

AWS, CI/CD

DAY-ONE

■ [EC2-Hands-on]

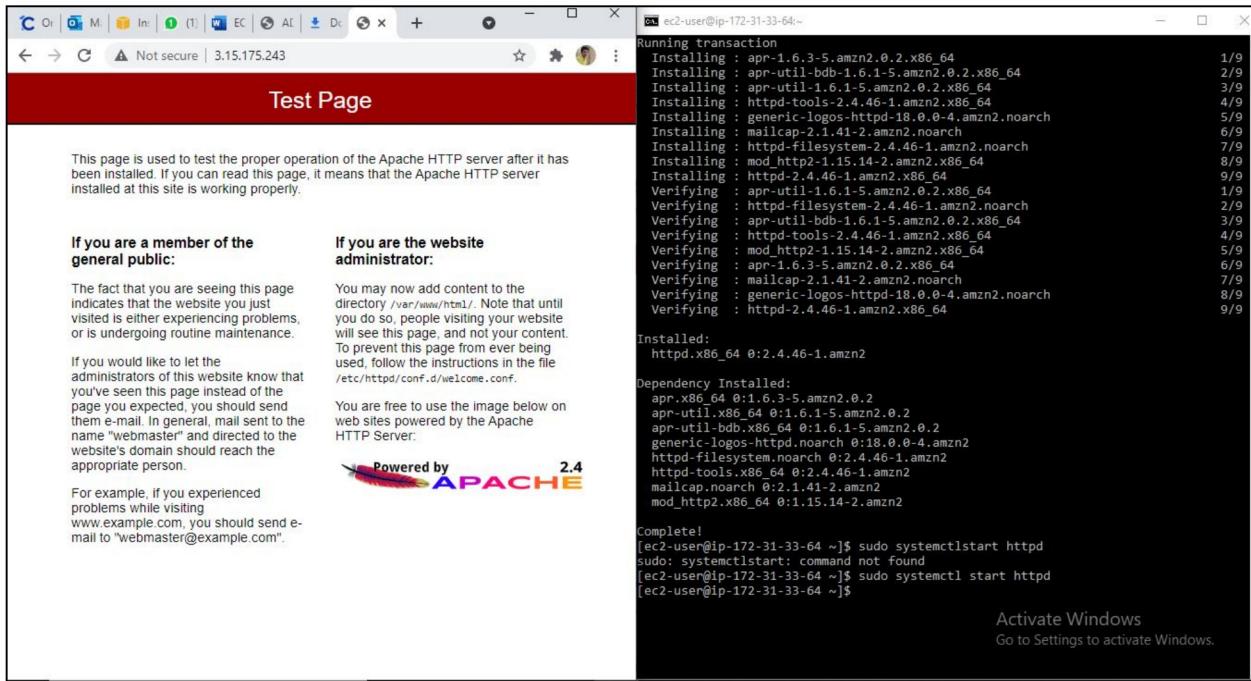
Create an EC2 instance, connect to it from your local system and install apache web server on the EC2 instance.

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with options like EC2 Dashboard, Events, Tags, Limits, Instances (with sub-options like Instances, Instance Types, Launch Templates, etc.), and Images. The main area is titled 'Instances (1/1) Info' and shows a table with one row. The row details are: Name (Shylesh-EC2), Instance ID (i-00a7da2eef7f502d9), Instance state (Running), Instance type (t2.micro), Status check (2/2 checks passed), Alarm status (No alarms), and Availability Zone (us-east-2c). There are buttons for 'Connect', 'Actions', and 'Launch instances' at the top of the table.

This indicates my instance is up and running

The screenshot shows a browser window with two panes. The left pane is the 'EC2 Instance Connect' interface, which includes fields for Instance ID (i-00a7da2eef7f502d9), Public IP address (3.15.175.243), and User name (ec2-user). It also has a note about the user name being correct. The right pane is a terminal window titled 'ec2-user@ip-172-31-33-64:~'. It displays a warning about host fingerprint authentication, a warning message from Amazon Linux 2 AMI, and a command prompt showing the user is connected to https://aws.amazon.com/amazon-linux-2/. The terminal window also shows a note about package updates.

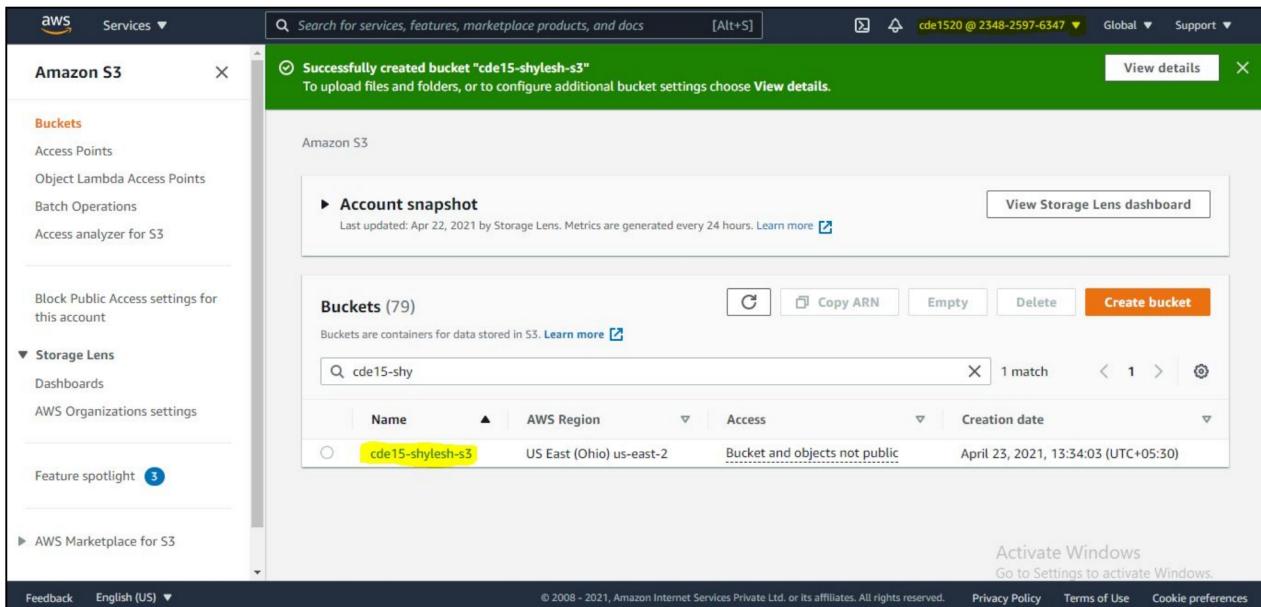
Connected to the EC2 instance



Type the public IPV4 address on the browser url bar and we should get the above shown screen

■ [S3-Hands-on]

Create a S3 bucket and store an object in it. Enable to object for public access so that anyone can access it through a web browser.



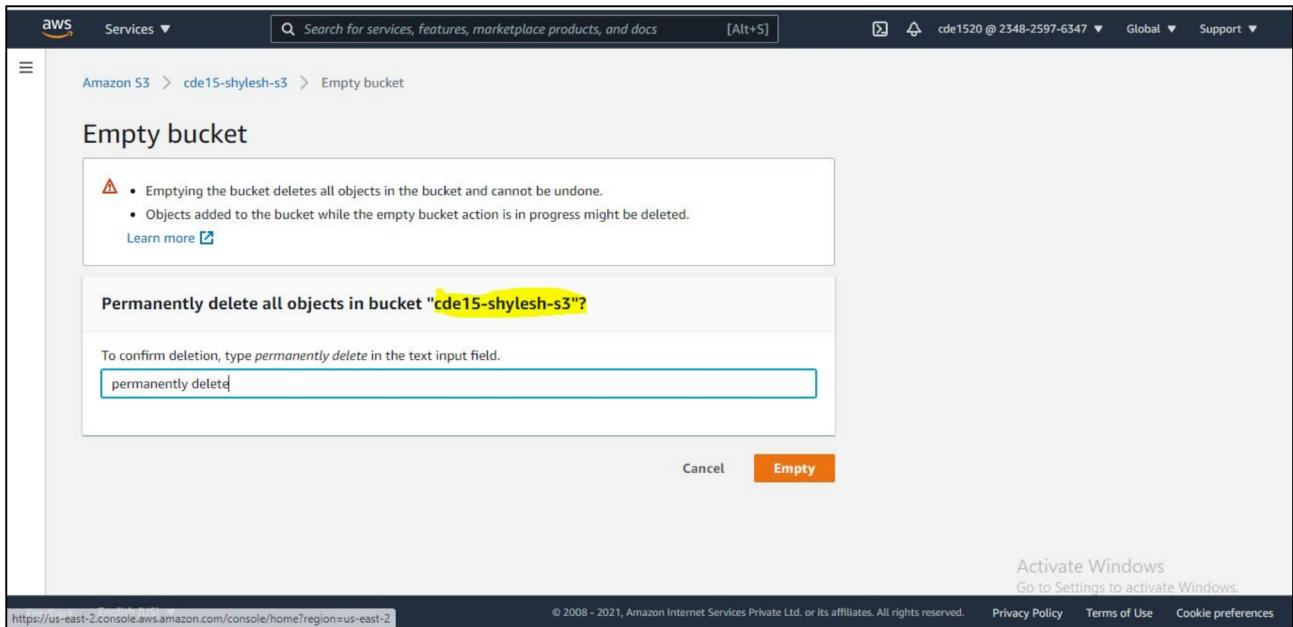
Once successfully S3 created, we will be taken to the above shown

The screenshot shows the AWS S3 console after a successful upload. The top bar indicates "Upload succeeded". The "Summary" section shows the destination as "s3://cde15-shylesh-s3" with 2 files uploaded successfully (6.3 KB, 100.00%) and 0 files failed. Below this, the "Files and folders" tab is selected, showing a list of 2 items: "demo.html" (text/html, 39.0 B, Succeeded) and "demo.png" (image/png, 6.3 KB, Succeeded). A search bar at the top of the list table allows "Find by name". The footer includes links for Feedback, English (US), Privacy Policy, Terms of Use, and Cookie preferences.

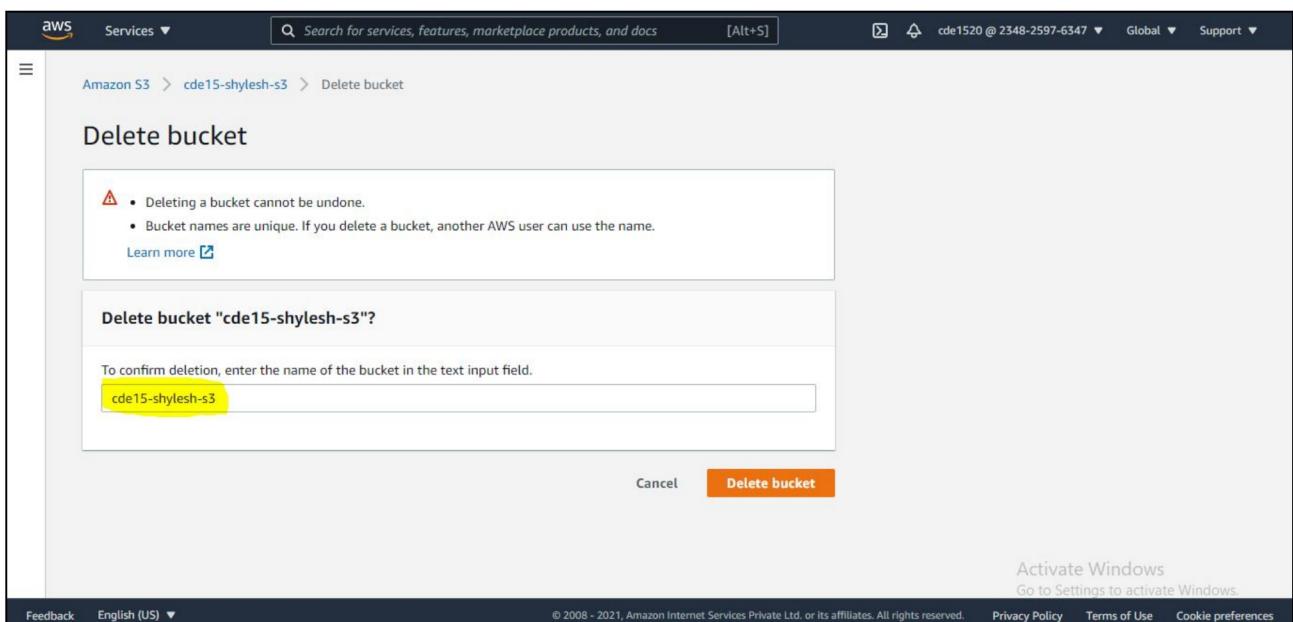
Once files and folders will be uploaded we will be taken to the status screen as shown above

The screenshot shows the AWS S3 object details for "demo.png". The left panel displays the object's properties: Owner (Amazon), AWS Region (US East (Ohio) us-east-2), Last modified (April 23, 2021, 13:41:03 (UTC+05:30)), and Size (6.3 KB). It also includes "Copy S3 URI" and "Object actions" buttons. The right panel shows a preview of the image in a browser window, displaying a green and white interface for "HTML" and "CSS" editing. The footer of the S3 page includes links for Feedback, English (US), Privacy Policy, Terms of Use, and Cookie preferences.

Access the object by clicking in the Object URL and we can view the object in the browser



Click on the “Empty” button to delete the contents of the bucket



Type the name of the bucket we are deleting and click on the “Delete bucket” button

DAY-TWO

■ [RDS-Hands-on]

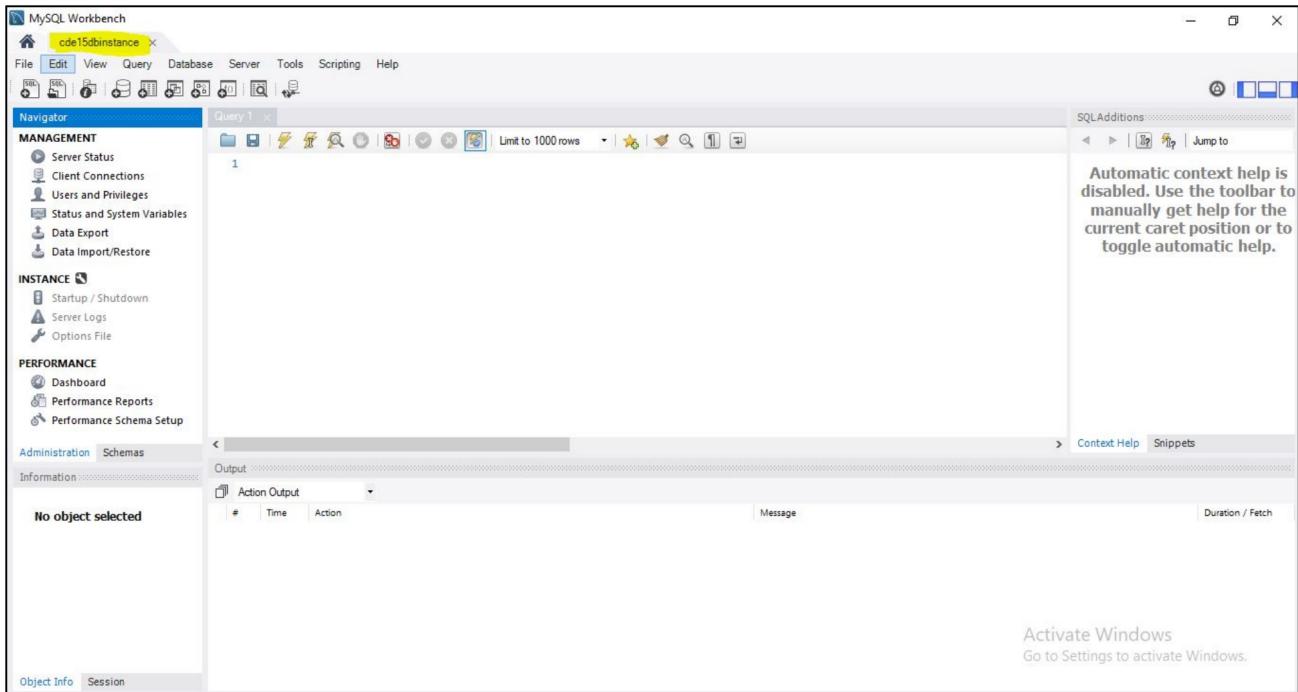
Create a RDS database in AWS and access it through the local client tool.

The screenshot shows the AWS RDS (Relational Database Service) console. In the top navigation bar, the search bar contains "Search for services, features, marketplace products, and docs". The top right corner shows "cde1520 @ 2348-2597-6347" and "Ohio". The main area is titled "Creating database cde15dbinstance" with the sub-instruction "Your database might take a few minutes to launch.". Below this, the "Databases" section lists one entry: "cde15dbinstance" (Instance, MySQL Community, us-east-2b, db.t2.micro). A search bar at the top of the list table contains "cde15db". The left sidebar includes links for Dashboard, Databases (which is selected), Query Editor, Performance Insights, Snapshots, Automated backups, Reserved instances, Proxies, Subnet groups, Parameter groups, Option groups, Events, Event subscriptions, Recommendations (with 39 notifications), and Certificate update.

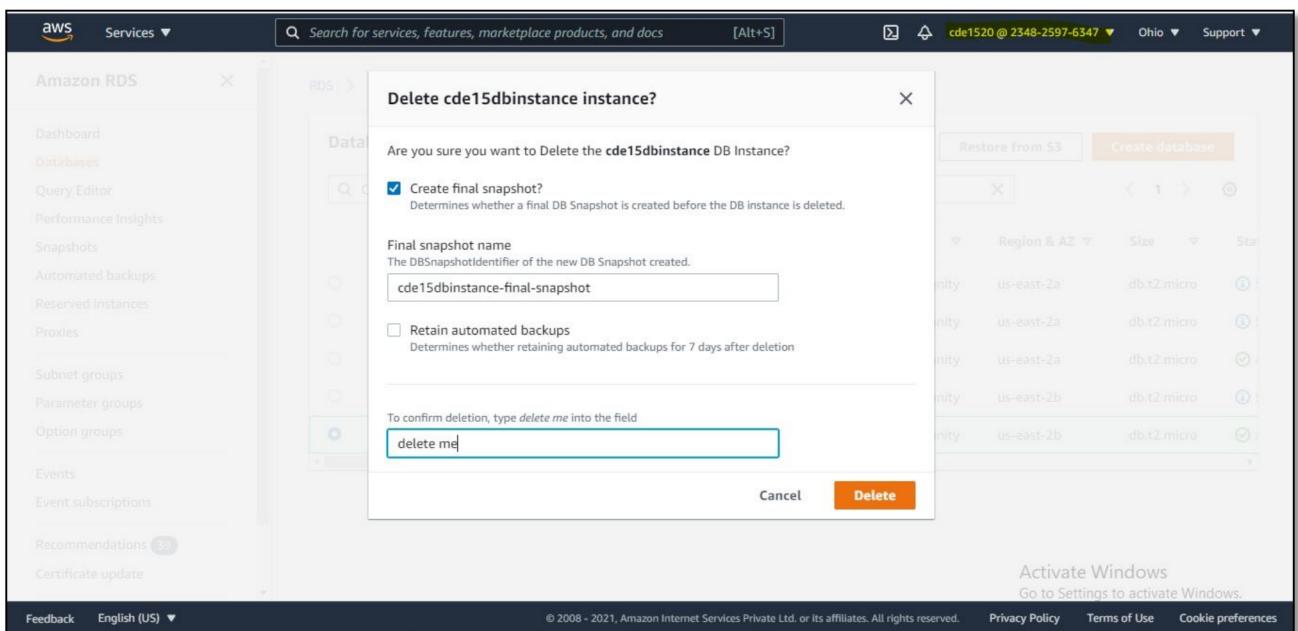
We can see above screen that our database is being created

The screenshot shows the MySQL Workbench application. The title bar says "MySQL Workbench". The main window displays a "Setup New Connection" dialog box. The "Connection Name" field is set to "cde15dbinstance". The "Connection Method" is "Standard (TCP)". Under "Parameters", the "Hostname" is "im0e5u9u.us-east-2.rds.amazonaws.com", "Username" is "root", and "Default Schema" is empty. The "SSL" tab is selected. The "Advanced" tab is visible but empty. A message box in the center says "Successfully made the MySQL connection" with details: "Host: cde15dbinstance.cibxim0e5u9u.us-east-2.rds.amazonaws.com", "Port: 3306", "User: root", and "SSL enabled with TLS_AES_256_GCM_SHA384". Below this, it says "A successful MySQL connection was made with the parameters defined for this connection." At the bottom of the dialog are "OK", "Test Connection", "Cancel", and "OK" buttons. To the left of the dialog, the "MySQL Connections" tree view shows "root" and "shylesh" under "shylesh", and "127.0.0.1:3306". The status bar at the bottom right says "Activate Windows Go to Settings to activate Windows."

We will be connected to the RDS MySQL Server



We can see above screen that we are connected to the RDS MySQL server and query window is opened. Here we can issue any sql commands we want to execute against the RDS database



Deleting the RDS database by typing “delete me” in the confirm textbox and press “Delete” button

■ [AWS-lab-hands-on-practise]

Create various infrastructure components that will be used to build a web server

within the AWS cloud environment.

The screenshot shows the AWS VPC dashboard with the following details:

Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR
md_abdullah_baig_897017	vpc-04fd134e78787f12d	Available	10.0.0.0/16	-
CDE15-SHYLESH-VPC	vpc-05e18a7453bcc0546	Available	10.0.0.0/16	-
Meghna_LabVPC	vpc-0f4fbee15cf592171	Available	10.0.0.0/16	-
-	vpc-550e813e	Available	172.31.0.0/16	-
LABVPC	vpc-0b111f1fdc172238d	Available	10.0.0.0/16	-

We can see above screen that VPC is created

The screenshot shows the AWS VPC dashboard with the following details:

Name	Internet gateway ID	State	VPC ID
md_abdullah_baig	igw-008d51c955e61fa2a	Attached	vpc-04fd134e78787f12d md_abdull...
Meghna-IG	igw-025851ce77ee5676	Attached	vpc-0f4fbee15cf592171 Meghna_La...
CDE15-SHYLESH-VPC-IG	igw-04315e751c2fdf21d	Detached	-
vaibhav-internet-gateway	igw-06343623bafec8227	Attached	vpc-0b111f1fdc172238d LABVPC
-	igw-3e745c56	Attached	vpc-550e813e _

We can see above screen that INTERNET GATEWAYS is created

The screenshot shows the 'Attach to VPC' interface. At the top, it says 'Attach to VPC (igw-04313e751c2fdf21d)'. Below this, there's a section titled 'Available VPCs' with a note: 'Attach the internet gateway to this VPC.' A search bar labeled 'Select a VPC' contains the text 'vpc-05e18a7453bcc0546 - CDE15-SHYLESH-VPC'. There is also a link '▶ AWS Command Line Interface command'. At the bottom right are 'Cancel' and 'Attach internet gateway' buttons.

Attach to VPC

The screenshot shows the 'Internet gateways (5)' list. A green banner at the top says 'Internet gateway igw-04313e751c2fdf21d successfully attached to vpc-05e18a7453bcc0546'. The table lists five internet gateways, all of which are 'Attached'. One row, 'CDE15-SHYLESH-VPC-IG', is highlighted with a yellow background. The table has columns: Name, Internet gateway ID, State, and VPC ID. At the bottom, there's a note 'Select an internet gateway above' and a 'Create internet gateway' button.

Internet gateway is successfully attached to VPC

The screenshot shows the AWS Route Tables interface under the 'Edit routes' section. A new route is being added for destination '10.0.0.0/16' with target 'igw-'. The status is 'active' and propagation is set to 'No'. The route ID is 'igw-04313e751c2fdf21d' and it is associated with 'CDE15-SHYLESH-VPC-IG'. The 'Save routes' button is visible at the bottom right.

We can see above screen that ROUTES is created

The screenshot shows the AWS VPC Subnets interface under the 'Create subnet' section. In the 'VPC' tab, a VPC ID 'vpc-05e18a7453bcc0546 (CDE15-SHYLESH-VPC)' is selected. In the 'Associated VPC CIDRs' tab, an IPv4 CIDR '10.0.0.0/16' is listed. In the 'Subnet settings' tab, a single subnet is being created with the name 'Subnet 1 of 1'. The 'Create Subnet' button is located at the bottom right.

Create Subnet

You have successfully created 1 subnet: subnet-006b5086228432480

Name	Subnet ID	State	VPC	IPv4 CIDR Range
CDE15-SHYLESH-SUBNET2	subnet-006b5086228432480	Available	vpc-05e18a7453bcc0546 CD...	10.0.2.0/24
CDE15-SHYLESH-SUBNET1	subnet-0dca5da34ca3edddf	Available	vpc-05e18a7453bcc0546 CD...	10.0.0.0/24

We can see above screen that SUBNETS is created

Security Groups (1/3) Info

Name	Security group ID	Security group name	VPC ID	Description
-	sg-0b74f60bd22111d0d	awscodestar-cde15-sp...	vpc-550e813e	Enable HTTP access via...
-	sg-0ebaa28e55fd8474	aws-cloud9-cde15-spri...	vpc-550e813e	Security group for AW...
-	sg-0fb26b75bba17677	CDE15-SHYLESH-SG	vpc-05e18a7453bcc0546	CDE15-SHYLESH-SG

We can see above screen that SECURITY GROUPS is created

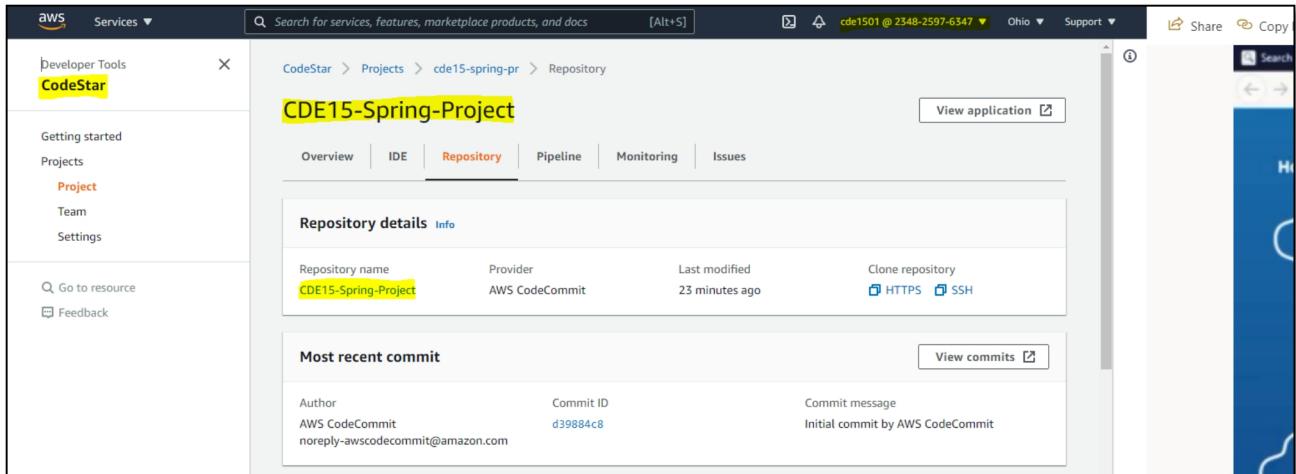
"Hello World from ip-10-0-1-246.ec2.internal"

We can see above screen that browser is display HELLO WORLD

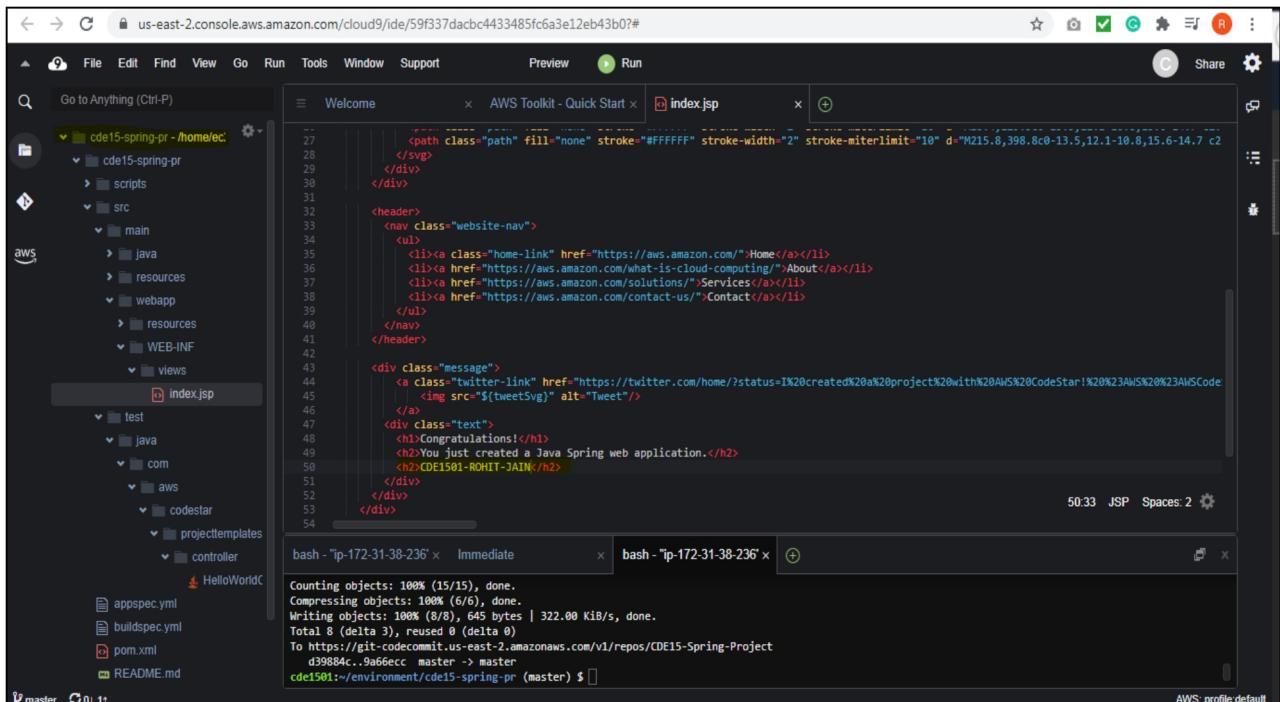
DAY-THREE

▪ [CCID-lab-hands-on-practise]

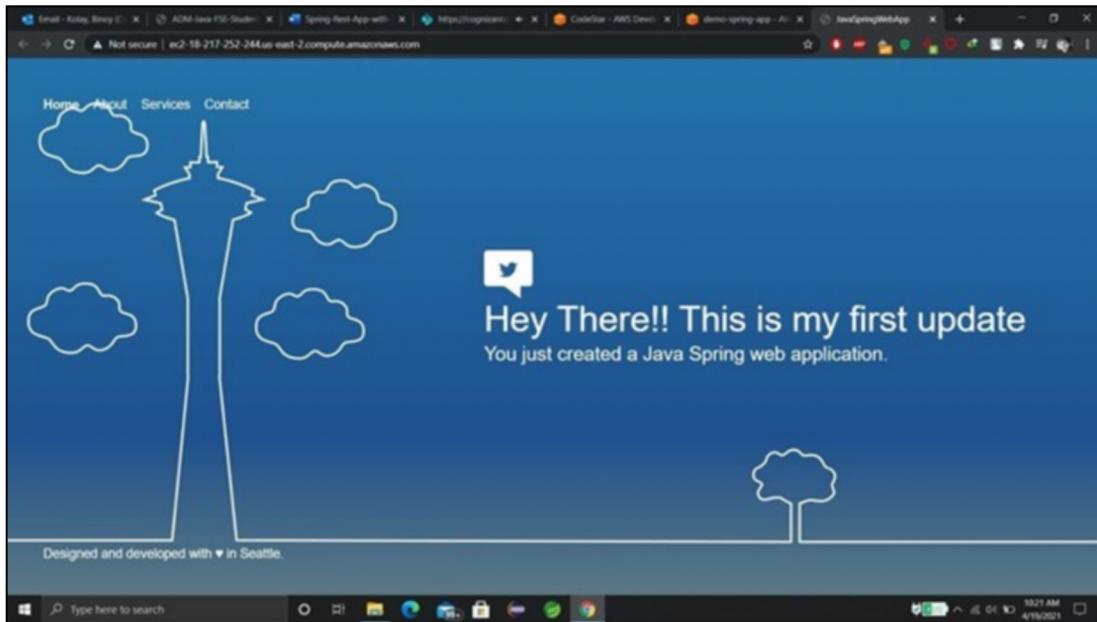
We will able to deploy a spring web application using a Continuous Integration (CI)/ Continuous Delivery (CD) pipeline and the IDE provided by AWS.



We can see above screen that SPRING PROJECT is created



AWS IDE

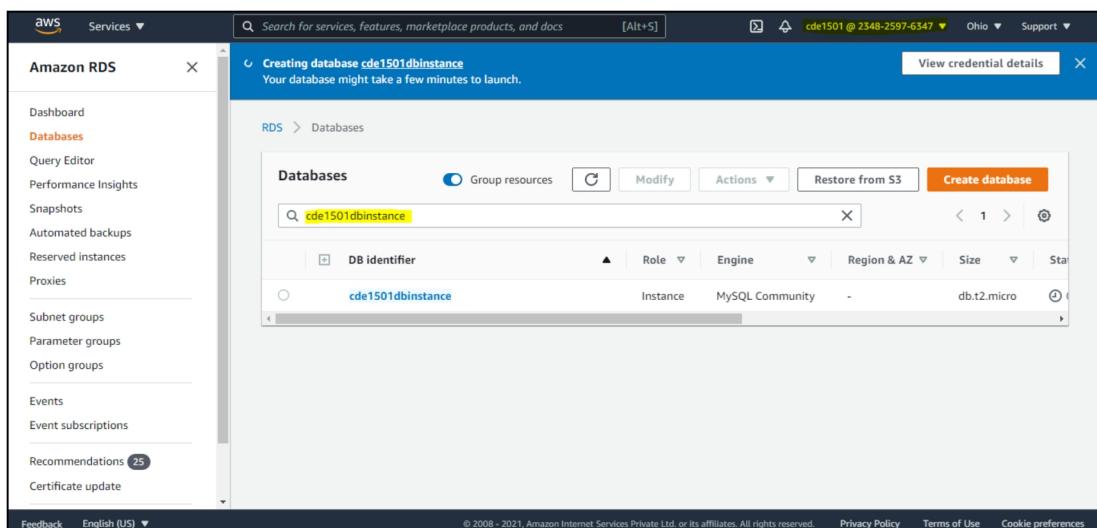


Successfully deploy a spring web application using CI/CD

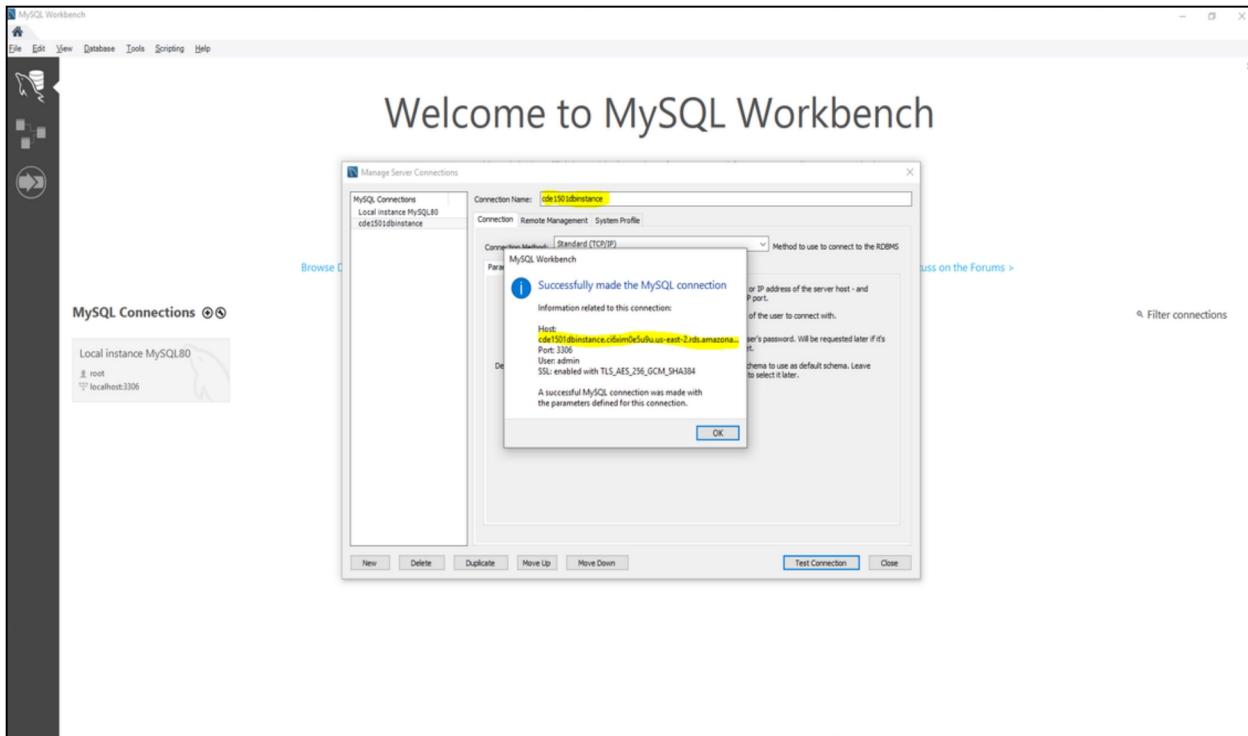
DAY-FOUR

- **[Spring-REST-with-RDS-Backend]**

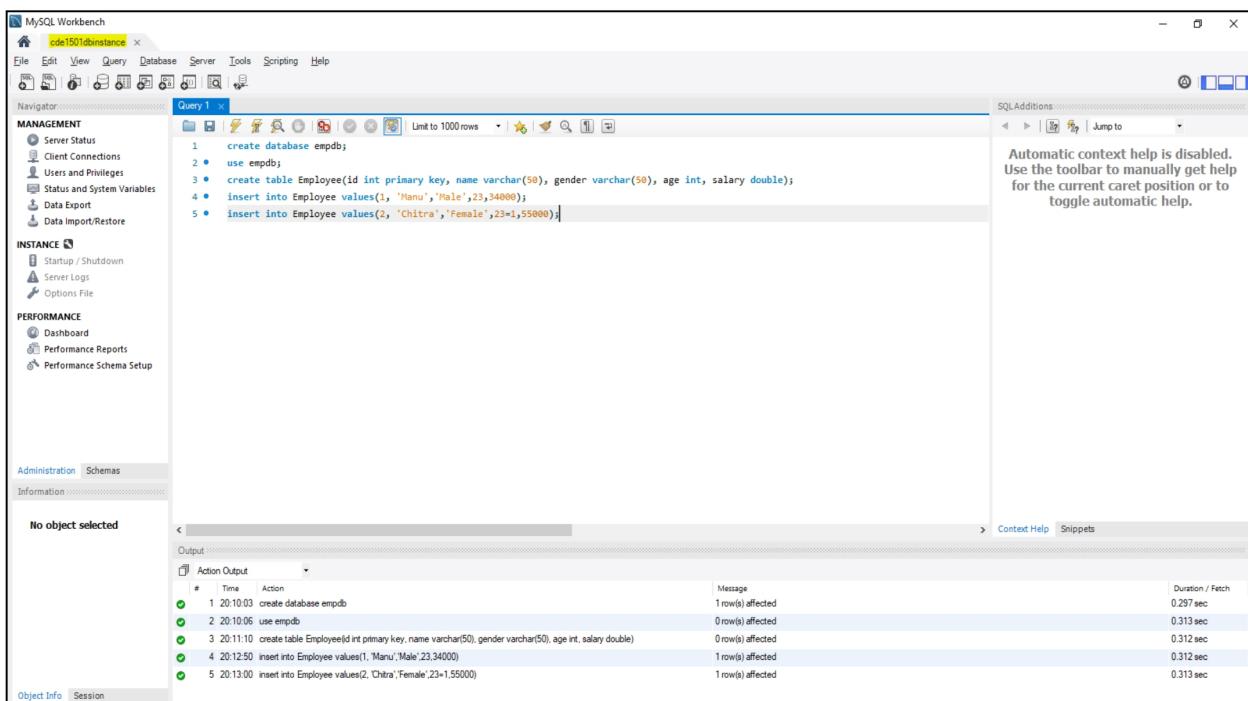
Create a Spring REST application that perform Read and Insert operation on RDS database. Deploy the application in AWS Elastic Beanstalk and access the application from anywhere.



We can see above screen that our database is being created



We will be connected to the RDS MySQL Server



Enter the SQL commands shown in the screen above

```

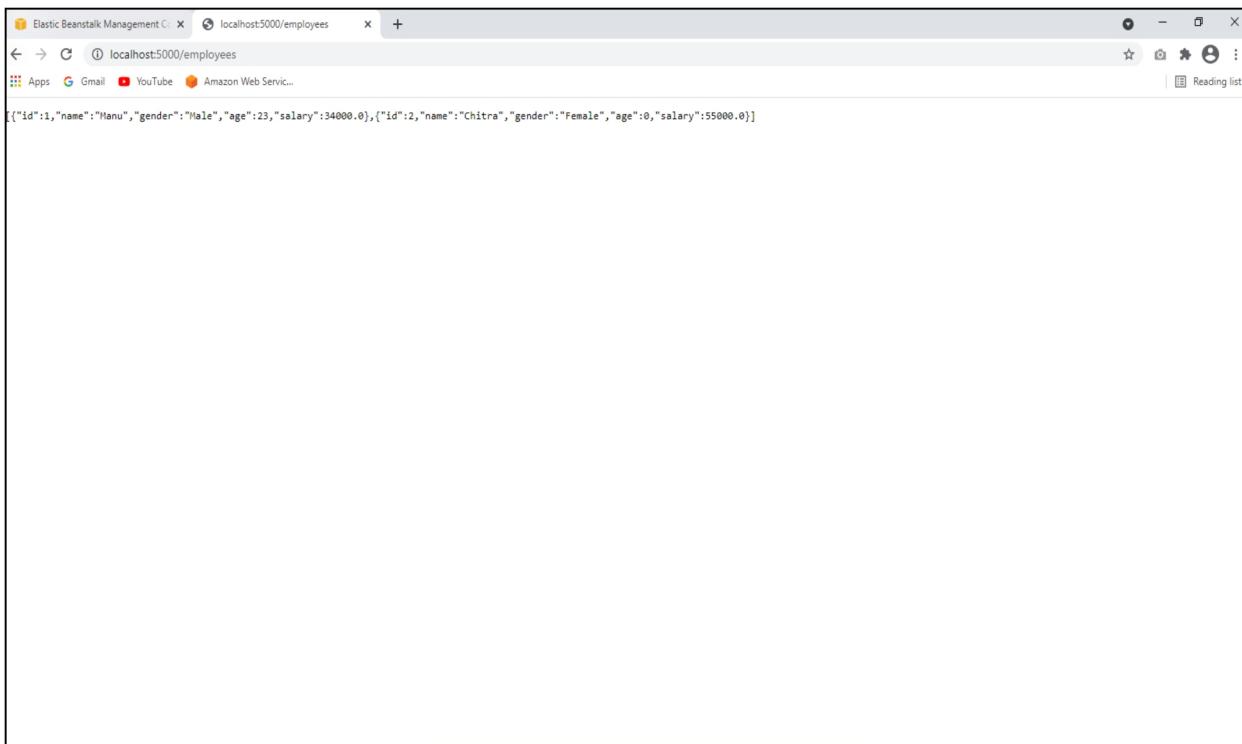
workspace-spring-tool-suite-4-4.4.0.RELEASE - AWS/src/main/java/com/cts/AwsApplication.java - Spring Tool Suite
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer AWS [boot] [devtools]
src/main/java com.cts
  AwsApplication.java
  com.cts.controller EmployeeResource.java
  com.cts.model Employee.java
  com.cts.repository EmployeeRepository.java
  package-info.java
  com.cts.service EmployeeService.java
  EmployeeServiceImpl.java
src/main/resources application.properties
src/test/java JRE System Library [JavaSE-11]
Maven Dependencies target/generated-sources/annotations target/generated-test-sources/test-annotations
src HELP.md mvnw mvnw.cmd pom.xml
target Day 1 - Session 1 [boot] [devtools]
Day 1 - Session 2 [boot] [devtools]
Day 2 - Session 3 [boot] [devtools]
Day 2 - Session 4 [boot] [devtools]
Movie-Cruiser (FinalCheck) [boot] [devtools]
TruYum (PracticeCheck) [boot] [devtools]

EmployeeRes... EmployeeServ... AwsApplication... EmployeeServ... Employee.java
1 package com.cts;
2
3 import org.springframework.boot.SpringApplication;
4
5 @SpringBootApplication
6 @ComponentScan(basePackages="com.cts.*")
7
8 public class AwsApplication {
9
10    public static void main(String[] args) {
11        SpringApplication.run(AwsApplication.class, args);
12    }
13 }
14
15 }
16

2021-04-21 20:43:24.671 INFO 8312 --- [ restartedMain] com.cts.AwsApplication : Starting AwsApplication on DESKTOP-1UQH9D with PID 8312
2021-04-21 20:43:24.674 INFO 8312 --- [ restartedMain] com.cts.AwsApplication : The following profiles are active: dev
2021-04-21 20:43:24.745 INFO 8312 --- [ restartedMain] .DevTools
2021-04-21 20:43:24.747 INFO 8312 --- [ restartedMain] .e.DevTools
2021-04-21 20:43:25.666 INFO 8312 --- [ restartedMain] .s.d.r.c.RequestMappingHandlerAdapter
2021-04-21 20:43:25.755 INFO 8312 --- [ restartedMain] .s.d.r.c.RequestMappingHandlerAdapter
2021-04-21 20:43:25.759 INFO 8312 --- [ restartedMain] .s.d.r.c.RequestMappingHandlerAdapter
2021-04-21 20:43:26.547 INFO 8312 --- [ restartedMain] .s.d.r.c.RequestMappingHandlerAdapter
2021-04-21 20:43:26.697 INFO 8312 --- [ restartedMain] .s.d.r.c.RequestMappingHandlerAdapter
2021-04-21 20:43:26.697 INFO 8312 --- [ restartedMain] .s.d.r.c.RequestMappingHandlerAdapter
2021-04-21 20:43:26.985 INFO 8312 --- [ restartedMain] .s.d.r.c.RequestMappingHandlerAdapter
2021-04-21 20:43:26.987 INFO 8312 --- [ restartedMain] .s.d.r.c.RequestMappingHandlerAdapter
2021-04-21 20:43:27.138 INFO 8312 --- [ restartedMain] .s.d.r.c.RequestMappingHandlerAdapter
2021-04-21 20:43:27.286 INFO 8312 --- [ restartedMain] .s.d.r.c.RequestMappingHandlerAdapter
2021-04-21 20:43:32.018 INFO 8312 --- [ restartedMain] .s.d.r.c.RequestMappingHandlerAdapter
2021-04-21 20:43:32.019 INFO 8312 --- [ restartedMain] .s.d.r.c.RequestMappingHandlerAdapter
2021-04-21 20:43:34.300 INFO 8312 --- [ restartedMain] .s.d.r.c.RequestMappingHandlerAdapter
2021-04-21 20:43:34.329 INFO 8312 --- [ restartedMain] .s.d.r.c.RequestMappingHandlerAdapter
2021-04-21 20:43:34.370 INFO 8312 --- [ restartedMain] .s.d.r.c.RequestMappingHandlerAdapter
2021-04-21 20:43:34.766 WARN 8312 --- [ restartedMain] .s.d.r.c.RequestMappingHandlerAdapter
2021-04-21 20:43:34.985 INFO 8312 --- [ restartedMain] .s.d.r.c.RequestMappingHandlerAdapter
2021-04-21 20:43:35.223 INFO 8312 --- [ restartedMain] .s.d.r.c.RequestMappingHandlerAdapter
2021-04-21 20:43:35.235 INFO 8312 --- [ restartedMain] com.cts.AwsApplication : Started AwsApplication in 0.05 seconds (JVM: 0.05s)
2021-04-21 20:43:40.442 INFO 8312 --- [nio-5000-exec-1] o.a.c.c.C.[
2021-04-21 20:43:40.442 INFO 8312 --- [nio-5000-exec-1] o.s.web.server
2021-04-21 20:43:40.442 INFO 8312 --- [nio-5000-exec-1] o.s.web.ser
2021-04-21 20:43:40.442 INFO 8312 --- [nio-5000-exec-1] o.s.web.ser
2021-04-21 20:45:37.073 INFO 8312 --- [ Thread-4] j.LocalContainer
2021-04-21 20:45:37.875 INFO 8312 --- [ Thread-4] com.zaxxer
2021-04-21 20:45:37.881 INFO 8312 --- [ Thread-4] com.zaxxer
2021-04-21 20:45:41.208 INFO 8312 --- [ Thread-4] com.zaxxer

```

Create a spring REST application using Spring Boot.



Start the application locally and type the url in the browser window as shown above

The screenshot shows the AWS Elastic Beanstalk console. On the left, there's a sidebar with 'Elastic Beanstalk' selected. The main area has three sections: 'Application information', 'Application tags', and 'Platform'. In the 'Application information' section, the 'Application name' field contains 'CDE1501-Rohit-EB'. Below it, there's a note about character limits. In the 'Application tags' section, there's a placeholder for adding tags and a note about applying up to 50 tags. The 'Platform' section is partially visible at the bottom.

We can see above screen that ELASTIC BEANSTALK is created

The screenshot shows a browser window displaying a JSON response. The URL is 'Not secure | rdsapp-env.eba-2hajhzn.us-east-2.elasticbeanstalk.com/employees'. The JSON data lists three employees with their details: Name, Gender, Age, and Salary. The salary values are highlighted in blue.

```

[{"id": 1, "name": "Panu", "gender": "Male", "age": 23, "salary": 34000}, {"id": 2, "name": "Chitra", "gender": "Female", "age": 21, "salary": 55000}, {"id": 4, "name": "Sneha", "gender": "Female", "age": 23, "salary": 45000}]

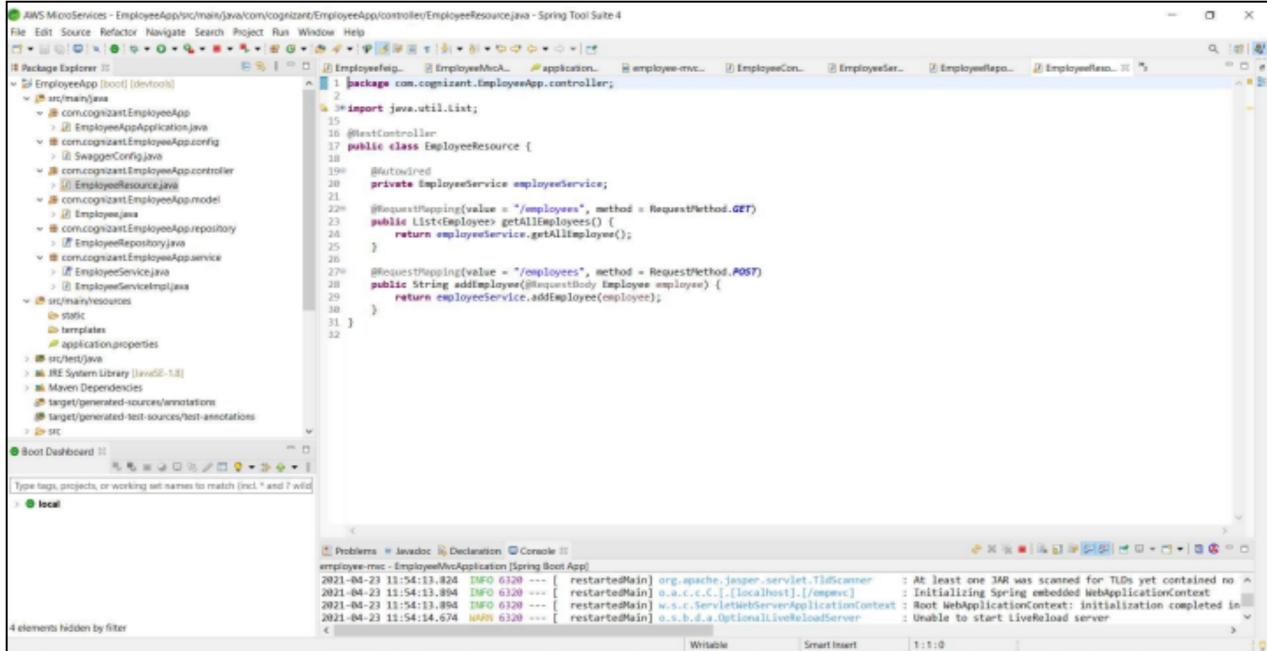
```

We can see that the record has been successfully inserted into the RDS database

DAY-FIVE

■ [Swagger-Hands-on]

Make use of Swagger to create documentation for RESTful/microservices.



Create a simple RESTful service using Spring BOOT

The screenshot shows a Maven dependency tree. A red box highlights the section of the tree where Swagger dependencies are listed. An arrow points from this highlighted area to the text 'Swagger Dependencies'.

```

<dependency>
  <groupId>org.projectlombok</groupId>
  <artifactId>lombok</artifactId>
  <optional>true</optional>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-test</artifactId>
  <scope>test</scope>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
<dependency>
  <groupId>io.springfox</groupId>
  <artifactId>springfox-swagger-ui</artifactId>
  <version>3.0.0</version>
</dependency>
<dependency>
  <groupId>io.springfox</groupId>
  <artifactId>springfox-swagger2</artifactId>
  <version>3.0.0</version>
</dependency>
</dependencies>

<build>
  <plugins>
    <plugin>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-maven-plugin</artifactId>
      <version>2.4.3</version>
    
```

Include the Swagger dependencies

```

1 package com.cognizant.EmployeeApp.config;
2
3 import org.springframework.context.annotation.Bean;
4 import org.springframework.context.annotation.Configuration;
5
6 import springfox.documentation.builders.RequestHandlerSelectors;
7 import springfox.documentation.spi.DocumentationType;
8 import springfox.documentation.spring.web.plugins.Docket;
9 import springfox.documentation.swagger2.annotations.EnableSwagger2;
10
11 @Configuration
12 @EnableSwagger2
13 public class SwaggerConfig {
14
15
16    @Bean
17    public Docket api() {
18        return new Docket(DocumentationType.SWAGGER_2).select()
19            .apis(RequestHandlerSelectors.basePackage("com.cognizant.EmployeeApp")).build();
20    }
21
22 }
23
24

```

Problems | Javadoc | Declaration | Console | Progress

```

EmployeeApp - EmployeeApp Application [Spring Boot App] [Program Filesets:4.9.0.RELEASE] [plugins/org.eclipse.jdt.core]:[http://127.0.0.1:8080/] [File: /src/main/java/com/cognizant/EmployeeApp/SwaggerConfig.java] [Line: 12, Column: 1]
2021-04-20 09:48:07,626 INFO [http-nio-8080-exec-1] [restartedMain] org.hibernate.Version : HHH000412: Hibernate ORM core version 5.4.30.Final
2021-04-20 09:48:07,626 INFO [http-nio-8080-exec-1] [restartedMain] o.h.Hibernate.annotations.common.Version : HCANN000001: Hibernate Commons Annotations (5.1.2.Final)
2021-04-20 09:48:08,247 INFO [http-nio-8080-exec-1] [restartedMain] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...
2021-04-20 09:48:08,595 INFO [http-nio-8080-exec-1] [restartedMain] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
2021-04-20 09:48:10,793 INFO [http-nio-8080-exec-1] [restartedMain] org.hibernate.dialect.Dialect : HHH000400: Using dialect: org.hibernate.dialect.MySQL5InnoDBDialect
2021-04-20 09:48:10,886 INFO [http-nio-8080-exec-1] [restartedMain] org.hibernate.dialect.Dialect : HHH000400: Using JtaPlatform implementation: [org.hibernate.dialect.MySQL5InnoDBDialect]
2021-04-20 09:48:13,993 INFO [http-nio-8080-exec-1] [restartedMain] o.h.e.t.j.p.i.JtaPlatformInitiator : HHH000040: Using JtaPlatform implementation: [org.hibernate.dialect.MySQL5InnoDBDialect]
2021-04-20 09:48:14,022 INFO [http-nio-8080-exec-1] [restartedMain] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'EmployeeApp'
2021-04-20 09:48:14,282 INFO [http-nio-8080-exec-1] [restartedMain] org.hibernate.boot.registry.StandardServiceRegistryBuilder : LiveReload server is running on port 35729
2021-04-20 09:48:16,417 INFO [http-nio-8080-exec-1] [restartedMain] org.hibernate.cfg.Configuration : spring.jpa.open-in-view is enabled by default. There are multiple reasons to disable it for production environments.
2021-04-20 09:48:17,465 INFO [http-nio-8080-exec-1] [restartedMain] o.s.s.concurrent.ThreadPoolTaskExecutor : Initializing ExecutorService "applicationTaskExecutor"
2021-04-20 09:48:18,179 INFO [http-nio-8080-exec-1] [restartedMain] o.s.b.w.embedded.tomcat.TomcatServer : Tomcat started on port(s): 8080 (http) with context path ''
2021-04-20 09:48:19,700 INFO [http-nio-8080-exec-1] [restartedMain] i.c.b.EmployeeApp.EmployeeAppApplication : Started EmployeeAppApplication in 20.848 seconds (JVM running for 21.022)
2021-04-20 09:48:12,542 INFO [http-nio-8080-exec-1] [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet "dispatcherServlet"
2021-04-20 09:48:32,542 INFO [http-nio-8080-exec-1] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet "dispatcherServlet"
2021-04-20 09:48:32,568 INFO [http-nio-8080-exec-1] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 4 ms

```

Create a Swagger configuration class as shown above

```

1 package com.cognizant.EmployeeApp.model;
2
3 import javax.persistence.Entity;
4 import javax.persistence.GeneratedValue;
5 import javax.persistence.Id;
6
7 import io.swagger.annotations.ApiModel;
8 import io.swagger.annotations.ApiModelProperty;
9 import lombok.AllArgsConstructor;
10 import lombok.Data;
11 import lombok.NoArgsConstructor;
12
13 @Entity
14 @Data
15 @AllArgsConstructor
16 @NoArgsConstructor
17 @ApiModel(description = "Employee class which is acting as the DTO")
18 public class Employee {
19
20    @Id
21    private int id;
22    @ApiModelProperty(propertyNotes = "Name should contain only Alphabets")
23    private String name;
24    @ApiModelProperty(propertyNotes = "Age should be between 18 and 60")
25    @ApiModelProperty(propertyNotes = "Age should be between 18 and 60")
26    private int age;
27    private double salary;
28 }
29

```

Swagger specific annotations to customize the descriptions of model class and the properties

Use Swagger specific annotations to customize the descriptions of model class and the properties

The screenshot shows the Postman application interface. The left sidebar includes 'My Workspace' with sections for Collections, APIs, Environments, Mock Servers, Monitors, and History. A central area displays a cartoon character holding a briefcase icon, with the text 'You don't have any collections'. Below this, a section titled 'Collections' explains how they group related requests. A large button labeled 'Create Collection' is present. The main workspace shows a request for 'http://localhost:8080/v2/api-docs' with a status of '200 OK'. The response body is a JSON document representing an API specification:

```
1  "swagger": "2.0",
2  "info": {
3    "description": "Api Documentation",
4    "version": "1.0",
5    "title": "Api Documentation",
6    "termsOfService": "urn:tos",
7    "contact": {},
8    "license": {
9      "name": "Apache 2.0",
10     "url": "http://www.apache.org/licenses/LICENSE-2.0"
11   }
12 },
13 "host": "localhost:8080",
14 "basePath": "/",
15 "tags": [
16   {
17     "name": "employees-resource",
18     "description": "Employee Resource"
19   }
20 ],
21 "paths": {
22   "/employees": {
23     "get": {}
24   }
25 }
```

Visit the api endpoint “localhost:8080/v2/api-docs” and we can see the complete API documentation of our service.

Api Documentation

Api Documentation

[Apache 2.0](#)

swagger-api-controller : Swagger API Controller

ShowHide | List Operations | Expand Operations

Method	Path	Description
GET	/employees	getEmployee
POST	/employees	addEmployee

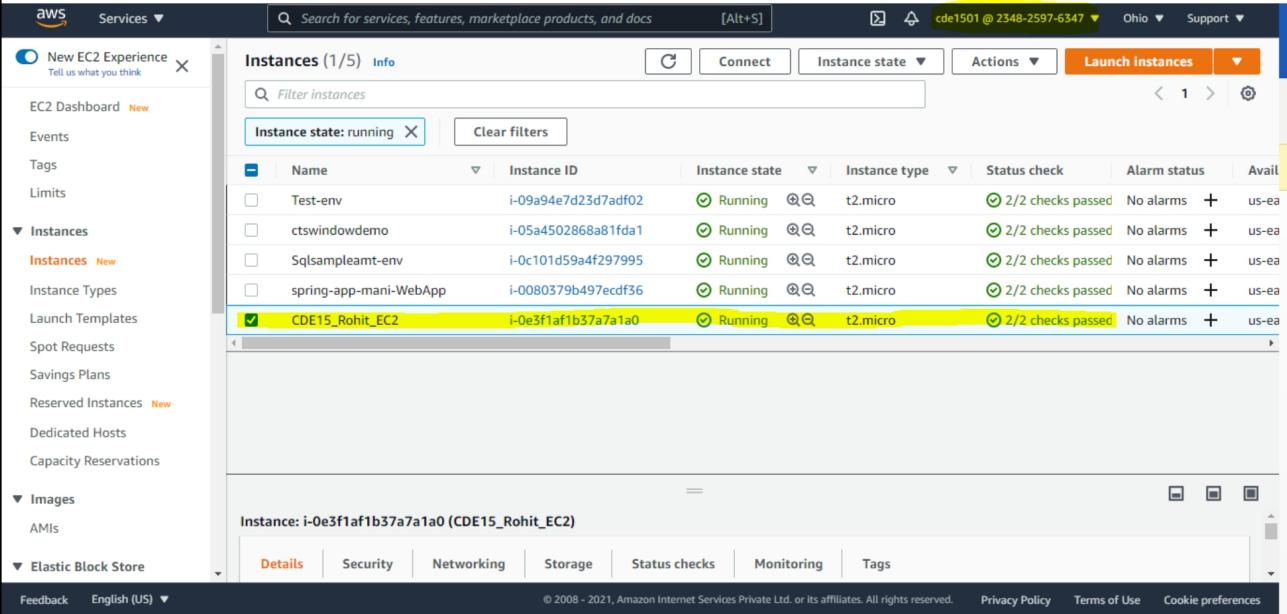
[BASE URL: / , API VERSION: 1.0]

Hit the URL in our web browser and see the Swagger API functionalities

DAY-SIX

▪ [Spring MVC Client For Spring REST Service]

Create an EC2 instance, connect to it from your local system and install apache web server on the EC2 instance.



The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with options like New EC2 Experience, EC2 Dashboard, Events, Tags, Limits, Instances (selected), Images, and Elastic Block Store. The main area displays a table of instances with columns: Name, Instance ID, Instance state, Instance type, Status check, Alarm status, and Avail. There are five instances listed: Test-env, ctswindowdemo, Sqlsampleamt-env, spring-app-mani-WebApp, and CDE15_Rohit_EC2. The last instance is highlighted with a yellow background. Below the table, there's a section for the selected instance 'CDE15_Rohit_EC2' with tabs for Details, Security, Networking, Storage, Status checks, Monitoring, and Tags.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Avail
Test-env	i-09a94e7d23d7adf02	Running	t2.micro	2/2 checks passed	No alarms	+ us-ea
ctswindowdemo	i-05a4502868a81fda1	Running	t2.micro	2/2 checks passed	No alarms	+ us-ea
Sqsampleamt-env	i-0c101d59a4f297995	Running	t2.micro	2/2 checks passed	No alarms	+ us-ea
spring-app-mani-WebApp	i-0080379b497ecdf36	Running	t2.micro	2/2 checks passed	No alarms	+ us-ea
CDE15_Rohit_EC2	i-0e3f1af1b37a7a1a0	Running	t2.micro	2/2 checks passed	No alarms	+ us-ea

This indicates my instance is up and running

```
import java.math.*;
import java.security.*;
import java.text.*;
import java.util.*;
import java.util.concurrent.*;
import java.util.regex.*;
```