**Spring Framework and Spring boot**

Spring is an open source framework which provided lot of modules or layer to develop any type of application.

Spring core

Spring context

Spring MVC

Spring Dao

Spring ORM

Spring boot

Spring micro service

Spring cloud

Spring security

Spring MVC : Model View Controller

Spring MVC without taking help of Spring boot.

Spring boot = all spring modules + few annotation – no xml file + in build web server ie tomcat or jetty etc.

Spring MVC without taking the help of spring boot

Index.html or index.jsp ----🡪web.xml (inside this file we need to configure front controller --🡪 it will pass the request to spring configuration file (xml or Java classes). --🡪 pass controller class @Controller or @RestController

Write more than one method with annotation as @RequestMapping or @GetMapping or @PostMapping etc. and base upon path we can redirect to specific view.

View and Controller

IOC and DI :

Inversion of Control : it a design pattern or programming pattern. Allow to create by container. If container will create it create properly as well as maintain properly.

We have to pull, use it and leave it. Any resources like database object, file resource, normal class object. IOC is a concept.

Dependency Injection : DI is an implementation of IOC.

Type of DI

1. Constructor base DI
2. Setter base DI

@Component

class Employee { POJO

private int id;

private String name;

private float salary;

@Autowired

private Address add;

}

@Component

class Address {

private String city;

private String state;

}

We need to configure using xml file or using annotation.

@Controller : if we use then view must jsp or html

@RestController : if my controller is rest controller then my view can be any technologies like Angular, React, Python, Asp.net, Php etc.

In Spring Boot we can connect database using

1. Using JDBC
2. Using ORM (Hibernate or JPA)
3. Using Spring Data

Entity classes (Hibernate Or JPA or Spring Data)

Cart

Wishlist

Admins

Categories

Orders

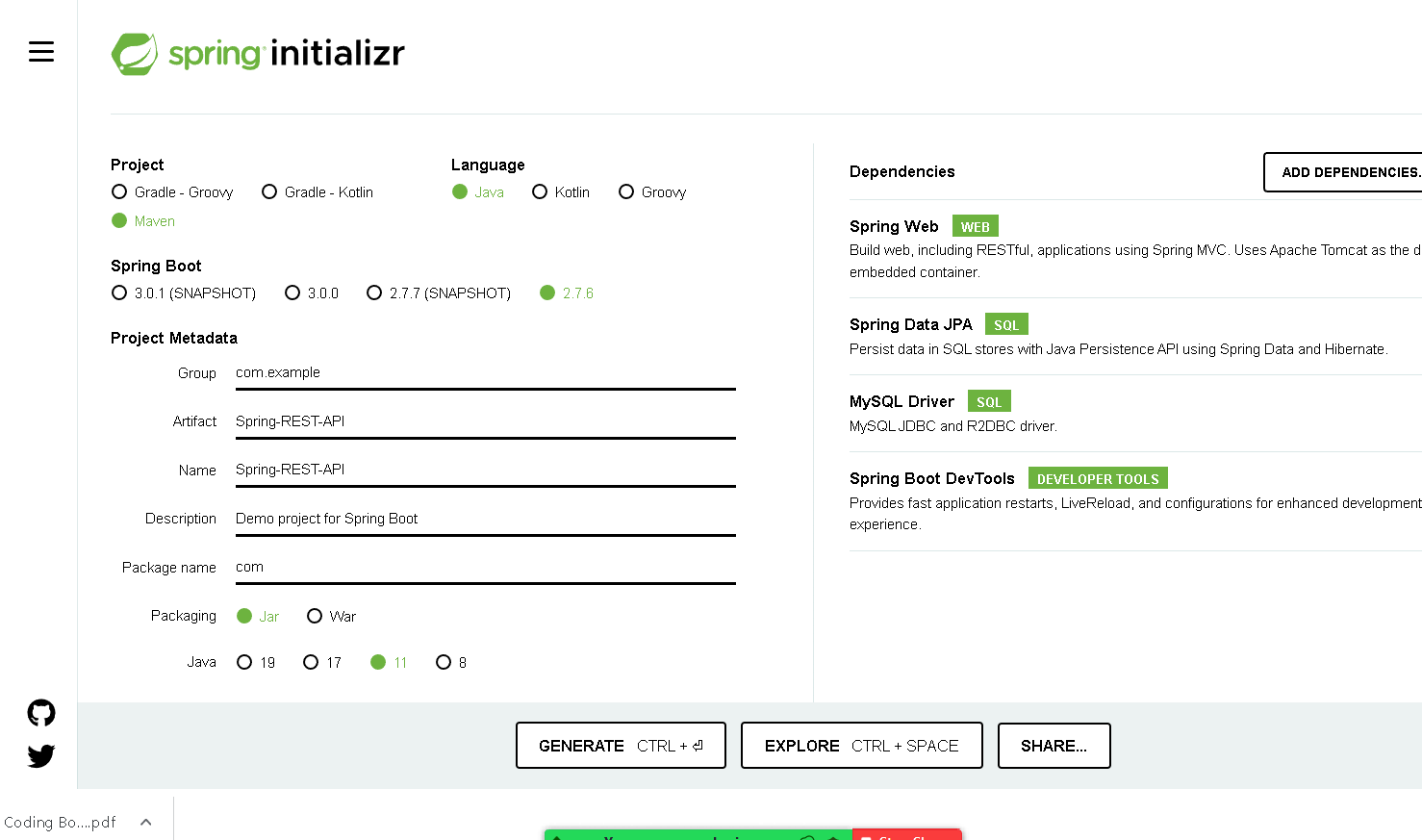
Products

Shipments

User

Using Spring initlizer we need to create the Spring boot project

<https://start.spring.io/>



Then add swagger dependencies

<dependency>

<groupId>org.springdoc</groupId>

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

<version>1.6.4</version>

</dependency>

Then create controller package, service package, entity package, repository package.

In Application.properties file add the database details

server.port=9090

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

spring.datasource.url=jdbc:mysql://localhost:3306/ecommerce\_db

spring.datasource.username=root

spring.datasource.password=root@123

spring.jpa.hibernate.ddl-auto=update/create-drop

Product.java

This contains all variable setter and getter method

With minimum @entity and @Id annotation

ProductRepository.java

We need to create normal interface and that interface extends JpaRepository<EntityClassName,DataTypeOfPrimaryKey>

ProductService.java

Do auto wired for repository and call method belongs repository depending upon our requirements.

Productcontroller

Inside that class we need to auto wired for service class and create more than method with @GetMapping, @PostMapping, @DeleteMapping, @PutMapping @PatchMapping etc.

Check these all rest api using swagger ui

<http://localhost:9090/swagger-ui/index.html>

Public class Demo extends HttpServlet/GenericServlet {

doGet(req,res)

doPost

doDelete

doPut

doPatch

}

ORM : Object Relation Mapping

JDBC limitation

Object Relation

@Entity

Class Employee { Employee

@Id Id,Name,Salary

Int, name,salary

}

Mapping

Employee -🡪Employee

Id ID PK

Name NAME

Salary SALARY

Using xml old version

Using annotation version

JPA Java Persistence API . JPA is a technologies part of java people

JPA is known as specification

Hibernate : Hibernate is a framework. Hibernate is known as implementation of jpa.

Servlet , jsp and Hibernate /JPA

Spring MVC with Hibernate /JPA

Spring boot with JPA

Spring jpa starter

Product p = new Product();

p.setId(100);

p.setName(“TV”);

p.setPrice(55000);

Products pp = new Products();

pp.setId(100);

pp.setName(“TV”);

pp.setPrice(55000);

List<Products> listOfProducts = new ArrayList<>();

CORS : Cross Origin Resource Sharing : when two domain going to communicate to each other through browser Cors policy enable.

Angular running by default port number 4200

Spring boot running port number 9090(default is 8080)

Reactive JS : Observable to handle event of asynchronous event of data.

To load the data from the Observable we need to use subscribe method.

This method take 3 parameter

1st : to load the data one by one

2nd : if any error generated by loading the that then second parameter get called.

3rd : after loaded successfully third parameter get called.

httpClient.get(“”)

httpClient.post(“”)

httpClient.put(“”)

httpClient.delete(“”)

all method return type is Observable of type of data.

ArrayList al =new ArrayList();

al.add(10); // auto – boxing : converting primitive to object.

al.add(10.10); : Object

al.add(“ravi”);

Object obj = al.get(0);

Integer i = (Integer)obj; // down level type casting

Converting one data type to another data types.

Byte

Short

Int

byte a=10;

short b =a;

short c = 100;

byte d = (byte)c;

class A {

dis1();

}

class B extends A {

dis1();

dis2();

}

A obj1 = new B();

obj1.dis1();

obj1.dis();

B obj2 = (B)obj1; down level type casting

obj2.dis1();

obj2.dis2();

**12-11-2022**

Frontend -🡪 Angular Framework

Template --🡪 component --🡪 service -🡪 HttpClient which provide get, post, put, delete, patch method to call backend technologies rest api.

We are consuming and produce data in json as well as text format.

Backend 🡪 Spring boot entity entity

Controller -🡪 Service -🡪 Repository -🡪 application.properties file -🡪 connect to data to store, retrieve, delete and update.

Database 🡪 MySQL

Table

SQL

Select \* from user; user is table name and it is not a case sensitive

JPQL

Select u from User u; User is entity class and u is a reference or object.

npm install @angular/cli

or

npm install @angular/cli –g

ng --version

monolithic and micro service

login module, customer module, product module, user module , category module.

Micro service : small service : we create login module using spring module or using any other technologies and we deploy independently.



To achieve micro service in spring framework. Spring boot provided two module ie spring cloud and spring micro service.

Spring framework provided one of the open source eureka server which help to deploy more than one micro service.

