**Capstone-Project- Integration & Deployment-BackEnd API**

**Introduction:-**

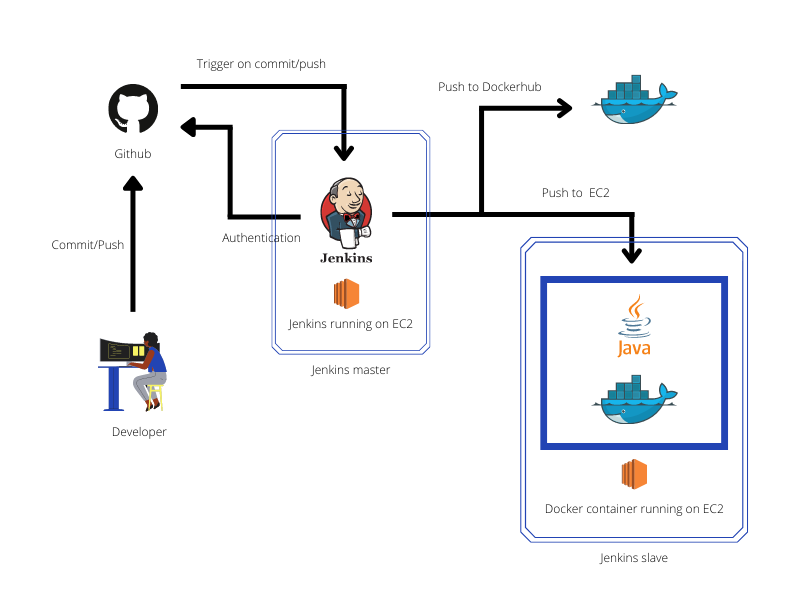
In this project, we will learn how to set up a continuous integration and continuous delivery (CI/CD) pipeline on AWS. A pipeline helps us to automate steps in your software delivery process, such as introducing automatic builds and then deploying to Amazon EC2 instances. We will use Jenkins, a service that builds, tests, and deploys the code every time there is a code change, based on the release process models defined. As part of our setup, we will plug other devops tools into jenkins to complete your software delivery pipeline. The other tools which are used are github and Maven and others based on requirement. Github acts as source code repository and Maven acts as Building tool. The detailed diagram will be created based on project requirement and this guide will show you how to create a very simple pipeline that pulls code from a source and automatically deploys it an amazon ec2 instances

**Team Details:-**

| **Project Leaders** | |
| --- | --- |
| Sangram | Jenkins & AWS CLI |
| Akshit | Docker & Npm |
| Anshul | ECR, Terraform, Documentation |
| Malcon | ECS |
| Narayanan | Github hook,Docker hub & Integration |

## 

## **Architecture**



| **AWS Instance Details** | Microservices-Slave |
| --- | --- |
| Instance Type | t2.Medium |
| ROM | 15GB |
| RAM | 4GB |
| OS | Linux |
| VPC/Subnet | Default |
| Security Group | Whatever Required Ports for Microservices |
| IP | 34.213.207.172 |
| Dependencies | **Maven 3.x,JQ,Java 17, Mysql,Rabbit MQ, Docker, Git, Python,AWS CLI** |

| **AWS Instance Details** | Jenkins-Master |
| --- | --- |
| Instance Type | t2.Medium |
| ROM | 8GB |
| RAM | 4GB |
| OS | Linux |
| VPC/Subnet | Default |
| Security Group | 8080 |
| IP | <http://18.237.63.157/> |
| Dependencies | Jenkins,AWS CLI, GIT |

**CI CD Docker Tool Stack**

1. **Github**It is a web-based application or a cloud-based service where people or developers collaborate, store and manage their application code using Git. We will create and store our application code here.
2. **AWS EC2 Instance**AWS EC2 is an Elastic Computer Service provided by Amazon Web Services used to create Virtual Machines or Virtual Instances on AWS Cloud. We will create an EC2 instance and install Jenkins and other dependencies in it.
3. **Java**This will be required to run Jenkins Server.
4. **AWS CLI**aws-cli i.e AWS Command Line Interface is a command-line tool used to manage AWS Services using commands.
5. **Maven**Maven is a build automation tool used primarily for Java projects. We will be creating a CI CD Docker Pipeline for the application.
6. **Docker**Docker is an open-source containerization platform used for developing, shipping, and running applications. We will use it to build Docker Images of our sample application.
7. **Jenkins**Jenkins is an open-source, freely available automation server used to build, test, and deploy software applications. We will be creating our CI CD Docker Pipeline to build, test and deploy our application on AWS EC2 using Jenkins
8. **Docker Hub**Docker Hub is a registry service on the cloud that allows you to download Docker images that are built by communities. We will be using Docker Hub to store Docker Images of our sample application.

## 

## **CI CD workflow and phases**

### **Workflow**

CI and CD Workflow allows us to focus on Development while it carries out the tests, build, and deployments in an automated way.

1. **Continuous Integration**  
   This allows the developers to push the code to the Version Control System or  Source Code Management System, build & test the latest code pushed by the developer and generate & store artifacts.
2. **Continuous Delivery**  
   This is the process that lets us deploy the tested code to the Production whenever required.
3. **Continuous Deployment**This goes one step further and releases every single change without any manual intervention to the customer system every time the production pipeline passes all the tests.

### **Phases**

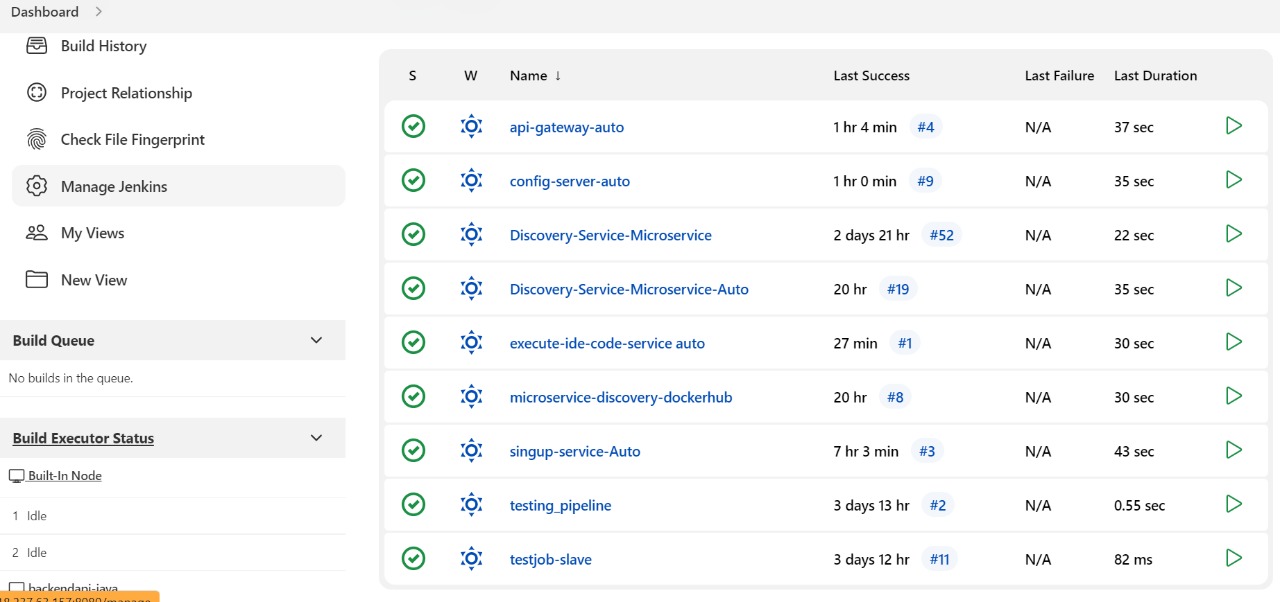
The primary goal of the automated CI CD pipeline is to build the latest code and deploy it. There can be various stages as per the need. The most common ones are mentioned below.

1. **Trigger**The CI CD pipeline can do its job on the specified schedule when executed manually or triggered automatically on a particular action in the Code Repository.
2. **Code Pull**  
   In this phase, the pipeline pulls the latest code whenever the pipeline is triggered.
3. **Build or Package**  
   Once all the tests pass, the pipeline moves forward and builds artifacts or docker images in case of dockerized applications.
4. **Push or Store**  
   In this phase, the code that has been built is pushed to the Artifactory or Docker Repository in case of dockerized applications.
5. **Deploy**This is the final stage in any CI CD pipeline. In this stage, the application is ready for delivery or deployment.
6. **Notification**

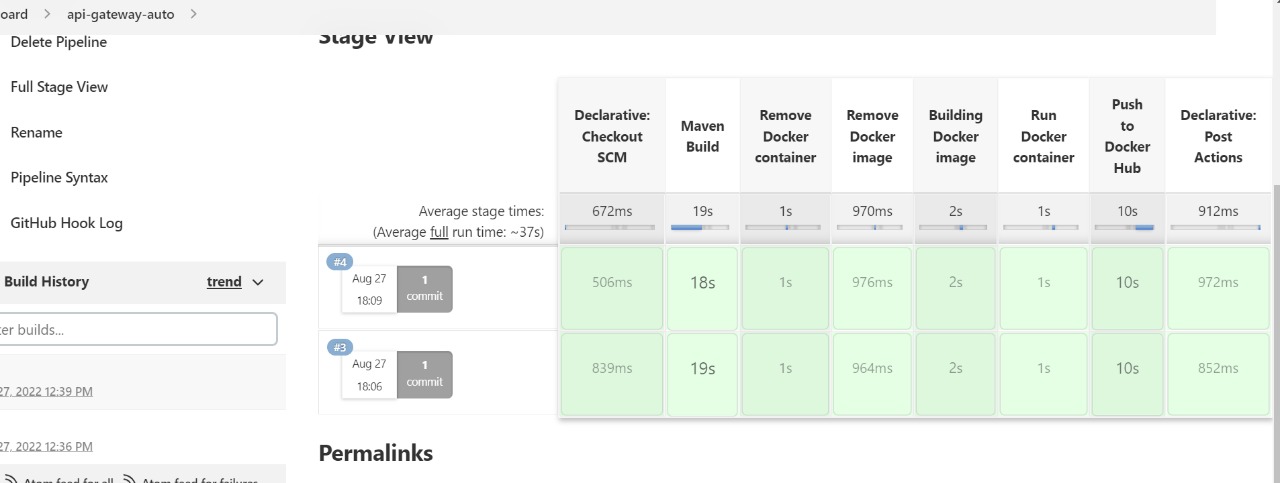
All the build notification(Email) will be sent to build admin users through jenkins whether build has been success or failure along with build number

### **CI CD workflow in action:**

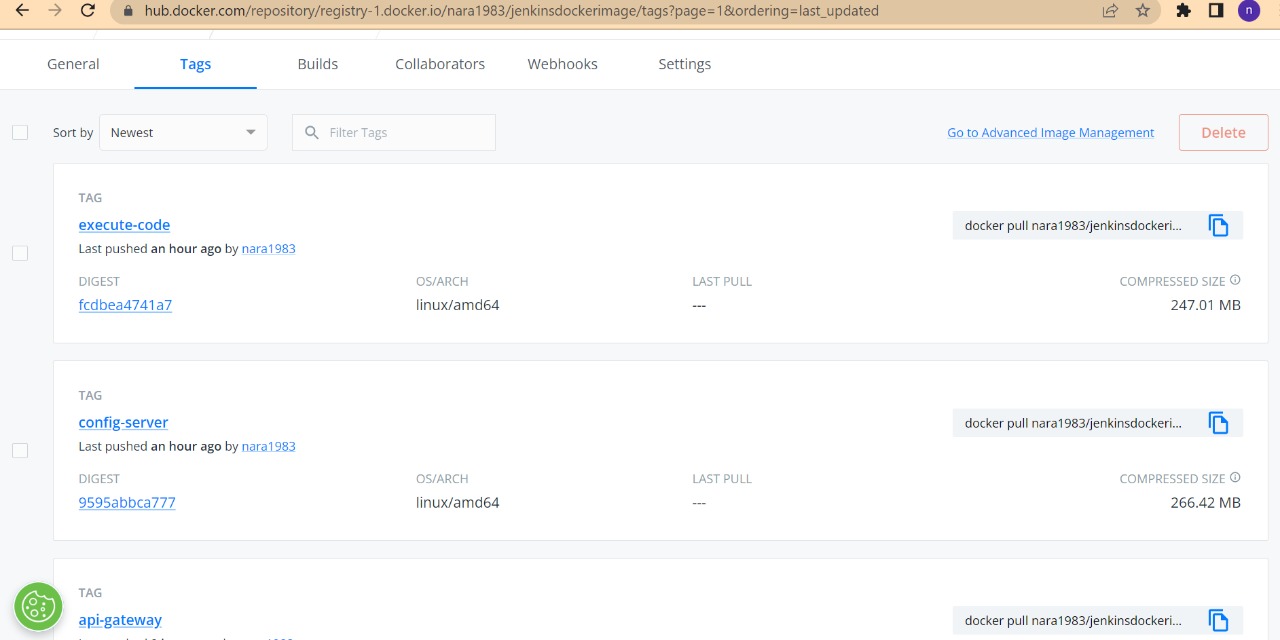
List of CI CD Pipelines in Jenkins for each service:



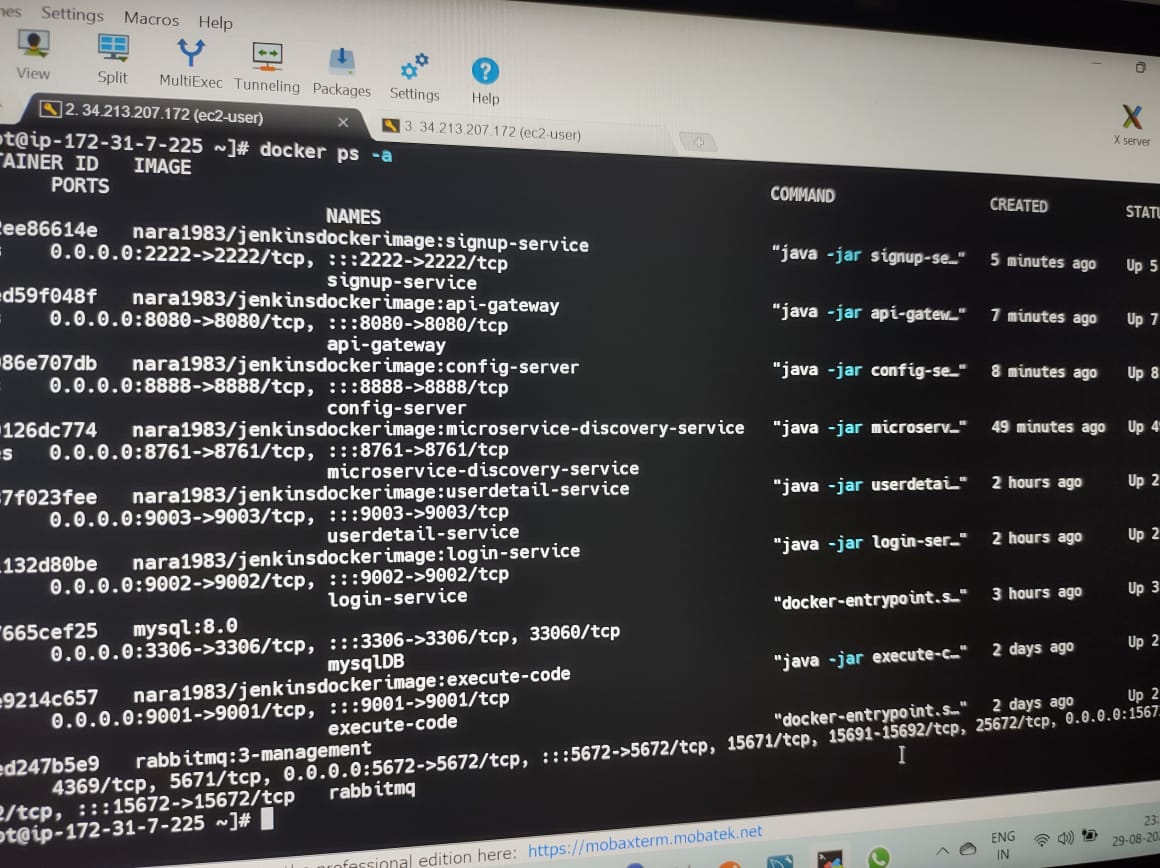
CI CD Pipeline running for a service on push/commit to repository:

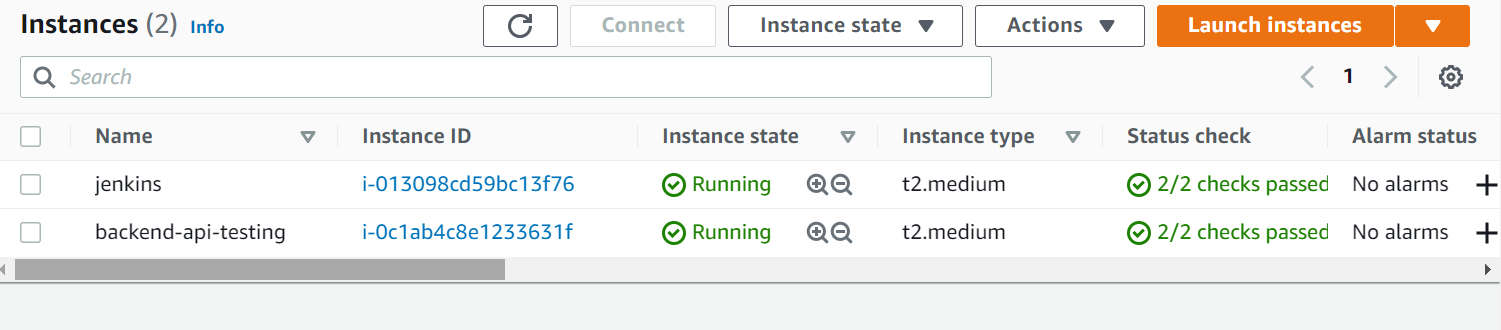


Images pushed to Docker hub as part of CI CD pipeline:

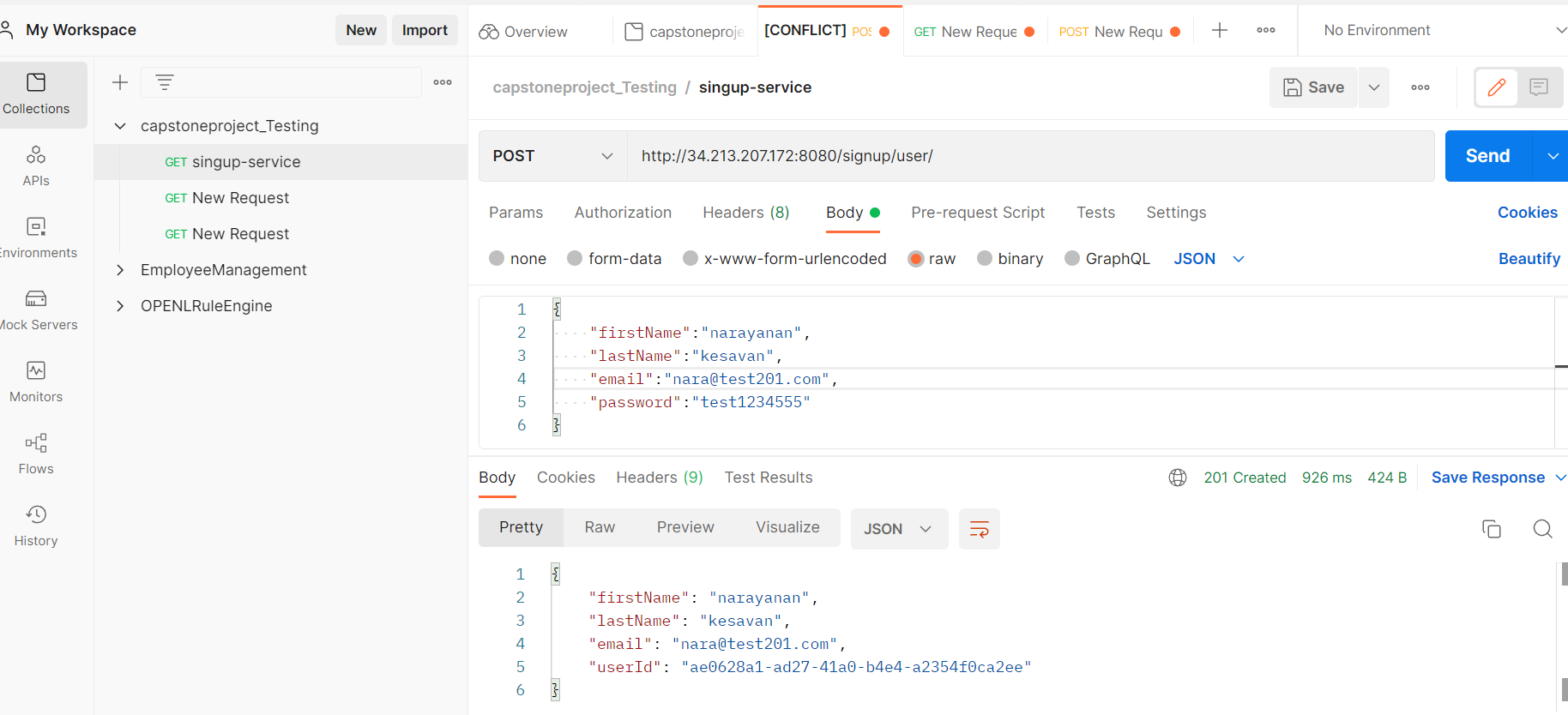


All services successfully deployed and running as Docker containers in EC2:

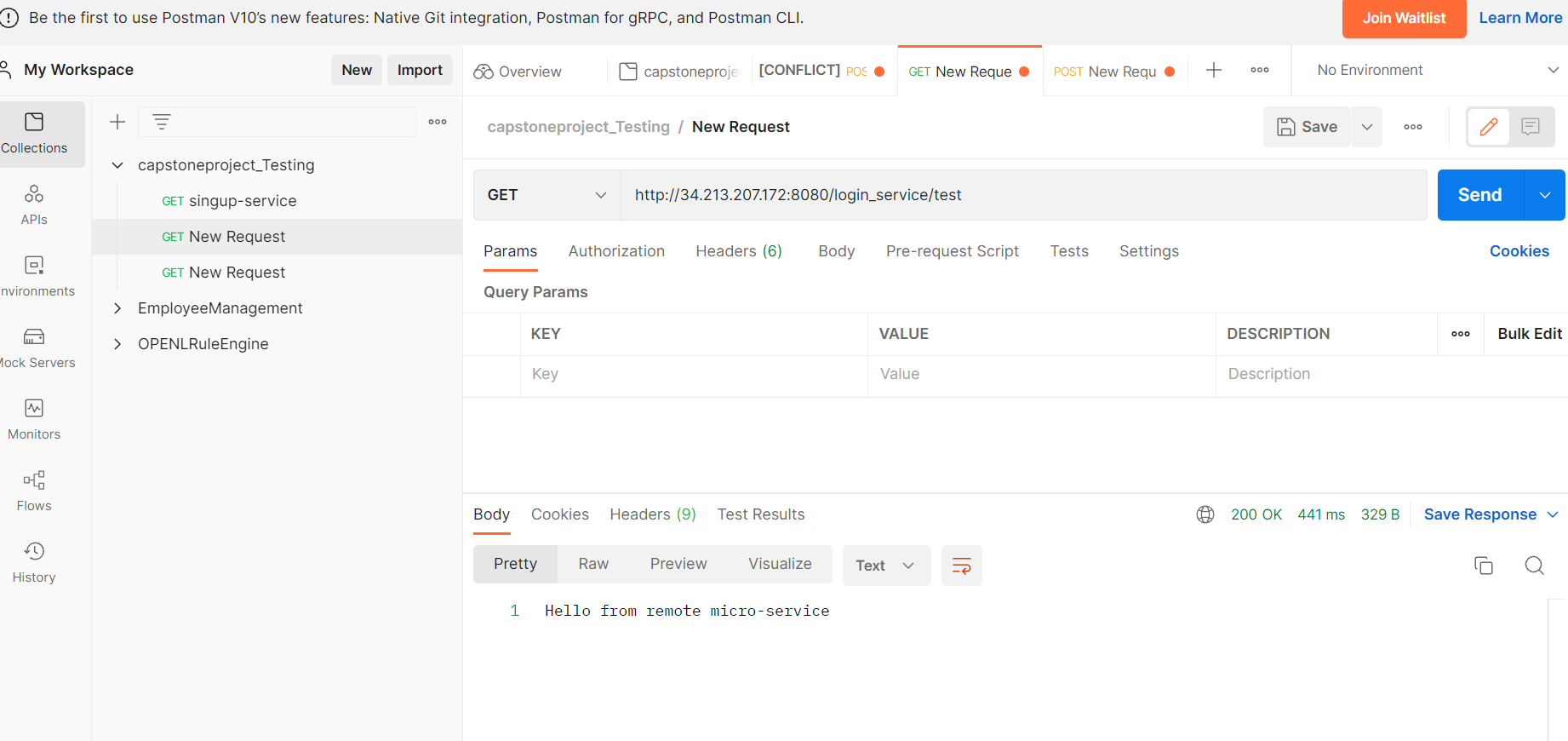




SignUp-Service:-



Login-Service:-



**Application URLS:-**

SignUp Microservice- <http://34.213.207.172:8080/signup/user/>

Jenkins- <http://18.237.63.157:8080/>

Eureka Server- <http://34.213.207.172:8761/>

Rabbit MQ- <http://34.213.207.172:15672/#>

MySQL- 34.213.207.172:3306:root

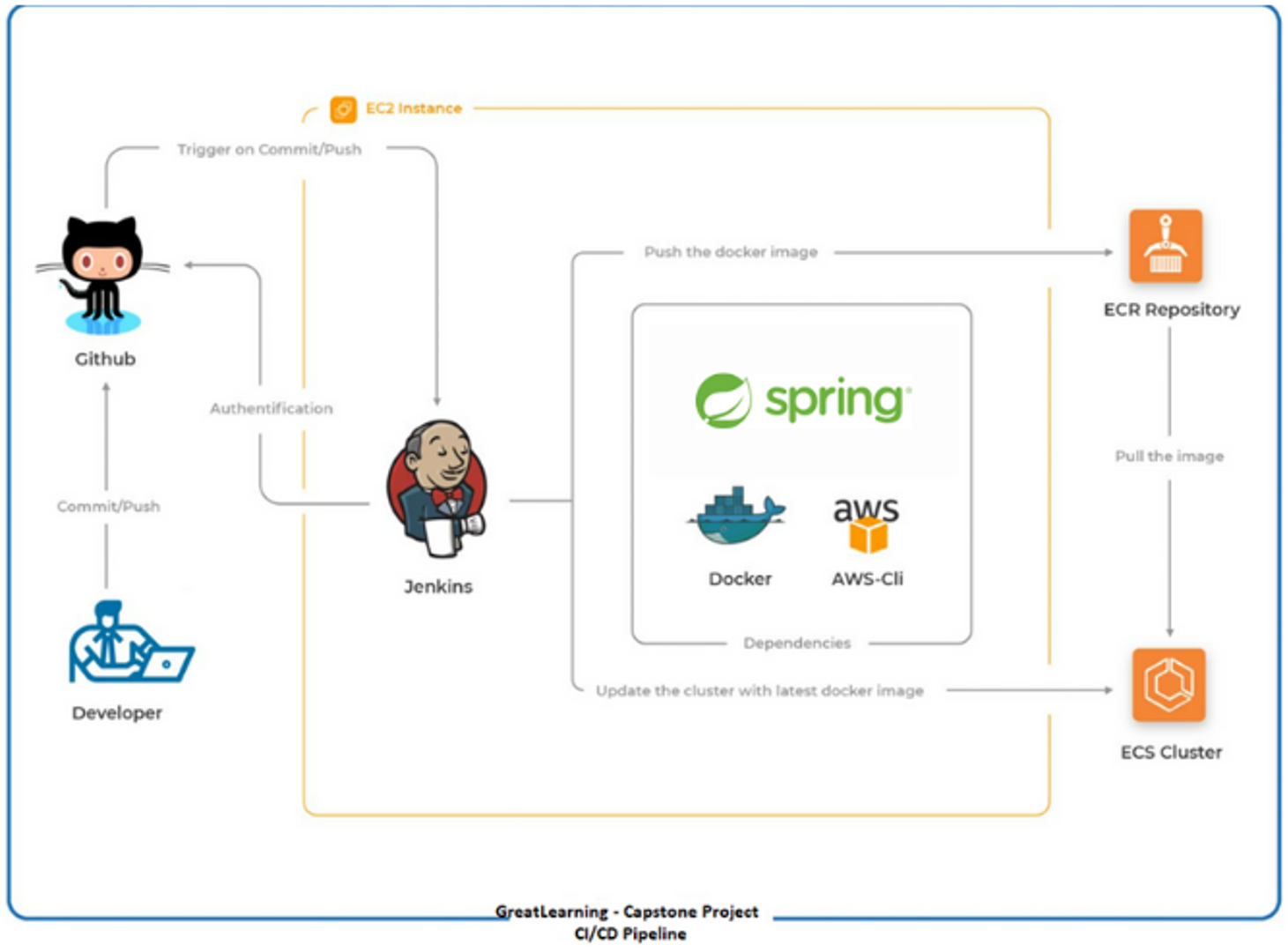
UserDetail- <http://34.213.207.172:8080/user_detail/admin/upgradeUserToAdmin>

Config-server- <http://34.213.207.172:8888/actuator/info>

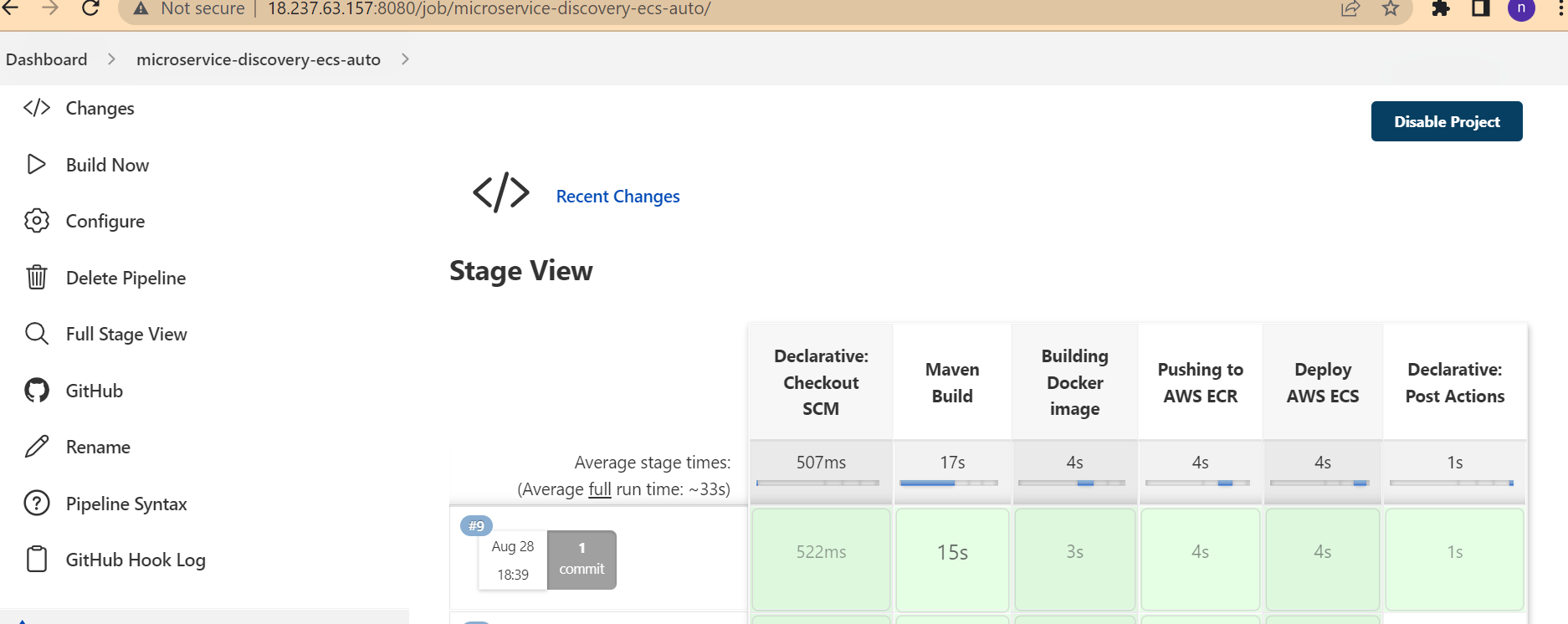
Login service- <http://34.213.207.172:8080/login_service/test>

**ECS Application URL:-**

<http://ec2co-ecsel-zo1y376vsx-264549342.us-west-2.elb.amazonaws.com:8761/>



**Microservice on ECR-ECS -Jenkins Job:-**

****

All services showing up on Discovery server:

