Day 13 Revisit

Spring Boot – Opinionated Framework build on top of Spring Framework

Spring Boot – Simplifies the process & configurations of Spring Based Applications

3 ways of Spring Boot Project

1) Using Spring Initilizr (start.spring.io)
2) Using any IDE (STS/VS Code/Theia)
3) Using SpringBoot CLI

Model, Repo, Service, Controller, Exception, Util, Config

Web Service –

Types of Web Service =➔ SOAP (Simple Object Access Protocol), REST (Representational State Transfer)

Terminologies

1) End Points, API End points, REST end points
2) URI (Uniform Resource Identifier)
3) Status Code (200-success, 201-created, 404-resource not available, 500-server error)
4) RequestBody & Response Body
5) Pay Load (Data that we adding to the request body object)

Annotations used in Spring Boot

1) @SpringBootApplication [@ComponentScan, @EnableAutoConfiguration, @SpringBootConfiguration]
2) @RestController
3) @RequestMapping (Generic for all http methods)
4) @GetMapping
5) @PostMapping
6) @PutMapping
7) @DeleteMapping
8) @RequestBody & @ResponseBody
9) @PathVariable
10) @Entity
11) @Id
12) @GeneratedValue
13) @Repository
14) @Service

Ways of Configuring SpringBoot

1) Using Application.properties or application.yml file
2) Use Java class annotated with @Configuration
3) Adding dependencies in the pom (while &/after creating the app)

Usage of H2 database (In-memory database 2mb in size) – It's available only during the runtime.

Default config of h2 is

Db url : jdbc:h2:mem:DemoDB

Driver com.h2.Driver

Username SA (case in-sensitive)

Password Nil

MySQL, DevTools, Lombok, Spring Data JPA, h2

Various End points

API naming convention api/v1/employees (GET & POST)

Api/v1/employees/{id} (Put, Delete, Get by id)

CRUD using MongoDB

Testing End points

1) Using PostMan
2) Using CURL
3) Using Swagger (API Documentation)

Day 14 Agenda

Introduction to Spring Cloud, Eureka, Circuit Breaker, Hystrix, Open Feign

Hystrix

- Knowing what is circuit breaking
- Setting up a micro service application
- Apply circuit breaker pattern

Eureka

- Locating services
- Load balancing
- Knowing failovers of middle tier

- Client-side balancing

Spring Cloud

- Monolithic Architecture (Tightly coupled)
- Micro-service based Architecture (SOA – Service Oriented Architecture)

Eureka Server – (Service Registry/Discovery)

Cloud Service Providers

1) AWS (Amazon Web Service)
2) GCP (Google Cloud Provider)
3) Azure (Microsoft Cloud)

https://www.geeksforgeeks.org/java-spring-boot-microservices-develop-api-gateway-using-spring-cloud-gateway/ - API Gateway Demo

API Routing

MicroService – Communication between services

https://github.com/syskantechnosoft/MicroServiceBatch1/tree/main/Day%208%2021-Nov-2022

reqres.in/api/users