**Documentation**

**Overview**

This document provides the steps required to set up and run the Spring Boot application locally or on a server. It covers prerequisites, setting up dependencies, building the application, running it, and other configurations such as database setup.

### ****1. Prerequisites****

Before running the Spring Boot application, ensure that the following software is installed on your machine:

* **Java 17 or newer** (for Spring Boot 3.x applications):
  + You can download it from the [official Oracle website](https://www.oracle.com/java/technologies/javase-jdk17-downloads.html) or install it via a package manager.
  + Spring Boot 3.x requires at least Java 17.
* **Maven**  (for building the application):
  + If you use Maven, you can download it from the [official Maven site](https://maven.apache.org/download.cgi).
* **IDE (Integrated Development Environment)**:
  + It is recommended to use IDEs like [IntelliJ IDEA](https://www.jetbrains.com/idea/) or [Spring Tool Suite](https://spring.io/tools) for easier development and running the application.
* **Database (e.g., MySQL)**:
  + If the application connects to a relational database, ensure that the required database system is installed.

### ****2. Clone the Repository****

To run the Spring Boot application, you first need to clone the repository from the source control (GitHub).

1. Open a terminal or Git Bash window.
2. Use the following command to clone the repository:

git clone <https://github.com/vaddemanibhargav/Task-management.git>

1. Navigate to the project directory:

cd your-repository-name

### ****3. Build the Application****

Once the repository is cloned, you need to build the application.

1. Open a terminal or command prompt in the project directory.
2. Run the following Maven command to build the project:

**maven clean install**

This will download dependencies, compile the code, run tests, and package the application as a JAR file.

### ****4. Running the Application from IDE****

* Open the project in your IDE (IntelliJ IDEA, Spring Tool Suite, etc.).
* Locate the main () method in the @SpringBootApplication class (usually named Application.java).
* Right-click the class and choose **Run** or use the IDE's run command.

### ****5. Configuration****

The application might require configuration for things like database connections, logging, and external APIs.

**Application Properties**

Spring Boot applications are typically configured using the application.properties or application.yml file located in the src/main/resources directory.

My file copy

|  |
| --- |
| spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver spring.datasource.url = jdbc:mysql://localhost:3306/taskmanager spring.datasource.username = root spring.datasource.password = root spring.jpa.show-sql = true spring.jpa.hibernate.ddl-auto = update spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQLDialect server.port=8090 |

 server.port: Specifies the port on which the application will run (default is 8080).

 spring.datasource.url: The URL to connect to the database.

 spring.datasource.username & spring.datasource.password: The database credentials.

 spring.jpa.hibernate.ddl-auto: Defines the Hibernate DDL mode (e.g., update, validate, none).

**Database Configuration**

If your Spring Boot application uses a database, you need to ensure that the database is set up properly.

For MySQL, for example:

1. Install MySQL locally or use a hosted database.
2. Create a database:

Create database taskmanager

1. Ensure that the username and password are correctly configured in application.properties.

### ****6. Testing API Endpoints with Postman****

We'll test a few common CRUD operations (Create, Read, Update, Delete) using Postman. Let’s assume you have the following API endpoints for managing tasks:

* **GET /api/tasks**: Retrieve all tasks.
* **GET /api/tasks/{taskId}**: Retrieve a specific task by its ID.
* **POST /api/tasks**: Create a new task.
* **PUT /api/tasks/{taskId}**: Update an existing task.
* **DELETE /api/tasks/{taskId}**: Delete a task by its ID

### ****Conclusion****

This documentation provides the steps required to clone, build, and run your Spring Boot application locally or in a Docker container. It also includes the configuration options for connecting to a database, as well as how to do postman testing.

**API Documentation**

Here I prepare API documentation using Postman you can refer <https://documenter.getpostman.com/view/39623159/2sAY52cyzY>

When server is up and running you can refer this also

<http://localhost:8090/swagger-ui/index.html#/>