**API Endpoints**

| **Endpoint** | **Method** | **Description** | **Request Body / Parameters** | **Response** |
| --- | --- | --- | --- | --- |
| /items/{id} | GET | Get item by ID | Path variable: id (int) | Returns the item with the specified ID |
| /items | GET | Get all items | None | Returns a list of all items |
| /items | POST | Save a new item | JSON object (item) | Returns the saved item with its ID |

**List of dependencies**

1. Spring Web
2. Spring Data JPA
3. MySQL

**Request and Response Formats**

**1. GET ---- /items/{id}**

**Description:** Retrieve an item by its ID.  
**Request Example:**

(in postman: GET) http://localhost:8080/items/1

**Response Example:**

{

"id": 1,

"name": "Laptop",

"description": "A high-end laptop",

"price": 1200

}

**2. GET --- /items**

**Description:** Retrieve a list of all available items.  
**Request Example:**

(in postman: GET) http://localhost:8080/items

**Response Example:**

[

{

"id": 1,

"name": "Laptop",

"description": "A high-end laptop",

"price": 1200

},

{

"id": 2,

"name": "Phone",

"description": "A flagship smartphone",

"price": 800

}

]

**3. POST --- /items**

**Description:** Save a new item to the database.  
**Request Example:**

(in postman: POST) http://localhost:8080/save-item

{

"name": "Tablet",

"description": "A 10-inch tablet",

"price": 500

}'

**Response Example:**

json

Copy code

{

"id": 3,

"name": "Tablet",

"description": "A 10-inch tablet",

"price": 500

}

**Setup Instructions**

**1. Prerequisites**

* **Java 17+** installed
* **MySQL** installed and running
* **Maven** installed
* **Git** installed

**2. Clone the Project**

Open your terminal or command prompt and run:

git clone https://github.com/UtpalPatel0055/Consult-Add

**3. Configure the Database**

1. Start your MySQL server.
2. Create a database named demo:

CREATE DATABASE demo;

**4. Update Database Configuration**

In the src/main/resources/application.properties file, update the following properties if needed:

properties

Copy code

spring.datasource.url=jdbc:mysql://localhost:3306/demo

spring.datasource.username=root

spring.datasource.password=user@1234

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

**5. Build and Run the Application**

In the root directory of the project, run the following Maven commands:

mvn clean install

mvn spring-boot:run

If everything is set up correctly, the application should start on **http://localhost:8080**.

**Error Handling**

* If an item with a specific ID is not found, the API will return a **404 Not Found** status.
* For any internal server errors, the API will return a **500 Internal Server Error** status along with an error message.