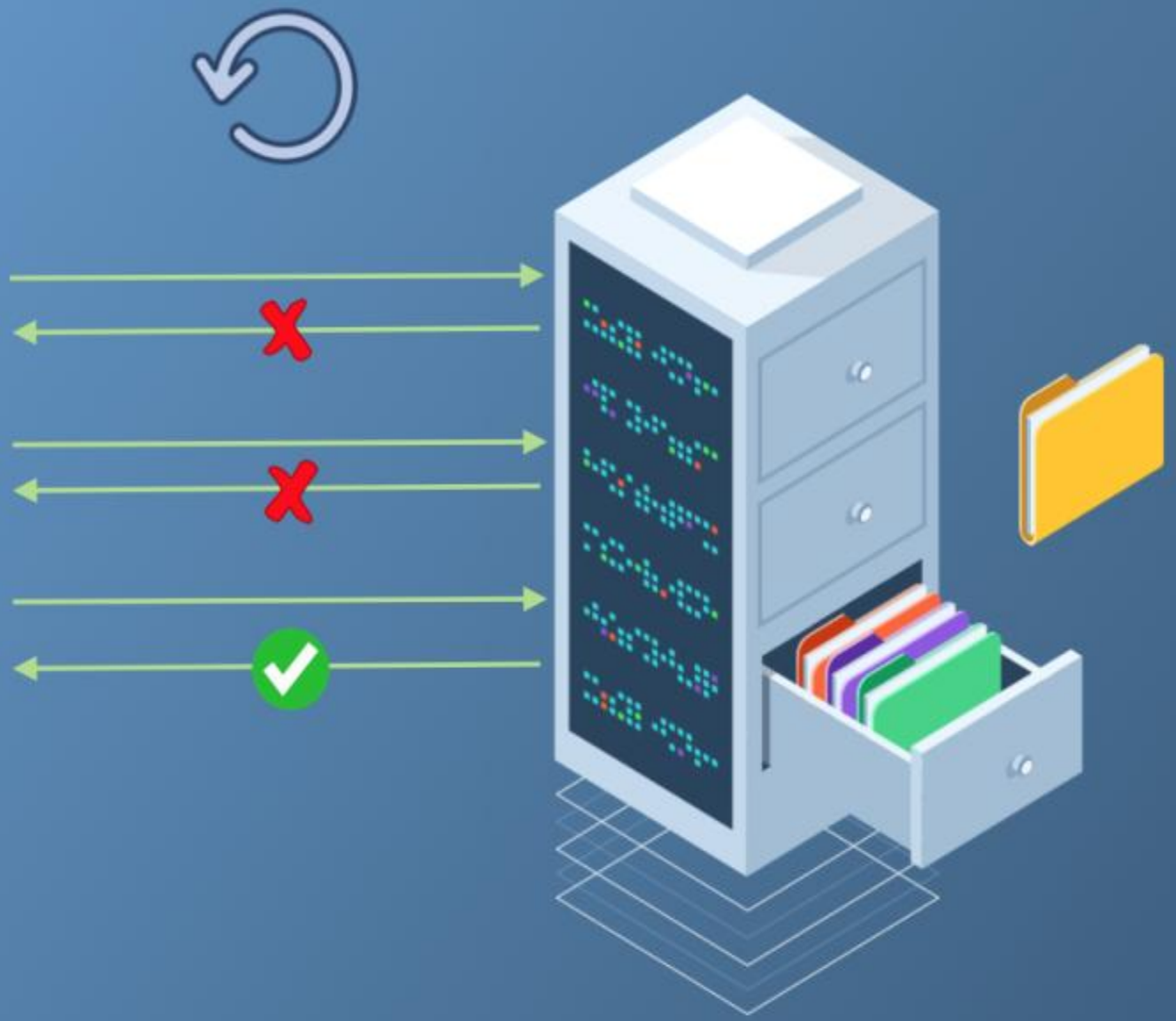


# Database Connectivity

# Introduction

- Simplifies Java enterprise development
- Built-in database support
- Auto configuration with application.properties
- Supports MySQL, PostgreSQL, Oracle, etc.

# Key Dependencies (pom.xml)

- Spring Boot Data JPA:
  - enables integration between Spring Boot and the Java Persistence API (JPA) for ORM (Object-Relational Mapping).
  - Handles all database mapping, query execution, and transaction management behind the scenes.
- Database connector:
  - allows Java applications to talk to a specific database type — in this case, **MySQL**.
  - It's the "bridge" between your Spring Boot app and your actual database.
- Spring Web:
  - This enables your Spring Boot app to run as a **web application** and expose RESTful APIs.
  - It lets your backend expose API endpoints and communicate over the web (HTTP).

# Database Configuration (application.properties)

```
spring.datasource.url=jdbc:mysql://localhost:3306/secondDB\
  ?createDatabaseIfNotExist=true
spring.datasource.username=root
spring.datasource.password=
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
```

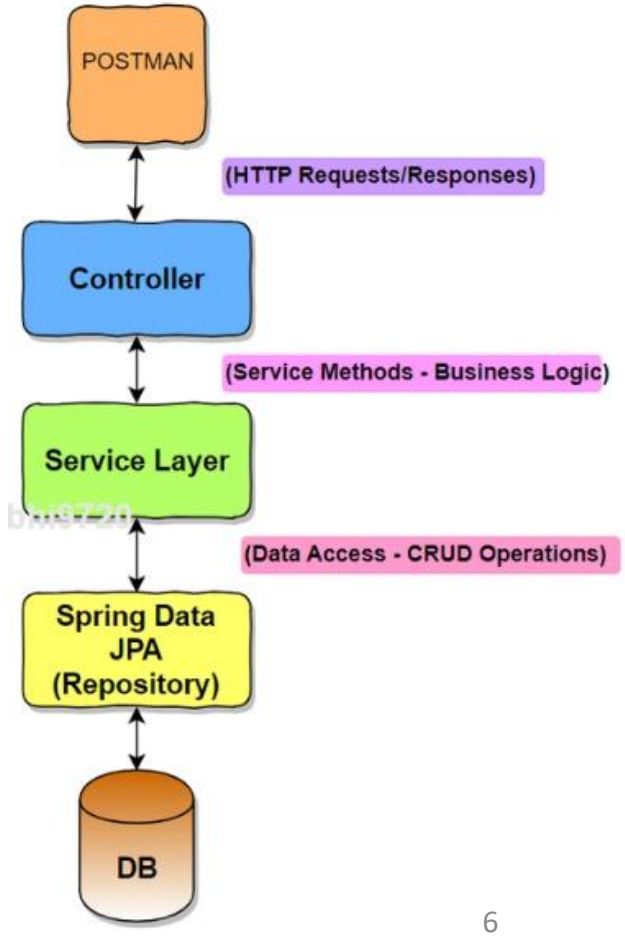
# Database Configuration

- spring.datasource.url:
  - Specifies the database **connection URL** (location and name of the database). It tells Spring Boot *which database to connect to*.
- spring.datasource.username:
  - Provides the **username** used to log in to the database.
- spring.datasource.password:
  - Provides the **password** for the above database user.
- spring.jpa.hibernate.ddl-auto:
  - Controls the **automatic schema generation** behavior. **updates** schema without dropping data
- spring.jpa.show-sql:
  - Enables **logging of SQL statements** in the console to show the queries
- spring.jpa.properties.hibernate.dialect:
  - Ensures that generated SQL matches the database type (e.g., MySQL, PostgreSQL).

# Flow of Data

1. Frontend sends HTTP request : */api/v1/employees*
2. Controller receives and forwards to Service
3. Service processes logic and calls Repository
4. Repository queries Database
5. Data flows back to Frontend via Controller

- Frontend → Controller → Service → Repository → DB



# Summary

- Spring Boot simplifies DB operations using JPA
- Minimal configuration required
- Layered architecture improves modularity
- DTO ensures clean data transfer