/Fail

TC02 Add new book POST /books with

JSON {title, author...}

Returns saved book with generated id

Pass/Fail

TC03 Delete existing book

DELETE /books/1 Deletes book with ID 1 and

returns no content

Pass/Fail

TC04 Get books when

none exist

GET /books (empty DB)

Returns empty list [] Pass/Fail

TC05 Add book with

missing data

POST /books with missing title

Returns error (400 Bad Request or validation msg)

Pass/Fail