Day 4 Revisit

JPA – Java Persistence API

Persistence – Is a way of storing the state of an object to a database entity. (Table)

Serialization – Is the process of storing state of an object to a flat file.

JPA is a Specification (Persistence is the important class in JPA)

All the classes and interfaces of JPA are defined inside **javax.persistence** package.

JPA Implementations

1. Hibernate
2. iBatis
3. EclipseLink
4. Spring Data JPA (It uses hibernate)

JPQL – Java Persistence Query Language (This helps to write Database independent Queries)

JPQL query always represents the entity bean class name only not the database table.

Internally JPA also uses JDBC API.

Criteria, Writing Complex Queries, NativeQuery, NamedNativeQuery.

<https://spring.io/projects/spring-data-jpa#learn> Spring Data JPA Docs

Spring REST – Spring REST is a way of creating RESTful Web Service.

REST – Representational State Transfer – It re-uses http methods to perform CRUD operation and to create End points.

Web Service is also called as SOA (Service Oriented Architecture)

Two types of Web Service

1. SOAP based
2. REST based

SOAP – Simple Object Access Protocol (XML)

REST – Representational State Transfer (JSON)

SOAP uses XML based configurations.

WSDL (pronounced as wizdel) Web Service Definition/Description Language

Jackson – This dependency in Springboot application handles JSON resquest, respone etc.,

Some Important annotations of Spring REST

@RestController – This annotation converts the POJO class into a Front Controller class

Controller class will act as a Controller in MVC design pattern

Controller will connect front end with back end through database,.

MVC

M – Model (Represents the data in Entity bean objext)

V – View (Template./ User Interface

C –Controller (Back end code for business logic)

MVC in SpringBoot

Model – Entity Bean class [@Entity annotations )

View – Front End code (Components,

Controller - Disptcher Servlet/ Front Controller

Folder Structure of REST ful web service

1. Base\_package
2. Base\_package + Entity bean class
3. Base\_package + Controllers
4. Base\_package + Service
5. Base\_package + Exception

Example SpringBoot Application folder structure

1. Src/main/java
2. Src/main/test
3. Src/main/java/dao
4. Src/main/java/service ()
5. Src/main/java/controller
6. Src/main/java/

MySQL (Community Editions)

Postgres Database (Community Editions)

Lombok -- <https://projectlombok.org/download>

Api/v1/employees?id=100 === @PathParameter

Api/v1/employees/100 ===@PathVariable

<https://www.enterprisedb.com/downloads/postgres-postgresql-downloads>

API documentation

Swagger – Open API Documentation

SpringFox Swagger

@EnableSwagger2

Add Swagger Dependencies in pom.xml

<https://www.techgeeknext.com/spring-boot/spring-boot-swagger3-example>

Open API = Specification (version 3)

<dependency>

<groupId>org.springdoc</groupId>

<artifactId>springdoc-openapi-ui</artifactId>

<version>1.5.9</version>

</dependency>

Swagger = Implementation Tool

Add “@OpenAPIDefinition” annotation to the starter class

This will create two end-points automatically

V3/api-docs

Swagger-ui/index.html