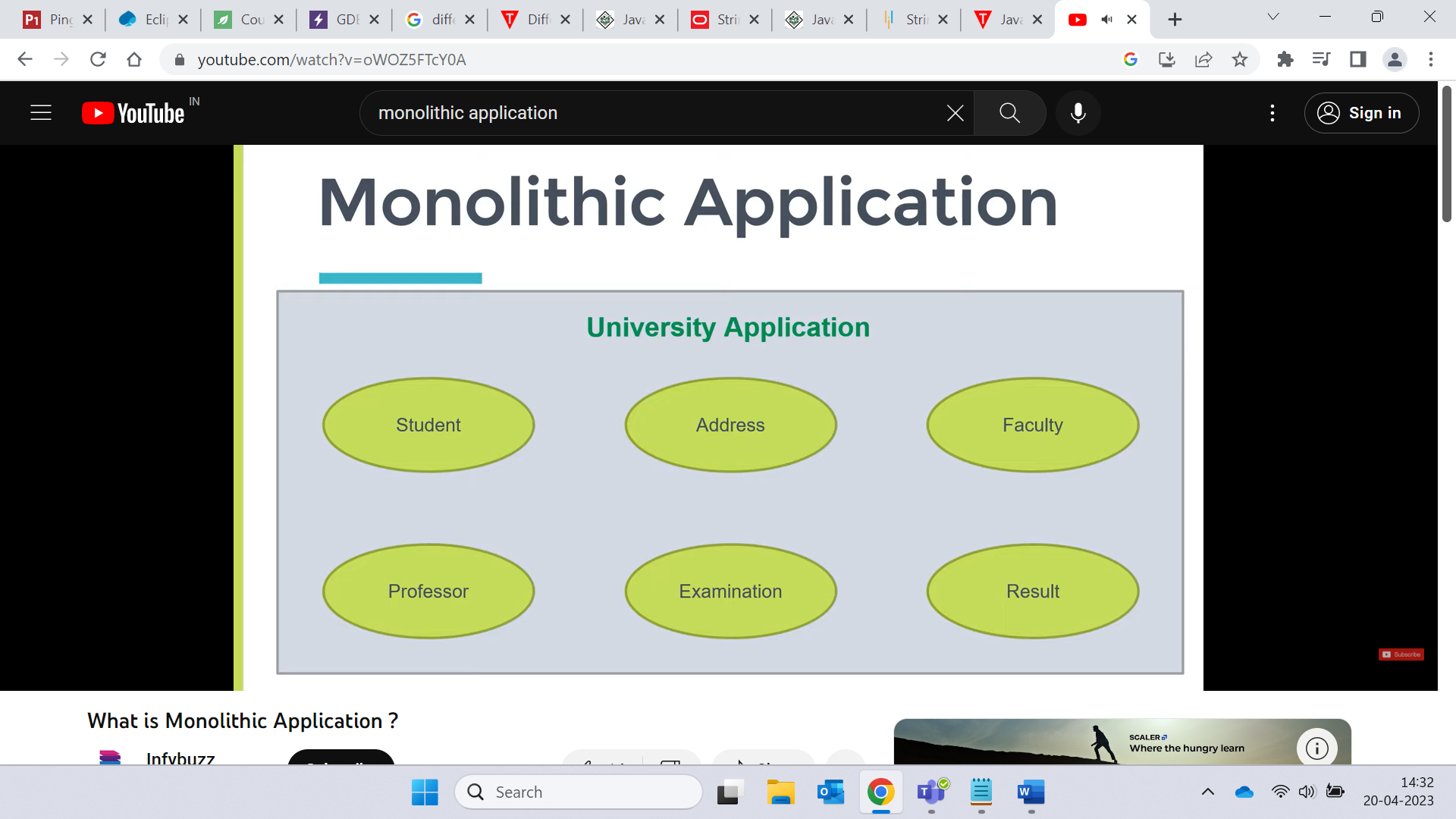
**Monolithic application 🡪** is an architectural style which structures the complete application into one executable component (use for small applications)

When we have multiple components in single application the we called it as Monolithic application

For ex. 🡪



🡪 all business logic on one machine

* Easy to maintain , test , deploy
* Mainly used for less than 3 entities

Drawback

* System becomes very complex as we have many components in single application
* If we do changes in one module then we need to test whole application.
* If we want to scale up one component for ex. During Examination the traffic on examination portal will be more so we need to scale up that component but in monolithic application we cannot scale the single component we need to scale all components because we have all the components in same application.
* Components are tightly coupled
* Can take longer time to develop
* High coupling in two modules
* No restriction on size
* Doing Changes are not easy
* Failure of one system cause failure of whole system where we use Microservices which is loosely coupled

**Microservices 🡪** small services / small spring boot applications (in context of spring boot)

-🡪 Monolithic application divide in small small spring boot applications called as microservice

* The components in Monolithic application become Microservice applications in Microservice.
* Advantages
  + As the services are different small applications it became easy to work on for teams
  + If we do changes in any service then we need to test only that service instead of testing whole service
* Disadvantages
* Complexity: Microservices can introduce additional complexity in terms of communication, deployment, and monitoring, which can require additional effort and resources.
* Testing: Testing and integration of microservices can be more complex than monolithic architecture.
* Increased Infrastructure: Each microservice will require its own infrastructure, which can increase operational costs.

Microservice architecture 🡪 small independently deployable service that work together, modelled around a business domain.

High cohesion 🡪 ex. Inheritance 🡪 reusability of code

* CICD
* Called through agile methodology

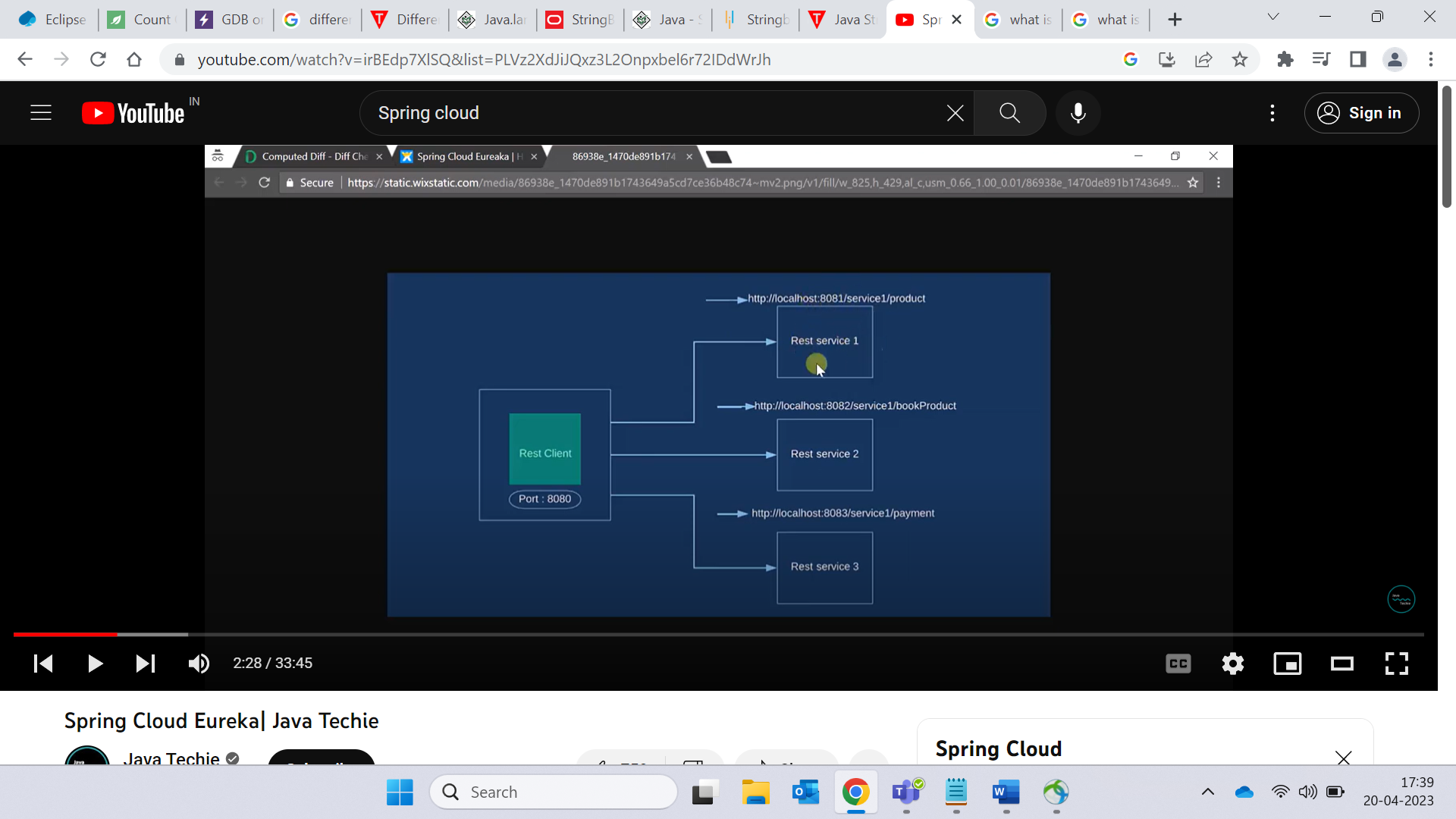
Challenge 🡪 breaking

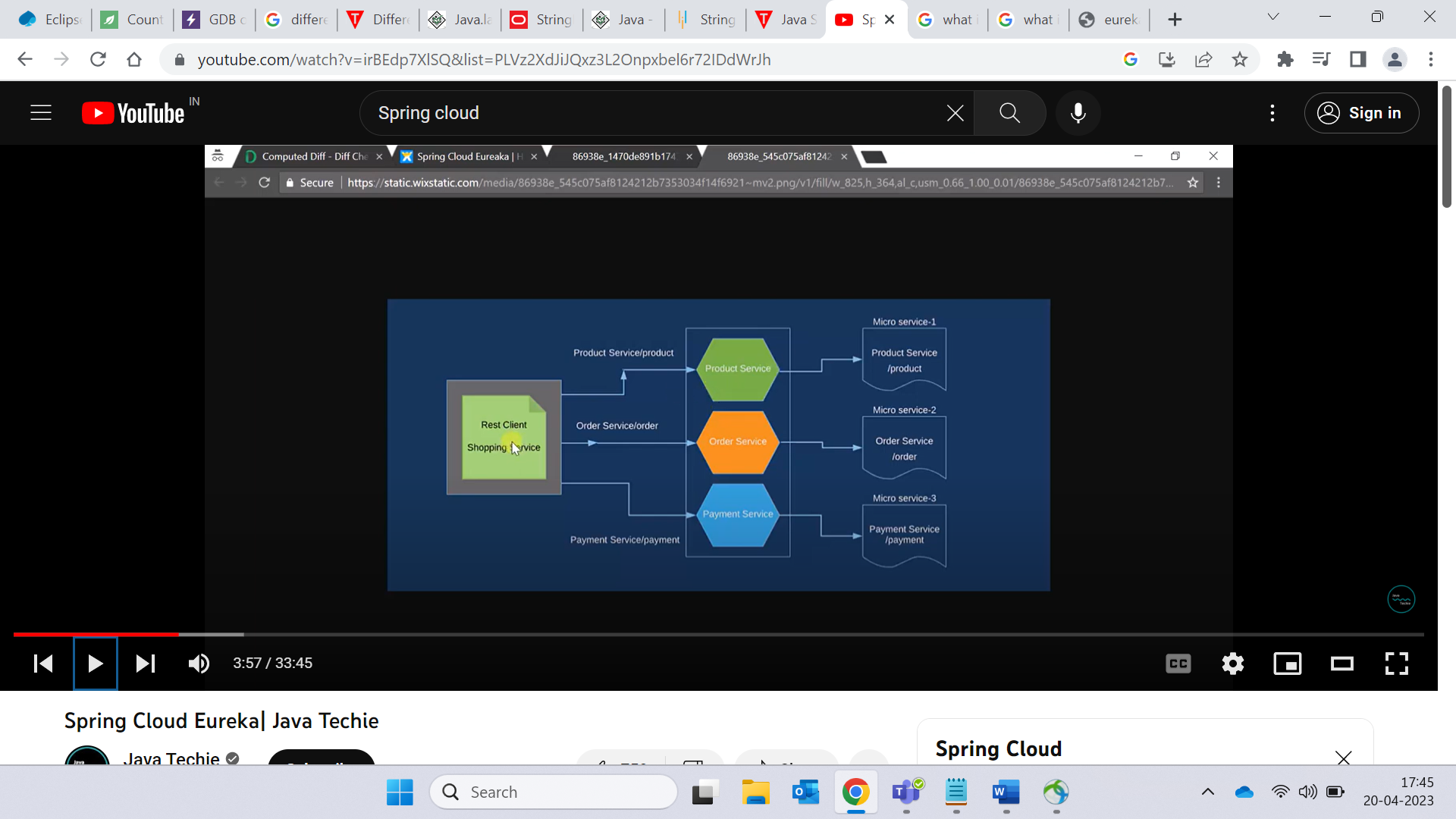
Spring cloud – features –the problem of discovery

Eureka service always run on 8761 port

**Eureka Server** 🡪 Eureka Server is an application that holds the information about all client-service applications. We can say it as service registry provided by netfix.

If our user want to use micro services for that there are different url (it includes different host name , portname , endpoint name etc) for each microservice which user cannot remember , so for this we can use eureka registry where we can register services and we can directly use services.





Server

**Components of Microservices**

**Service Registry** 🡪 is like a database which contains location of all available service. Also allow the services to communicate with each other.

**Load Balancer 🡪** a component that distributes the incoming requests across multiple instances of the same service to improve performance and availability.

**API Gateway🡪**  an entry point that manages the access to the microservices, routing the requests to the corresponding service and handling authentication, rate-limiting, and other cross-cutting concerns.

**Configuration Server 🡪** A configuration server is a central repository that stores the configuration settings for each service in the microservices architecture.

**Service Discovery 🡪** Service discovery is the process of locating available services in the microservices architecture. It involves querying the service registry to find the location and status of each service. This enables services to discover and communicate with each other dynamically. Service discovery can be performed using different techniques, such as client-side discovery or server-side discovery.

**Difference between Springboot and spring cloud**

**Springboot** is a tool that makes developing web applications and microservices easier with spring framework.

Spring boot have autoconfiguration capabilities i.e. no need to configure dependencies manually.

We can build standalone application with spring boot i.e the application that run on their own ,without relying on an external web server, by embedding web server such as Tomcat or Netty into your app during initialization process.

**Spring cloud 🡪** Spring Cloud is a set of tools and frameworks built on top of Spring Boot, designed to simplify the development of distributed systems and microservices-based applications. It provides features like service discovery, circuit breaker, load balancing, distributed tracing, and configuration management, making it easier to develop and manage large-scale applications.