WELCOME BACK
LESSON 8

Spring/Spring Boot Crash Course

For TD Bank

MEET YOUR CRASH COURSE TEAM

TANGY F.
CEO

HAL M.
DEVELOPER

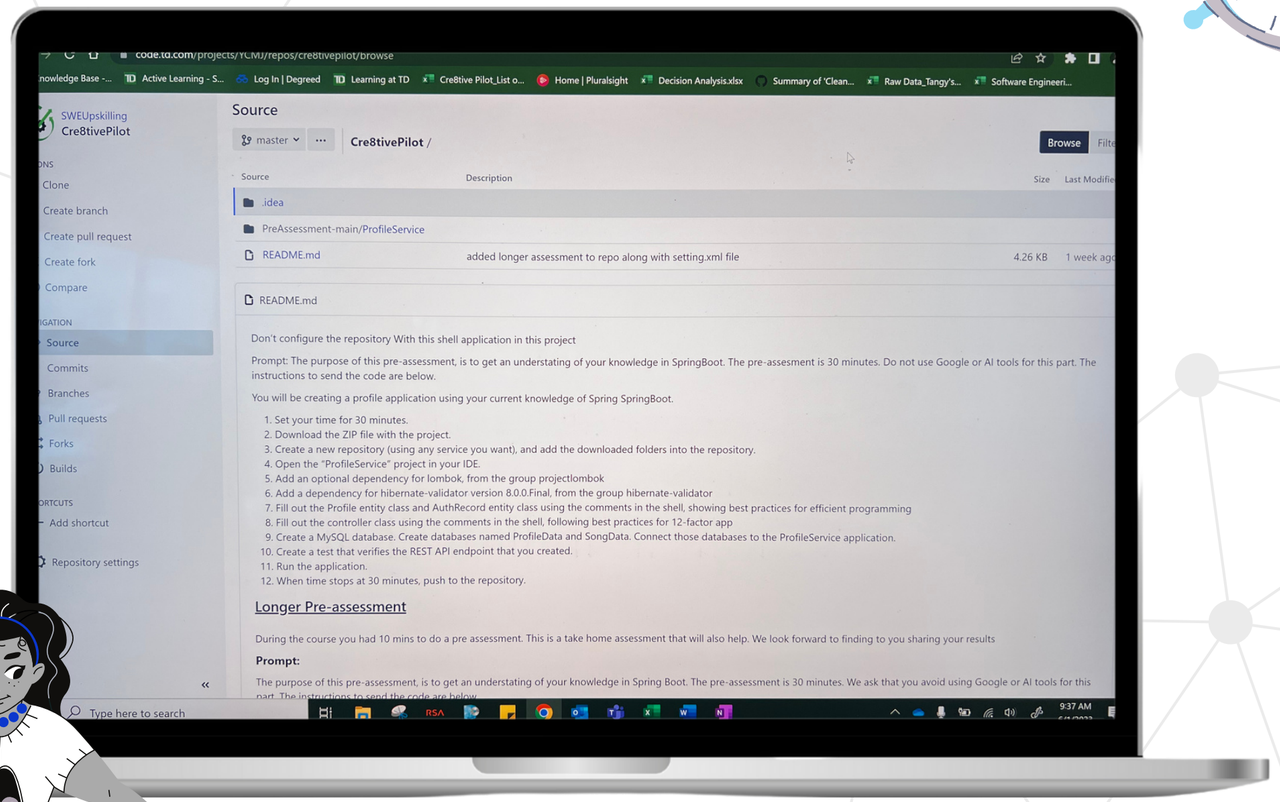
WILLIAM D
DEVELOPER

LESSON 8

AGENDA

1. Recap Lesson of 7
   >>Rapid review of Lesson seven
2. Current Lesson- Eureka
3. Q&A

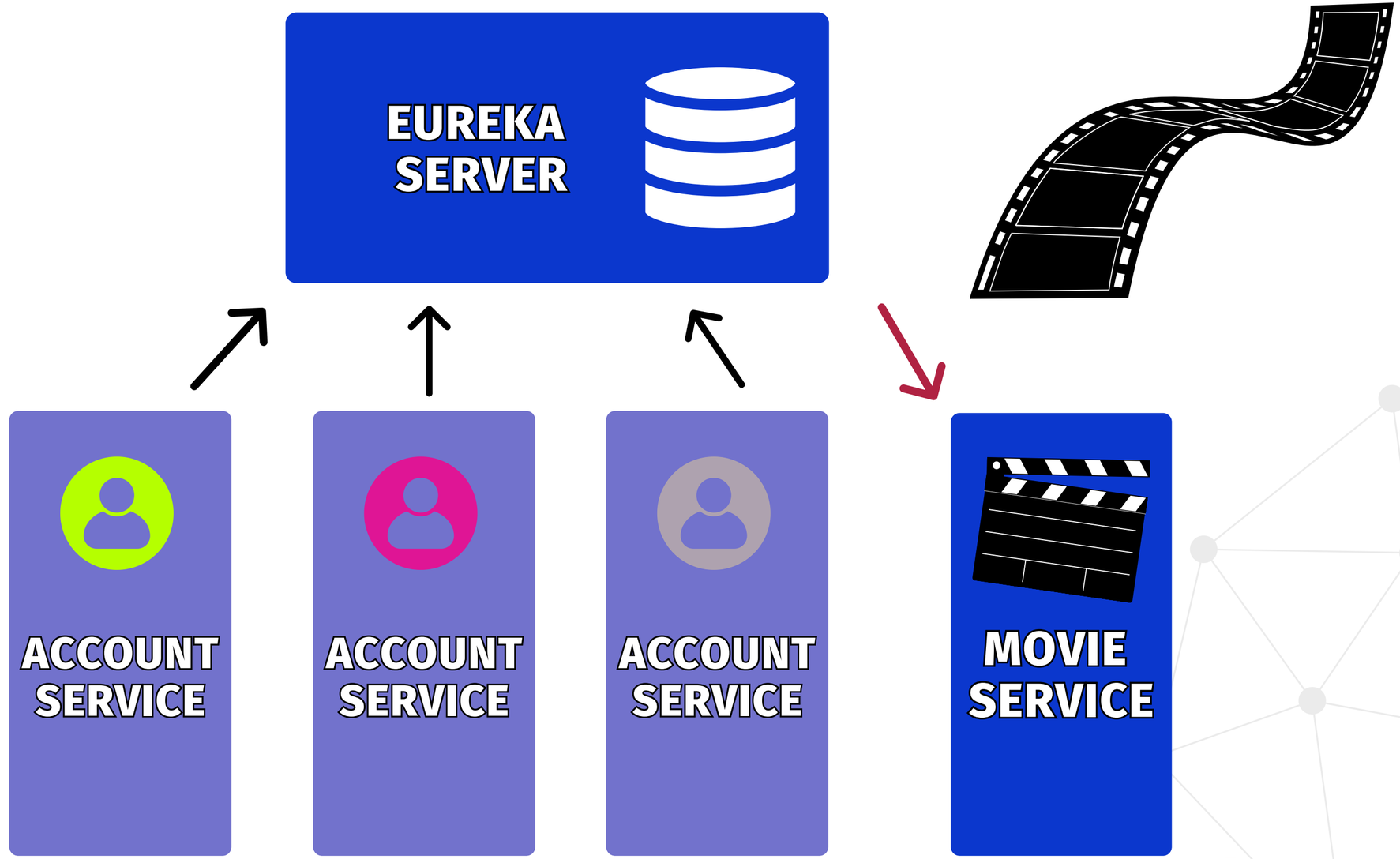
SETTING UP YOUR ENVIRONMENT



Cre8tive Devs. Software

WE ARE BUILDING

BITE SIZE
MOVIE REVIEW APP



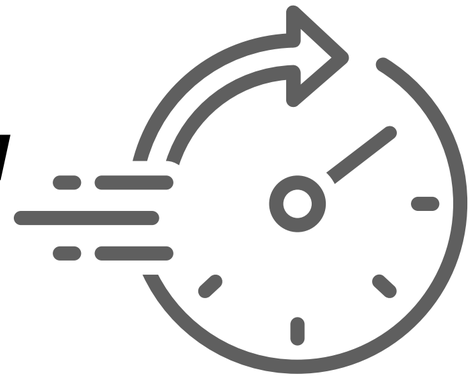
Creative Devs. Software

Rapid Review
of

LESSON 6&7

WELCOME TO LESSON 8

Rapid Review Trivia

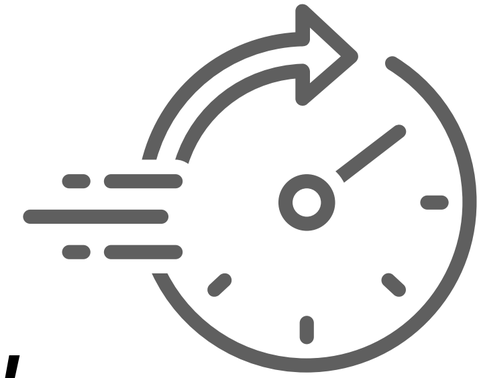


1.Name all seven Registered Claims defined by the JWT specification.

2.What is Mockito used for?

WELCOME TO LESSON 8

Rapid Review of LESSON 7



Rapid Review

Unit Testing & Hibernate

LESSON 8

Eureka Server

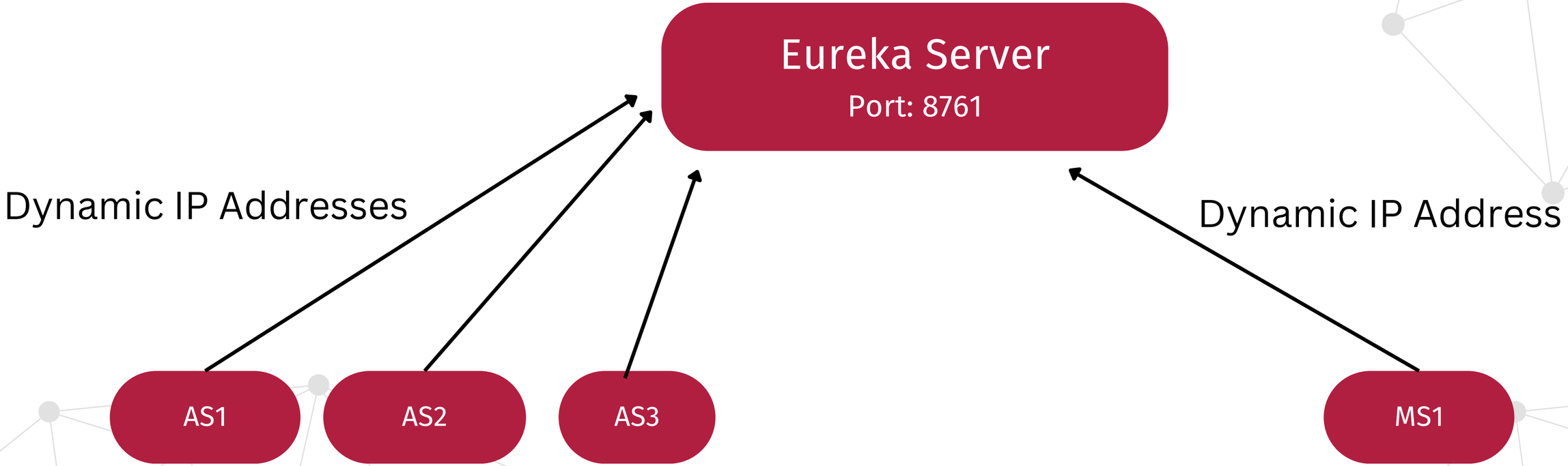
LESSON 8

EUREKA SERVER

1. What is Eureka Server and what are its components
2. Setting up the server and registering the Profile & Song services as Clients
3. Advantages/ Necessity of using Eureka Server

EUREKA

LESSON 8



Account Service Instances

Movie Service Instance

Reference
AS1 = Account Service 1
SS1 = Song Service 1

EUREKA SERVER

LESSON 8

Components of Eureka

- Eureka Server
- Eureka Client
- Load Balancing

Advantages

- Dynamic Scaling
- Load Balancing
- Health Metrics
- Self healing

Alternatives

- Apache ZooKeeper
- Kubernetes

12-FACTOR APP

Factor 3: Config

Watch for how we declare the ports for our services in the application.properties files

Factor 8: Concurrency

Take note of how we scale horizontally, how we run multiple instances of Account Service at once, how we use a distributed process manager to keep track of which processes are running at any given time.

CODING EXERCISE #1

Below is the beginning of the application.properties file inside our Eureka server. Write code that prevents our Eureka server from acting as its own client.

```
server.port=8761

//Code Goes Here
```

CODING EXERCISE #1

Below is the beginning of the application.properties file inside our Eureka server. Write code that prevents our Eureka server from acting as its own client.

```
server.port=8761
//Code Goes Here
```

eureka.client.registerWithEureka=false
eureka.client.fetchRegistry=false

CODING EXERCISE #2

Our Movie Service is intended to use our Eureka Server as a discovery service in order to connect with our Account Service clients, automatically choosing an Account Service client for each call to the discovery service. What is wrong with this configuration class in Movie Service?

```
@Configuration
public class BeanConfig {

    @Bean
    public ModelMapper modelMapper() { return new ModelMapper(); }

    @Bean
    public RestTemplate restTemplate() { return new RestTemplate(); }

}
```

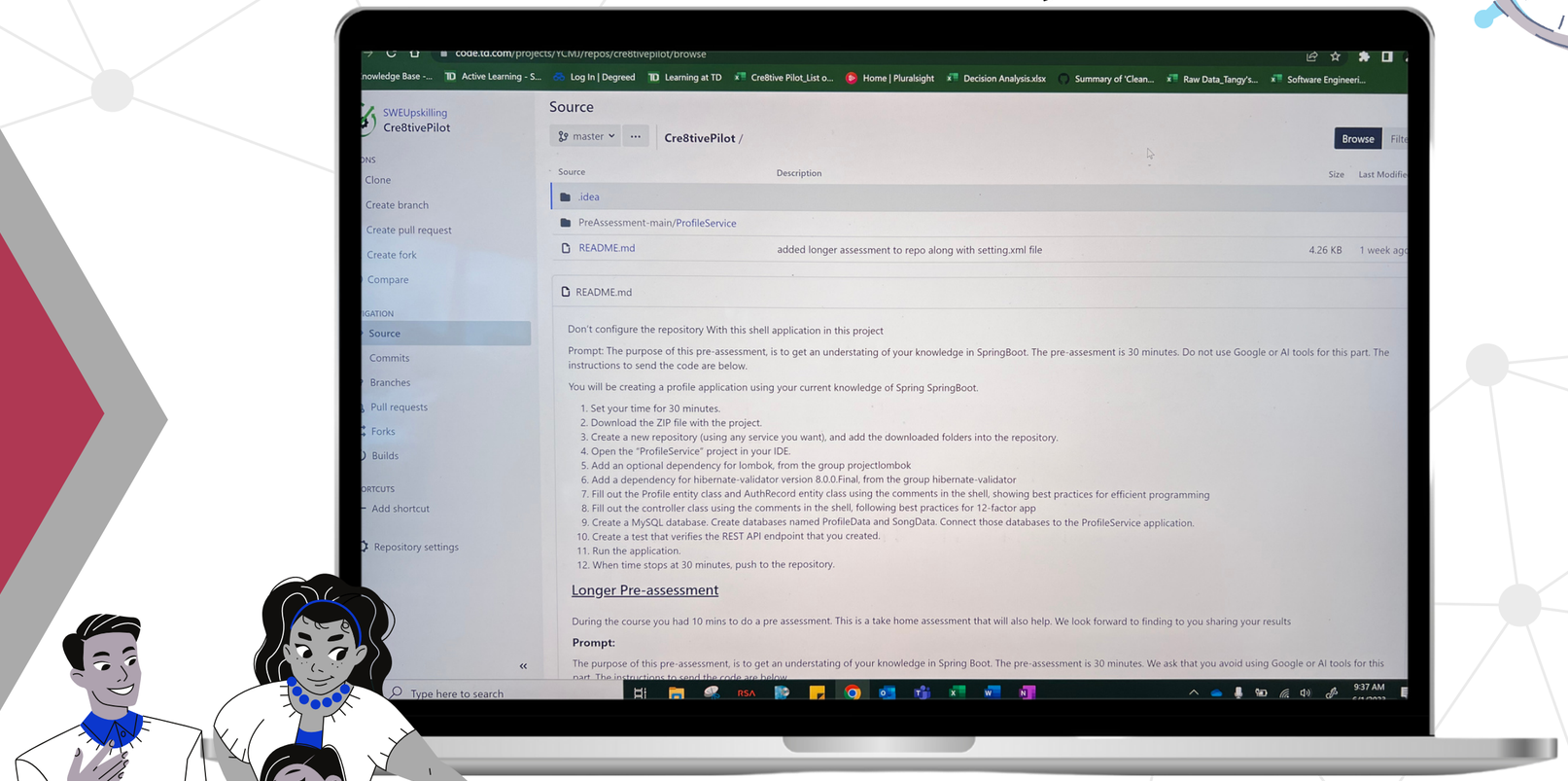
CODING EXERCISE #2

The RestTemplate is missing our load balancing, so we won't cycle between Account Service clients properly. The correct code is as follows.

```
@Bean
    @LoadBalanced
    public RestTemplate restTemplate() {
        return new RestTemplate();
    }
```

YOU CAN WORK ON YOUR HANDS ON PROJECT

HANDS ON PROJECT



Cre8tive Devs. Software

THANK YOU

Crash Course

We will see you on Monday