Week 1 Revisit

* RDBMS, ORM, JPA
* CRUD (JDBC, JPA)
* Java 8 Features (Functional Interfaces, Lambda, Method Reference, Streams)
* Version Control System (Git & Github)
* Web Service, SOA, Design Patterns,
* Spring Framework, IoC, DI, Spring Core, Spring Web, Spring Data, Spring Security, Spring Test, AOP etc.,
* Spring Boot, Different ways of creating Spring Boot based projects
* IDE (Eclipse EE IDE & STS), Postman, MySQL Server 8.0.x, Git-scm,
* Monolith VS Micro-Service

Agenda

* Spring Security
* API Documentation
* Another use-case of micro-service using some public API.
* Logging & Actuator

Eureka Discovery Server – API Gateway/ Registry Server which registers all the eureka client applications.

Add Eureka-Server dependency in spring boot project, add @EnableEurekaServer annotation to Starter class, add the following properties in application.properties file.

Web – Service – Types of Web-Service ( SOAP & REST)

SOAP – Simple Object Access Protocol (XML payload, WSDL – Web Service Definition/Description Language)

REST – Representational State Transfer (It Re-uses Http methods)

CRUD Operations

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sl No** | **Http Method** | **Method Name** | **DB Operation** | **PayLoad (Argument)** | **Return Type** | **DB Query** |
| 1 | Get | findAll() | Read All | No | List<Entity> | Simple Select Query |
| 2 | Get | findById(int id) | Read One | Primary key of Entity | Entity | Select query with where condition using pkey value |
| 3 | Post | Save(Entity obj) | Create/ Insert | Entity Object (W/o ID) | Entity (with ID) | Insert Query |
| 4 | Put | Save(int id, Entity Obj) | Update | pKey & Entity Object | Void/ No. of rows affected | Update Query |
| 5 | Delete | deleteById(int id) | Delete | Pkey | Void/No. of rows affected | Delete Query |

@SpringBootApplication = [@EnableAutoConfiguration + @SpringBootConfiguration + @ComponentScan]

Spring = Loosely Coupled Enterprise Java Application (Spider Man Toy using Lego blocks)

Spring Boot = It’s a way of creating Spring based Enterprise Applications

(Spring Initializr (start.spring.io), Using STS, Spring Boot CLI – Command Line Interface)

Monolith VS Micro-Service

Monolith Application == It’s considered as Big Mud Ball

Micro-service = Small Sweet Gems

Client – Server

* Client will a Send a Request Object
* Server will receive the request, validate it and create a response for each request
* If the request is valid it will create expected response by executing some server side code
* If the request is invalid, it will create error response
* By examining the status code of the response object we can find out that the operation status.
* If the status code is 200/201 – then it means success
* If the status code is 40x – then it is error (404-Not found, 403-UnAuthorized)

Amazon.in – Get

Any Web application

1. Register Page (First Time user Register using email/mobile) – POST
2. Returning users (Use email/Mobile) login to the application – POST
3. Add products to cart – Put
4. Buy products - Put
5. Make Payments -
6. Mange Orders
7. Track Orders
8. Rate the Delivery &/ Order
9. Write reviews on purchased products
10. Answer questions related to purchased products
11. Logout

DB Operations – CRUD [ Read / Modify (Create/Update/Delete)]

List of public APIs

<https://apilist.fun/>

<https://mixedanalytics.com/blog/list-actually-free-open-no-auth-needed-apis/>

<https://any-api.com/>

<https://rapidapi.com/collection/list-of-free-apis>

<https://github.com/public-apis/public-apis>

<http://localhost:8082/movies/200>

<http://localhost:8083/ratings/user/200>

<http://localhost:8081/catalog/200>

Spring Security

Securing your End points

Restricting Access (Edit & Delete access based on user role) and Providing Access (able to buy – only logged in user)

Authentication & Authorization

Authentication – Is the process of verifying the user’s credentials (username/email & password, mobile&otp etc.,) – ensuring that the user is already registered.

Authorization – Is the process of checking whether the currently logged-in user has the permission to perform the given operation or not.

Adding Spring Security as a dependency in spring boot will do

1. Add two end points namely “/login” &”/logout”
2. Also creates a login form for login end point.
3. When “/logout” URI is entered it will ensure the Logout operation from the user.
4. It also configure a userdetails object and auto-generate a password for default user, that will be displayed in the console.

CSRF - Cross Site Request Forgery

CORS = Cross Origin Resource Sharing

<https://www.codeusingjava.com/boot/db>

JWT – JSON Web Token

Token based Authentication & Authorization

JWT contains 3 parts (header+Payload(data)+Signature)

Encrypted with the help of Hashing Algorithms ( RSA, PSA, HSA, PSA 256/384/512 bit)

It will generate the token, /authenticate

Register, login end point, (POST)

Read (Get)

Accessing Public APIs (Using Different Micro-service) Communication between micro-service using RestTemplate.

Spring Security

JWT (Authen & Autho)

Examples





