Week-6 Spring Boot Fundamentals

📘 Blog System REST API - Spring Boot

Description

This project is a **RESTful Blog Management API** built using **Spring Boot 3.x**. It provides full CRUD operations for **Blog Posts, Comments, and Users**. The API follows the **Spring MVC architecture**, uses **Spring Data JPA** for database access, and is tested using **Postman**.

Reputable Sources (Cited)

- Spring Boot Official Docs: https://docs.spring.io/spring-boot/docs/current/reference/html/
- Spring Data JPA Docs: https://spring.io/projects/spring-data-jpa
- REST API Design Guidelines (MDN): https://developer.mozilla.org/en-US/docs/Glossary/REST
- HTTP Methods Documentation (MDN): https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods

🚀 Features

- ✓ CRUD operations for **Posts, Comments, Users**
- ✓ RESTful API endpoints
- ✓ Spring Data JPA integration
- ✓ Validation + error responses
- ✓ Layered architecture (Controller → Service → Repository → Entity)
- ✓ Postman-tested endpoints
- ✓ MySQL / H2 support
- ✓ Clean code structure

🛠 Technology Stack

- Java 17+
- Spring Boot 3.x
- Spring MVC
- Spring Data JPA
- Hibernate ORM
- MySQL / H2 Database
- Maven
- Postman

📁 Project Structure (Based on Your Screenshot)

```
src/
└── main/java/com/blog/
    ├── BlogApplication.java
    │
    ├── controller/
    │   ├── PostController.java
    │   ├── CommentController.java
    │   └── UserController.java
    │
    ├── model/
    │   ├── Post.java
    │   ├── Comment.java
    │   └── User.java
    │
    ├── repository/
    │   ├── PostRepository.java
    │   ├── CommentRepository.java
│       └── UserRepository.java
│
└── service/
        ├── PostService.java
        ├── CommentService.java
        └── UserService.java
resources/
 └── application.properties
pom.xml
```

🔗 API Endpoints

📌 Posts

Method	Endpoint	Description
GET	/posts	Get all posts
GET	/posts/{id}	Get post by ID
POST	/posts	Create post
PUT	/posts/{id}	Update post
DELETE	/posts/{id}	Delete post

📌 Comments

Method	Endpoint	Description
GET	/comments	Get all comments
GET	/comments/{id}	Get comment by ID
POST	/comments	Create comment
PUT	/comments/{id}	Update comment
DELETE	/comments/{id}	Delete comment

📌 Users

Method	Endpoint	Description
GET	/users	Get all users
GET	/users/{id}	Get user by ID
POST	/users	Create user
PUT	/users/{id}	Update user
DELETE	/users/{id}	Delete user

📦 How to Run

1️⃣ Clone the Repository

```
git clone <your-repository-url>
cd blog-system
```

2️⃣ Configure Database

In **application.properties**:

```
spring.datasource.url=jdbc:mysql://localhost:3306/blog
spring.datasource.username=root
spring.datasource.password=rajib@1234
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
```

3️⃣ Run the Application

```
mvn spring-boot:run
```

or

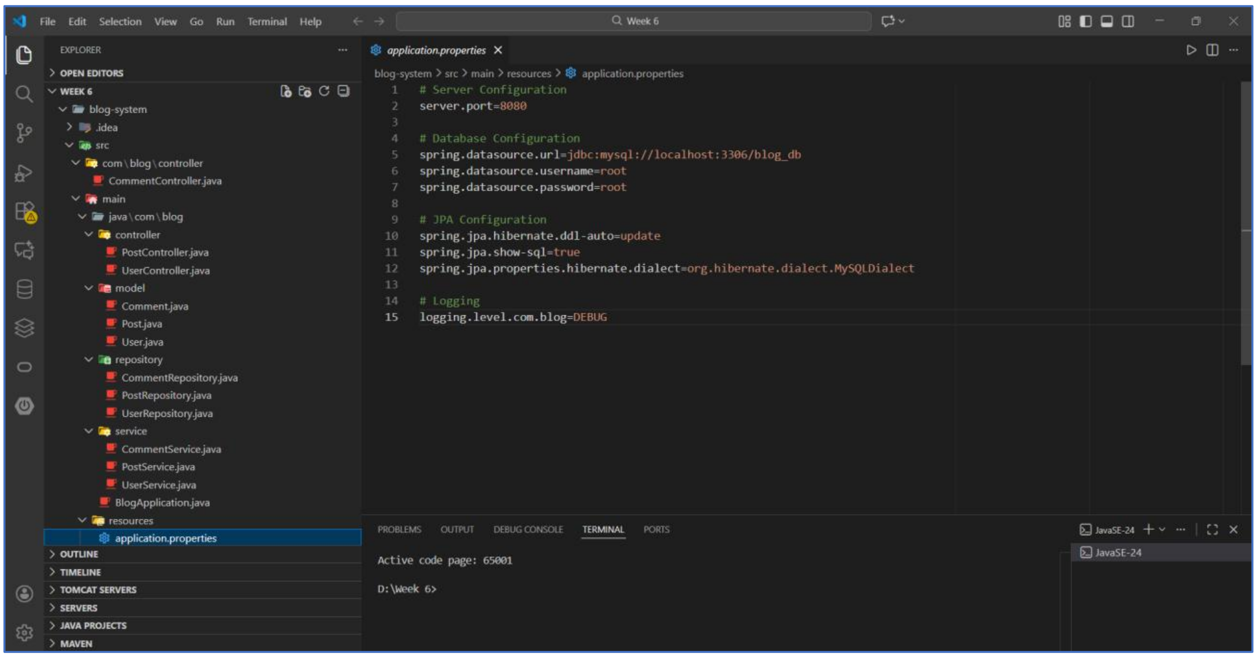
```
mvn clean package
java -jar target/blog-0.0.1-SNAPSHOT.jar
```

API runs at:
http://localhost:8080

🧪 Sample API Requests

Create a Post

```
curl -X POST http://localhost:8080/posts \
  -H "Content-Type: application/json" \
  -d '{
    "title": "New Blog Post",
    "content": "Spring Boot makes API building easy.",
    "author": "Rajib"
  }'
```



```
WEEK 6
  blog-system
    .idea
    src
      com\blog\controller
        CommentController.java
      main
        java\com\blog
          controller
            PostController.java
            UserController.java
          model
            Comment.java
            Post.java
            User.java
          repository
            CommentRepository.java
            PostRepository.java
            UserRepository.java
          service
            CommentService.java
            PostService.java
            UserService.java
          BlogApplication.java
        resources
          application.properties
```

blog-system > src > main > resources > application.properties

```
# Server Configuration
server.port=8080

# Database Configuration
spring.datasource.url=jdbc:mysql://localhost:3306/blog_db
spring.datasource.username=root
spring.datasource.password=root

# JPA Configuration
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect

# Logging
logging.level.com.blog=DEBUG
```

```
Active code page: 65001

D:\Week 6>
```

Get All Posts

```
curl http://localhost:8080/posts
```

📊 Data Flow Diagram (Simple)

```
User Request
   |
   ▼
Controller → calls → Service → interacts → Repository → DB
   |
   ▼
JSON Response
```