Agenda

MicroService

What?

Why?

Adv/Dis-Adv

SOA

MSA

Monolith & MSA

Terms used in MicroServices

API Documentation & Swagger

Let’s consider a Online-Shopping Application ( Amazon.in)

1. Users Mgmt Module (Signup/ Sign-in, Update Profile)
2. Product Mgmt Module ( Add/Update/ Deleting/ offers)
3. Shipping Module (tracking delivery)
4. Payment Module ( card/upi methods)
5. Rating Module (Feedback, comments, ratings)
6. Category Module

Monolith Application – Application which contains n number of modules.

SOA – Service Oriented Architecture

Web Services – Communication between Electronic Devices using Web.

SOAP based – Simple Object Access Protocol – WSDL (Web Service Definition/Description Language) - XML based response.

REST based – Representational State Transfer (It uses http) – JSON (JavaScript Object Notation)

REST ful Web Service. With the help of http accessing a method.

public String hello() {

return “HelloWorld!”;

}

Terms used in Web Services

1. End-point
2. URI (Uniform Resource Identifier) api/v1/employees – Response will be in JSON format

Class Employee{

Private int id;

Private String name;

}

Main() {

Employee emp = new Employee(100, “ABC”);

}

XML Representation (In XML tags are user defined)

<employees>

<employee>

<id>100</id>

<name>ABC</name>

</employee>

<employee>

<id>101</id>

<name>XYZ</name>

</employee>

<employee>

<id>102</id>

<name>MNO</name>

</employee>

</employees>

JSON Representation

Emp = { “id”:100, “name”:”ABC”}

Employees = [

{ “id”:100, “name”:”ABC”},

{ “id”:101, “name”:”XYZ”},

{ “id”:102, “name”:”MNO”}

];

SOA – Service Oriented Architecture (WebService based Applications)

CRUD Based / Streaming Applications

C- Create/Insert Operation

R – Read Operation (Read All, Read One by Id)

U – Update Operation

D – Deletion Operation

Each Micro Service can be individually created, tested, deployed and modified.

1. User MicroService (Separate application)
2. Product MicroService ( S.A)
3. Shipping MicroService (S.A)
4. Payment MicroService (S.A)
5. Rating MicroService (S.A)
6. Order MicroService (S.A)

MSA – Micro Service Architecture

Why Web/Micro Service

Let’s consider ATM withdraw example

ABC bank’s Debit Card, ABC bank’s ATM (JAVA)

ABC bank’s debit card, XYZ bank’s ATM ( .Net)

Creating A Web Service --- Using SpringBoot along with JAVA is very easy and simplest way of creating web Service.

|  |  |  |
| --- | --- | --- |
| Sl No | Monolith Application | MicroService Application |
| 1 | Tightly coupled modules make maintaining difficult | Loosely Coupled. Easy to maintain |
| 2 | Complexity increases when many modules are added | Simple services are based on Business requirements and easy to manage |
| 3 | Changing the DB, updating Code makes entire application down | Changing a single micro-service will not affect other services. |
| 4 | Scaling up, Load Balancing will be challenging | Load Balancing and Scaling are easy |
| 5 | Very Complex deployment, it takes more time to build & test. | Build, test and deployment time are very less |
| 6 | Heavy Weight | Light Weight |
| 7 | Since it’s single monolith deployment no additional efforts needed | Needs some additional effort ( Service Registry & Discovery |
| 8 | IT’s suitable for less than 5 services | More than 5 Services |
| 9 | Single DB for the entire App | Different Databases ( MySQL, Postgres, MS-SQL) |

Terms used in Web Service

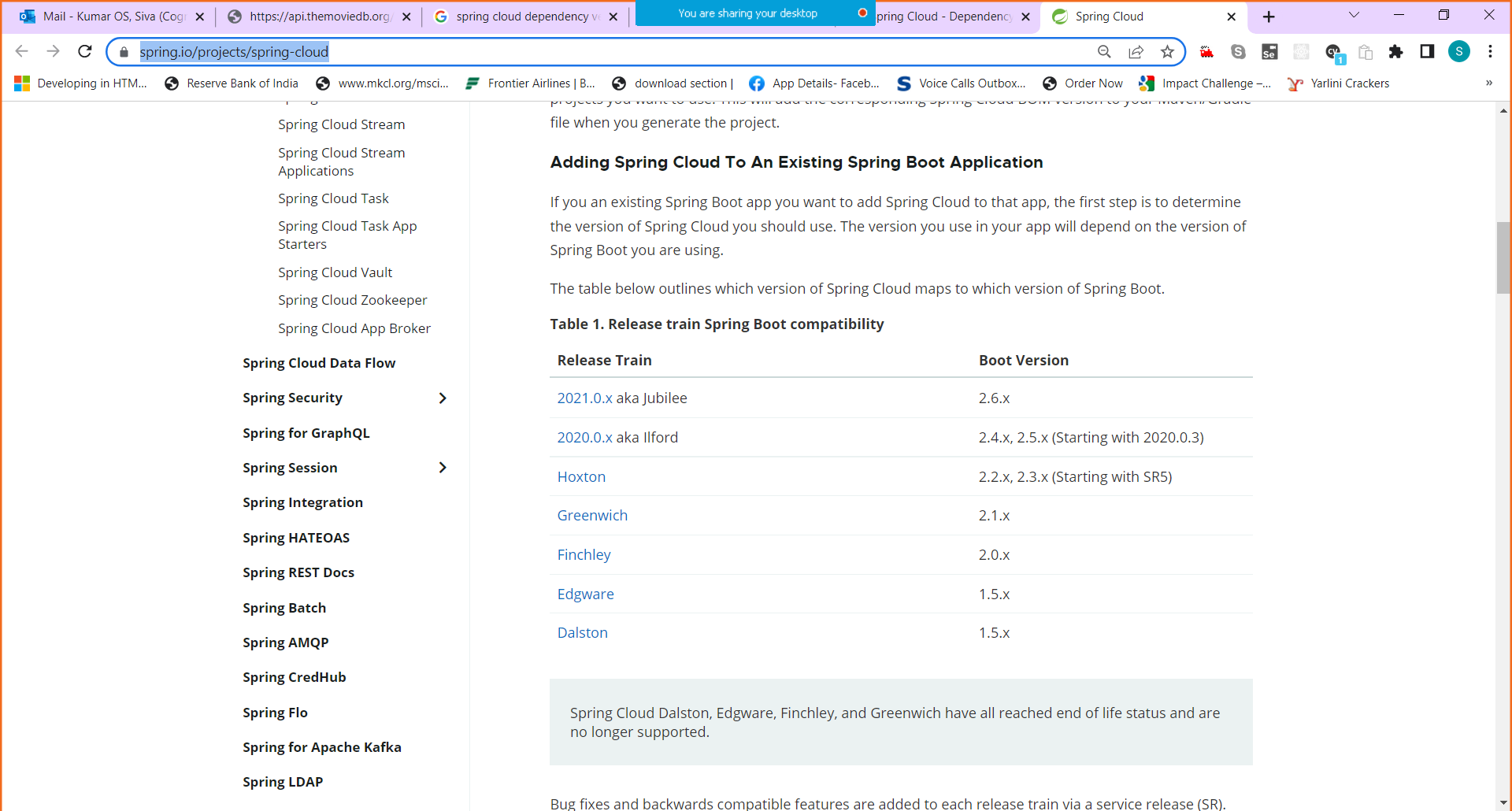
1. URI (Uniform Resource Identifier)
2. End-point ( N number of end points) – Request Mapping (Mapping a URI to a particular method in the controller) --- Generic Req Mapping [@RequestMapping] (for any http methods – get, post, put, delete…. Specific Mapping [ @GetMapping , @PostMapping, @PutMapping, @DeleteMapping
3. RestController – It’s a POJO class with @RestController annotation and it contains URI mapping.
4. Repository – It’s DAO (Data Access Object --- In-Memory[h2/derby]/Stand-alone database[oracle, postgres, DB2, MySQL, MSSQL) ---- Using this CRUD operation will be done.
5. Service - Connects controller with DAO

Many Web Service Applications (To Integrate – we need discovery Service --- API Gateway )

Eureka / Consul --- Discovery Service [Netflix OSS – Netflix Open Source Services]

Spring Cloud & Spring Boot Compatibility

<https://spring.io/projects/spring-cloud>



1. Creating Eureka (Discovery) Service (Server) [SpringBoot Dev Tools, Spring Web, Swagger/Open API annotation (optional) [External Dependencies]
2. MovieDetails ( Spring Boot Dev Tools, Spring Data JPA, h2,MySQL, Postgres, Eureka Discovery client)

<https://github.com/syskantechnosoft/microservice-demo>

