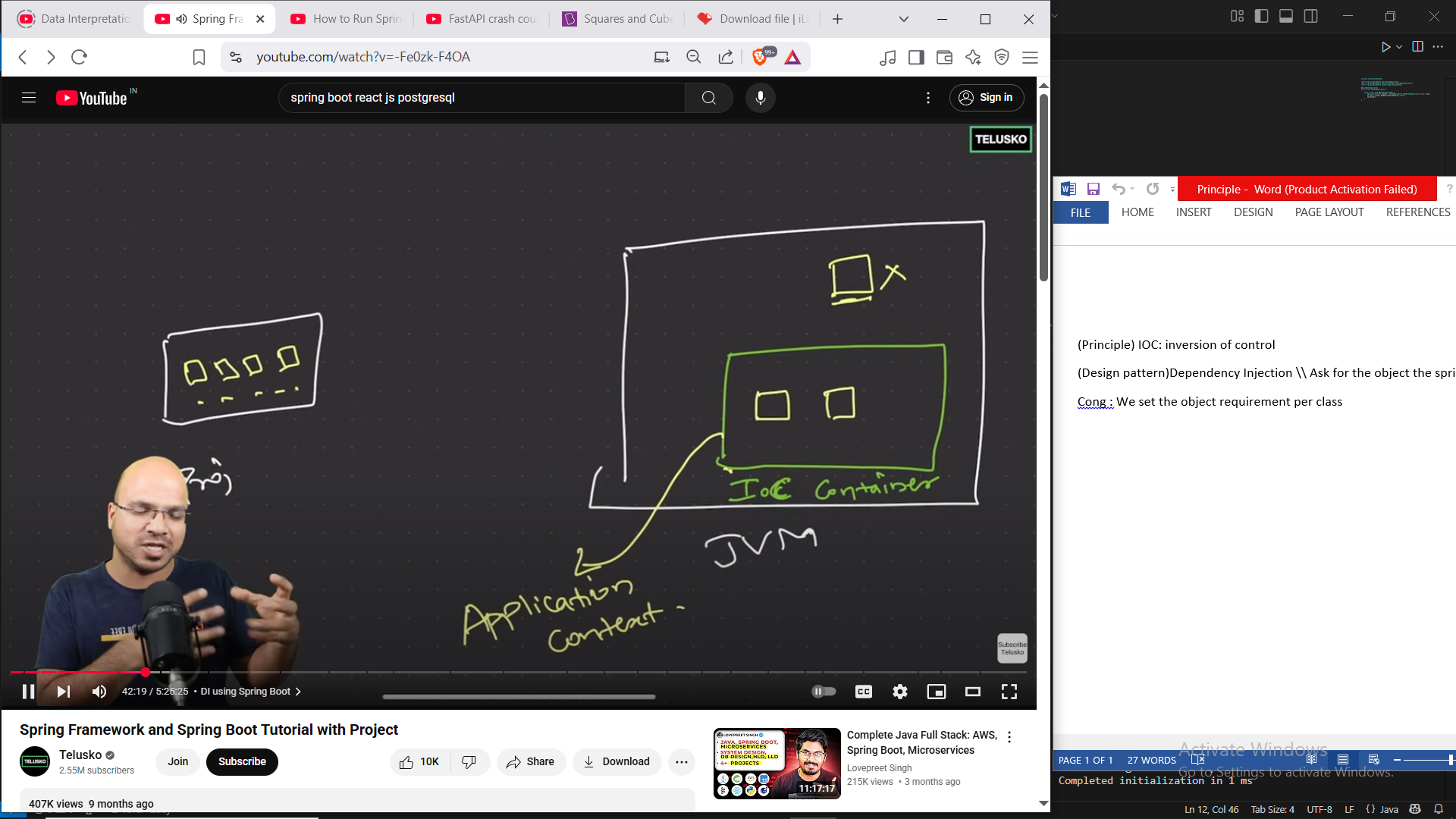
(Principle) IOC: inversion of control

(Design pattern)Dependency Injection \\ Ask for the object the spring will give you

Cong : We set the object requirement per class



**//Dependency Injection**

ApplicationContext context = Injects dependencies where needed

Getbean() => Which class object I want and if it exists then give

@componet => Make sure that this class object is manage by springboot

**//Autowireing (field injection**

**@Autowired**

**Private Laptop laptop;**

**or setter injection**

**@Autowired**

**Public void setLaptop(Laptop laptop){**

**this.laptop = laptop;**

**}**

**)**

**Contructor one is default**

If we want to connect two ioc container with @Autowired

@Primary = If namy options are there. Set the priority or we can use

@Qualifier(“laptop”)

**Spring framework**

**First we build the container by calling application context**

**But who will take care of our beans?**

**For that we have to make a xml file to call springboot**

**It will ask for the objects**

try (ClassPathXmlApplicationContext context = new ClassPathXmlApplicationContext("spring.xml")) {

            Dev obj = (Dev) context.getBean("dev");

            obj.build();

        }

**Setter method**

    <property name = "age" value = "20" />

**Construction**

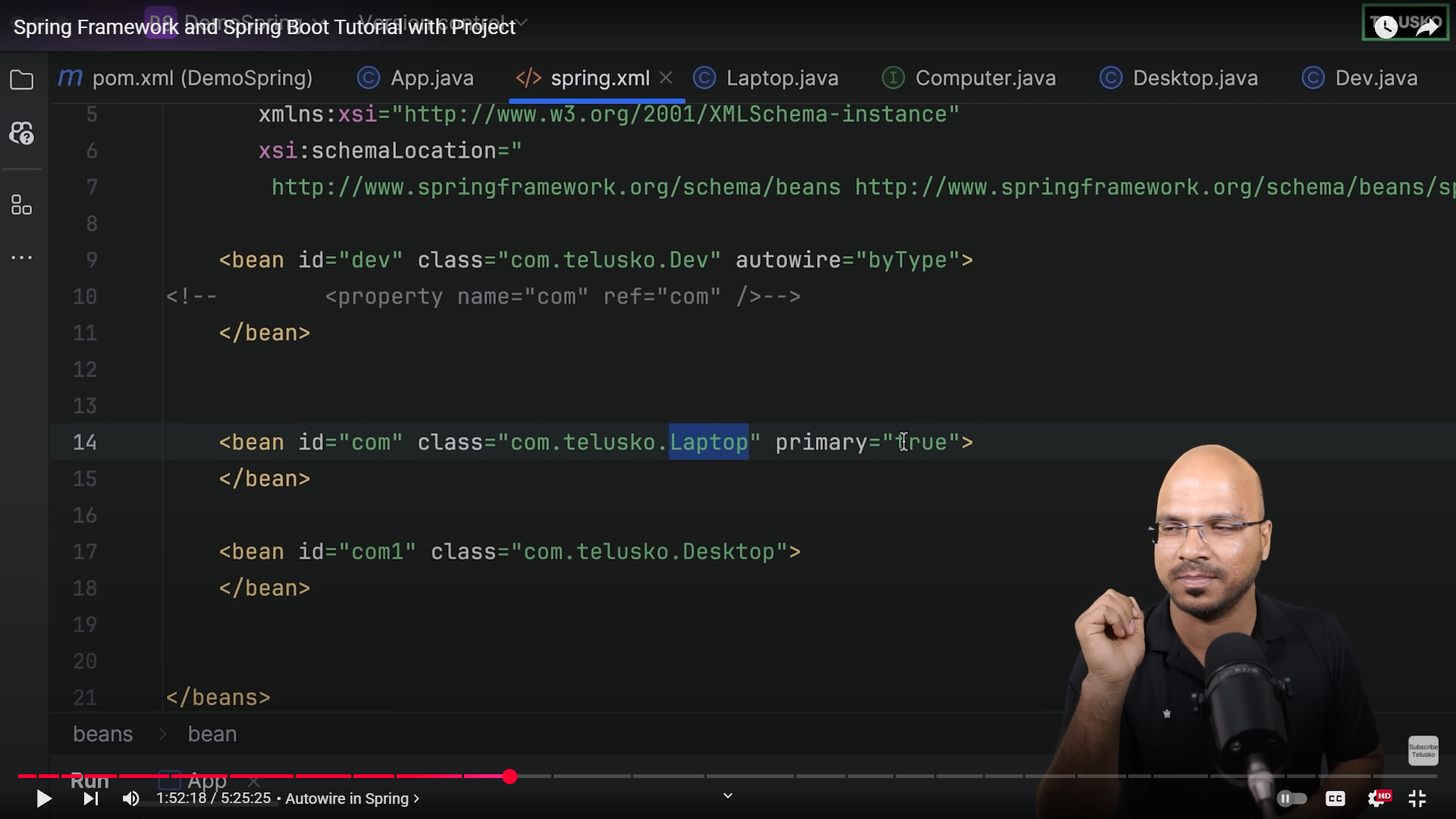
 <constructor-arg value = "20" />

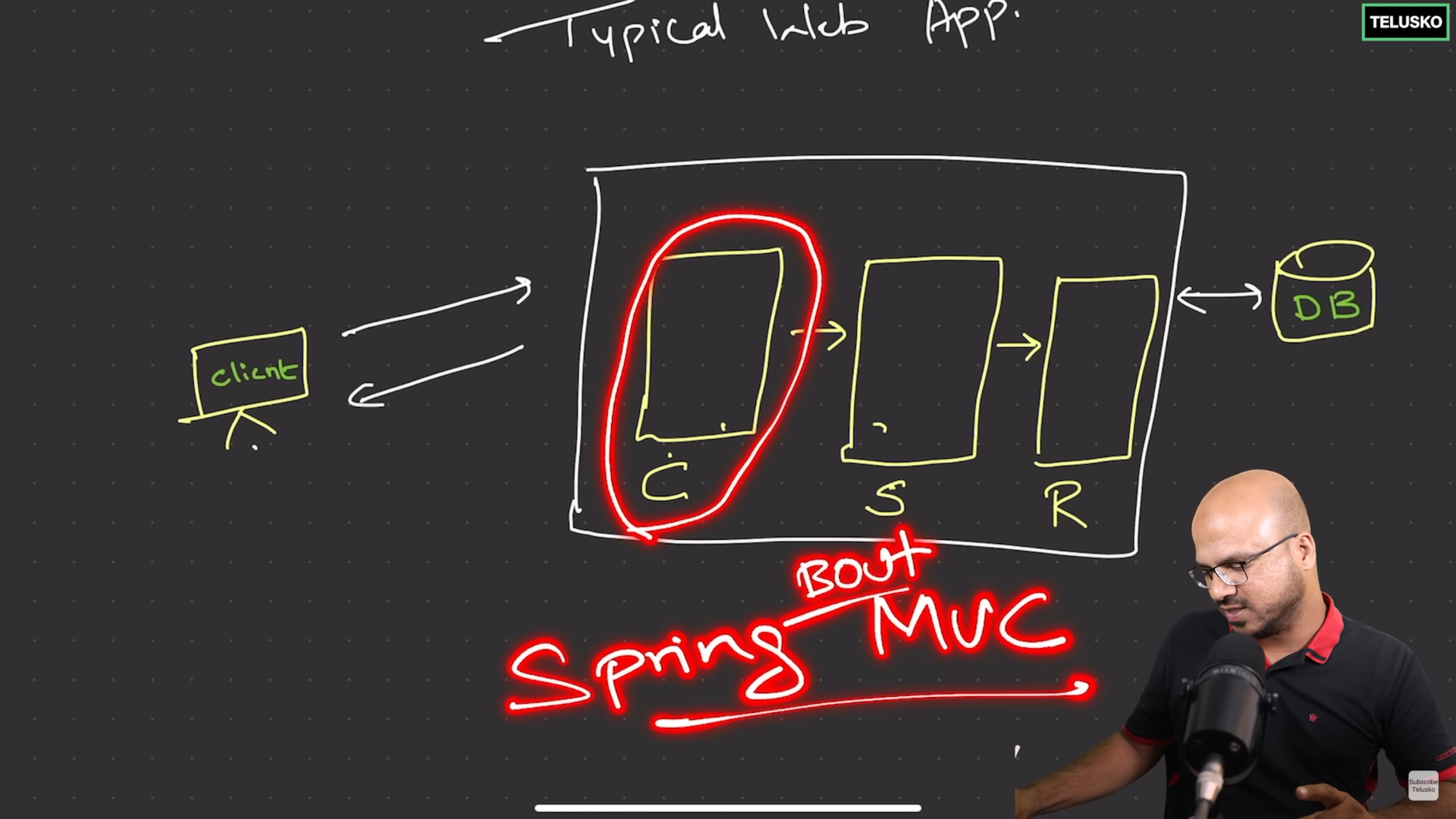
**Give reference to the other beans (Injection)**

<property name = "age" ref = "lap1" />

 <constructor-arg ref = "lap1" />

**Without Sprinboot Autowireing**





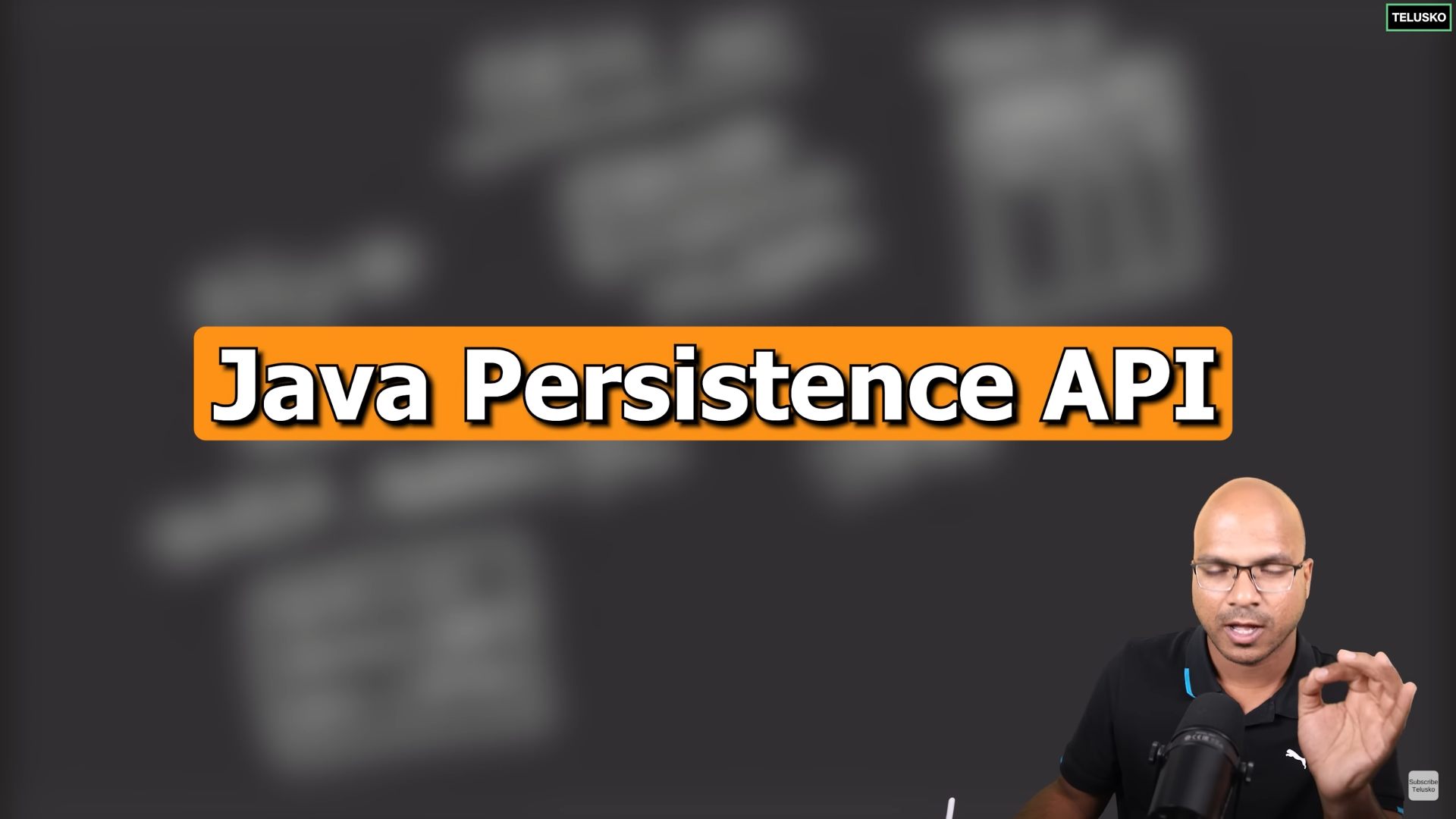
(Front Controller)|Control service and repo

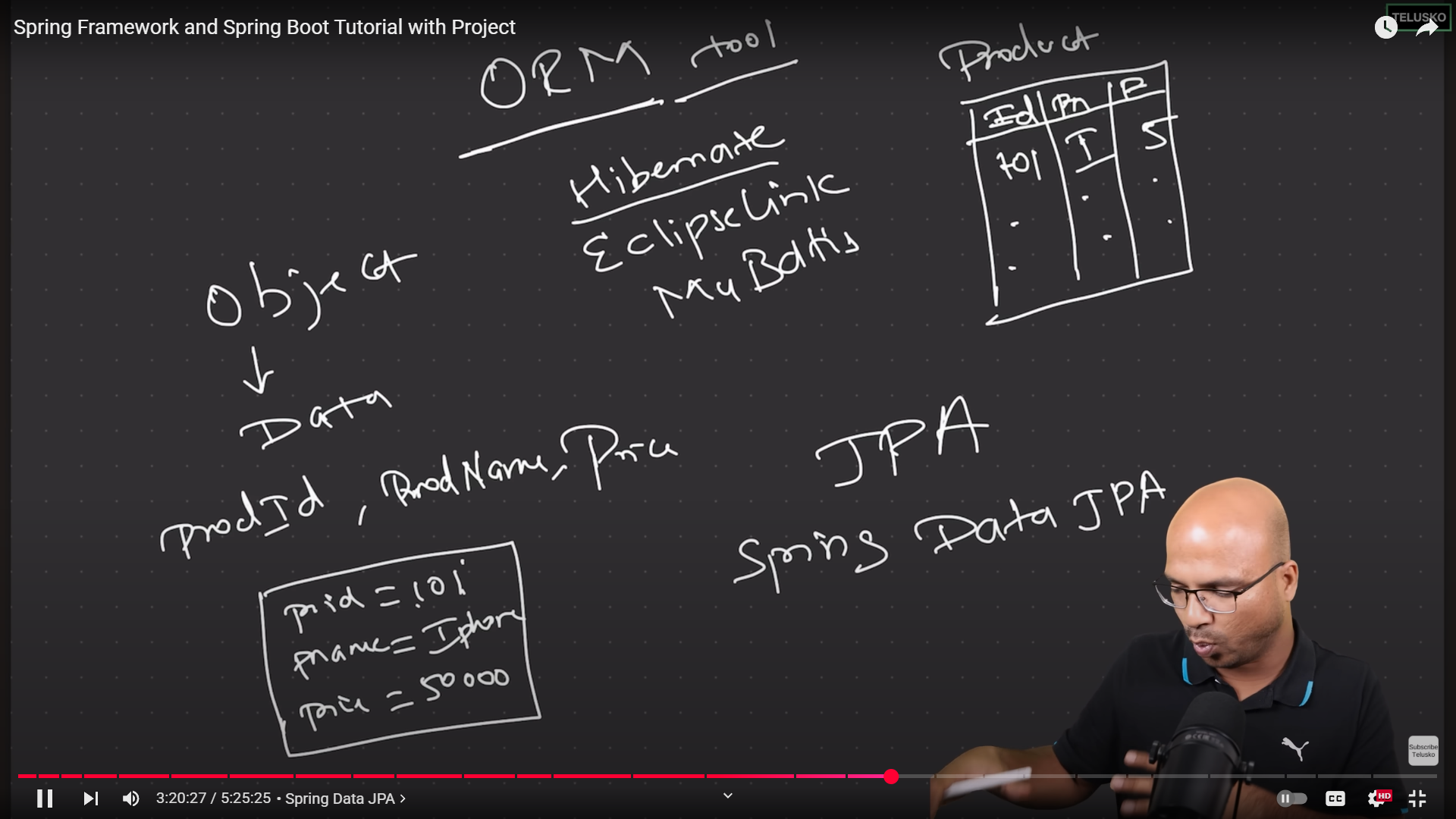
Rest: Representational State Transfer

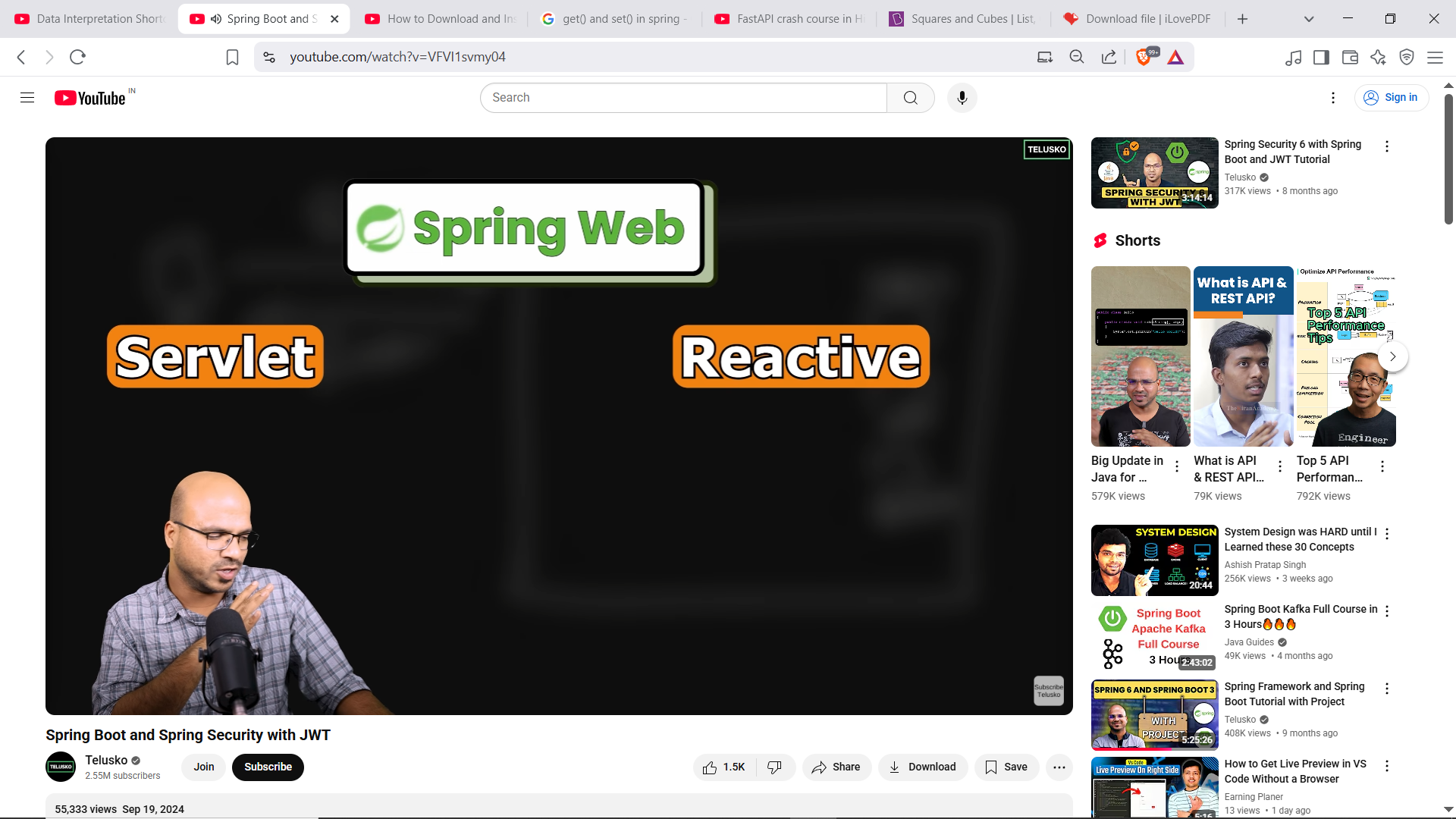
getProductById(@PathVariable int proId)

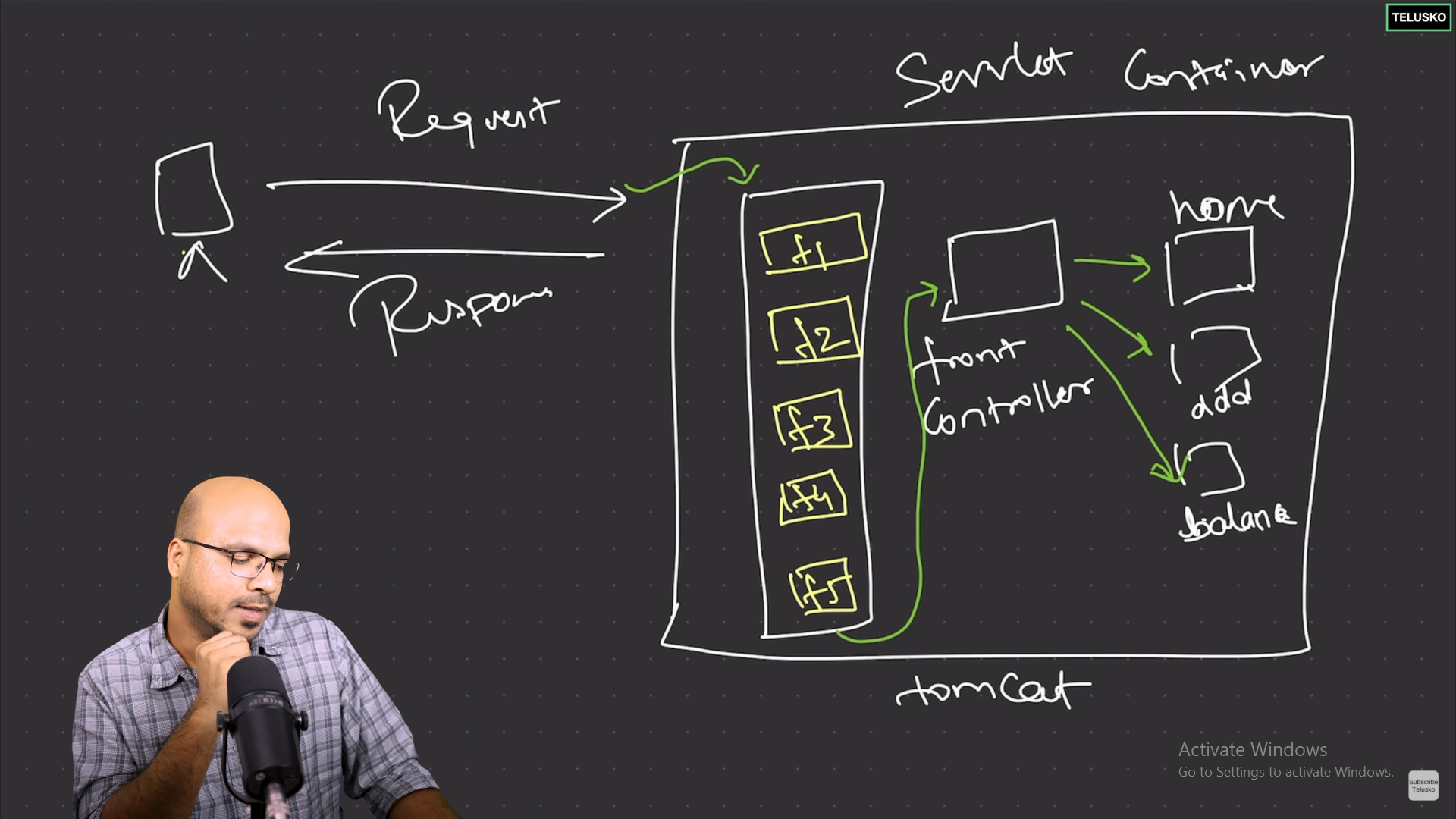
Jackson=>json to obj, obj to json











**Auth**

CSRF token

