Phase 5: CI CD Deployment For Spring Boot Application

Developer Details:

Name: Javeriya Soudagar

Email: javeriyamehreen143@gmail.com

Date created: 022-05-2022

Program Name: **CI CD Deployment Spring Boot**

GitHub Repository:

https://github.com/JaveriyaSoudagar/project5

Program Background :

As the project is in the final stage, management has asked you to automate the integration and deployment of the web application. You are required to set up an environment where the application will be hosted and accessed by users. The source code is supposed to be fetched from a GitHub repository.

Program Features:

- A search form in the home page to allow entry of the ID and Name of the Student
- Based on the details entered, it will show available list of students.

- Once a person selects an item to purchase, they will be redirected to the list of available items.
- In the next page, they are shown the complete list of details in the database in XML and JSON format.

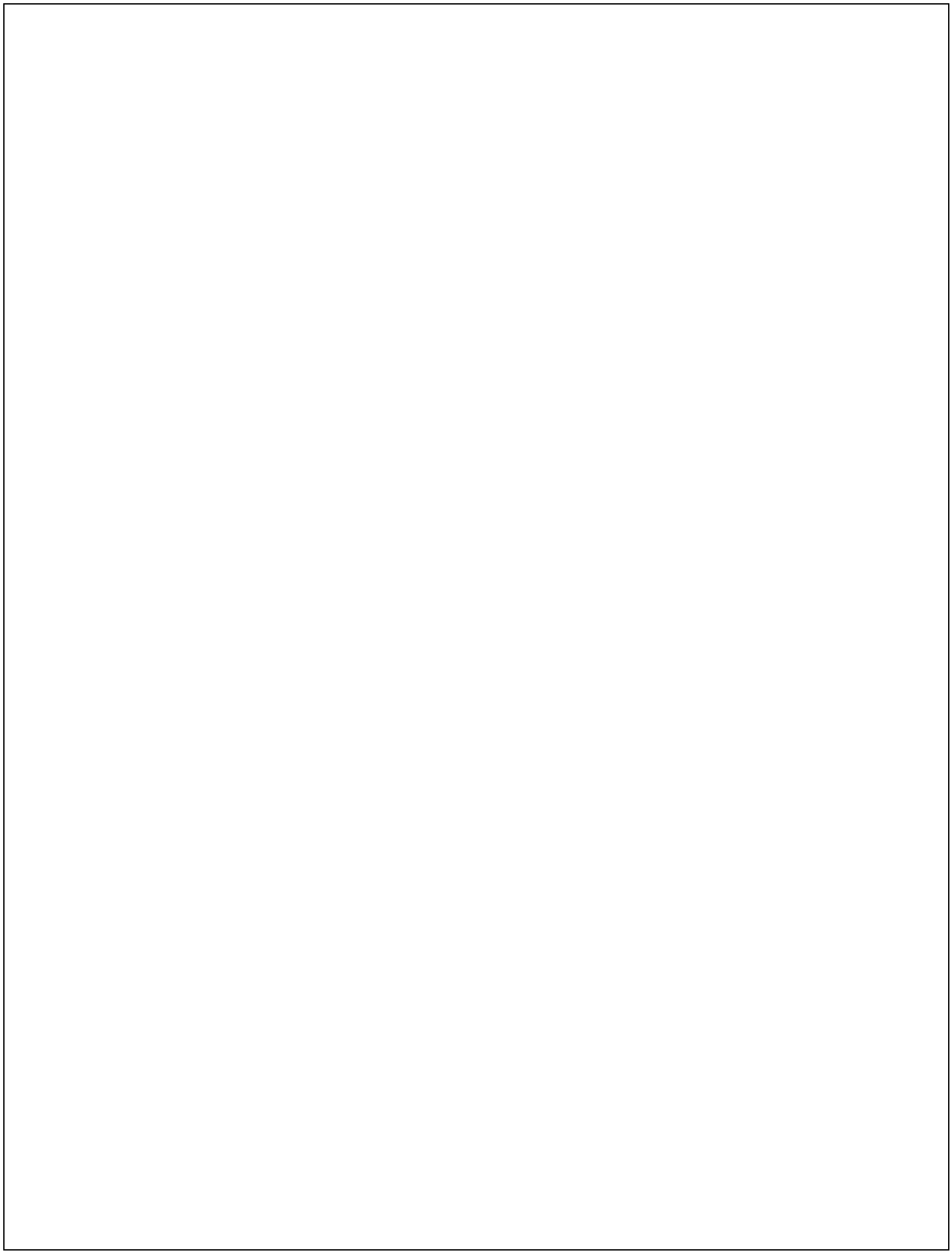
- There will be an admin to manage the Database. And can be able to delete and create the entries using the H2 Database which is inMemory Database.

The admin will be able to change his password if he wants, he should be able to:

- Manage the entries including categorizing them

Tools used for development :

1. Eclipse IDE
2. H2 Database
3. Apache Tom-Cat 10 server
4. Spring Boot
5. Amazon Web Services (AWS)
6. HTML
7. Apache Maven
8. Putty and EC2 Virtual Machine
9. GitHub



Sprint Table:

SPRINT	WORK DONE	TIME PERIOD	RESULT
1	Spring Boot Application coding	04/05/2022 to 05/05/2022	Done ✓
2	Designed HTML pages	05/05/2022 to 06/05/2022	Done ✓
3	Deploying on AWS - EC2	06/05/2022 to 07/05/22	Done ✓
4	Creating CI CD pipeline	07/05/22 to 08/05/22	Done ✓

Source codes

1.SpringBootDataJPARestApplication.java

```
package com.boot.demo;
 import org.springframework.boot.SpringApplication; import
org.springframework.boot.autoconfigure.SpringBootApplication;
import
org.springframework.web.bind.annotation.RestController;
@SpringBootApplication @RestController public
class SpringBootDataJpaRestApplication {

    public static void main(String[] args) {
        SpringApplication.run(SpringBootDataJpaRestApplication.
class, args);
    }

}
```

2.Home Controller.java

```
package com.boot.demo;
 import
java.util.List;
 import
java.util.Optional;
 import
org.springframework.beans.factory.annotation.Autowired; import
org.springframework.stereotype.Controller; import
org.springframework.web.bind.annotation.DeleteMapping; import
org.springframework.web.bind.annotation.PathVariable; import
org.springframework.web.bind.annotation.PostMapping; import
org.springframework.web.bind.annotation.PutMapping; import
org.springframework.web.bind.annotation.RequestBody; import
org.springframework.web.bind.annotation.RequestMapping; import
org.springframework.web.bind.annotation.RequestParam; import
org.springframework.web.bind.annotation.ResponseBody; import
org.springframework.web.servlet.ModelAndView;

@Controller public class
HomeController {

    @Autowired
    StudentRepo repo;

    @RequestMapping("/")
public  String  home()  {
return "home";
    }

    @RequestMapping("/addStudent")     public
String addStudent(Student student) {
repo.save(student);         return "home";
    }
```

```
    @RequestMapping("/getData")
    public   ModelAndView   getData(@RequestParam   int   id)   {
ModelAndView mv = new ModelAndView("showData");
        Student student = repo.findById(id).orElse(new
Student());          mv.addObject(student);
return mv;
    }

    @RequestMapping("/students")
    @ResponseBody            public   String
students()   {                        return
repo.findAll().toString();
    }

    @RequestMapping("/students/{id}")
    @ResponseBody     public String
studentsByID(@PathVariable("id") int id) {          return
repo.findById(id).toString();
    }

    @RequestMapping("/studentsList")
    @ResponseBody      public List<Student>
studentsList()  {                     return
repo.findAll();
    }

    @RequestMapping("/studentsIDList/{id}")
    @ResponseBody     public Optional<Student>
studentsByIDList(@PathVariable("id") int id) {
return repo.findById(id);     }
```

```
    @PostMapping("/students")     public Student
studentsInsert(@RequestBody Student student)
{
repo.save(student);
return student;
    }

    @DeleteMapping("/students/{id}")     public Student
studentsDelete(@PathVariable("id") int id) {
        @SuppressWarnings("deprecation")
Student student=repo.getOne(id);
repo.delete(student);         return
student;
    }

    @PutMapping(path="/students", consumes =
{"application/json"})     public Student
studentsUpdate(@RequestBody Student student)
{
repo.save(student);
return student;
    }
}
```

3. Student.java

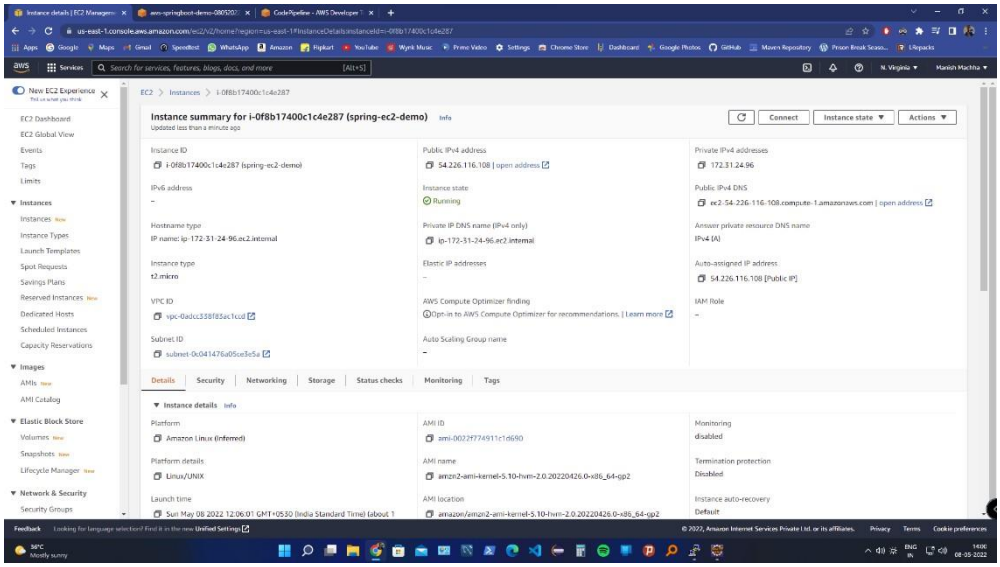
```
package com.boot.demo;
 import
javax.persistence.Entity; import
javax.persistence.Id;
 @Entity public class
Student {
        @Id     private
int id;     private
String name;
     public int getId()
{         return id;
    }      public void
setId(int id) {         this.id
= id;
    }      public String
getName() {         return
name;
    }      public void setName(String
name) {         this.name = name;
    }

    @Override     public String toString() {         return
"Student [id=" + id + ", name=" + name + "]";     }


}
```

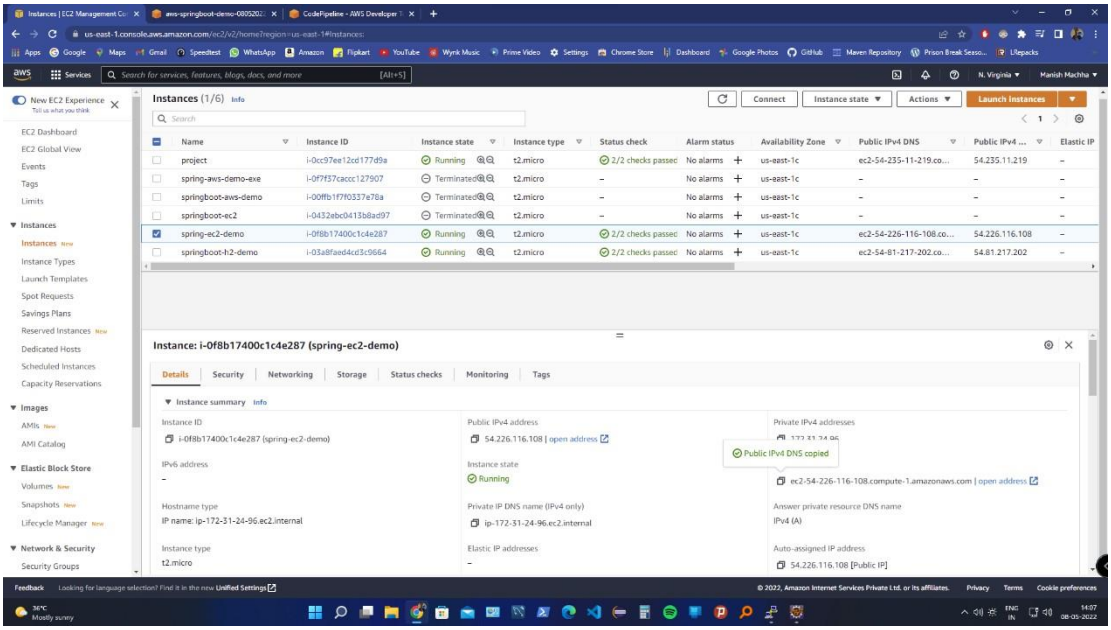
4. StudentRepo.java

```
package com.boot.demo;
 import org.springframework.data.jpa.repository.JpaRepository;

public interface StudentRepo extends JpaRepository<Student,
Integer>{

}
```

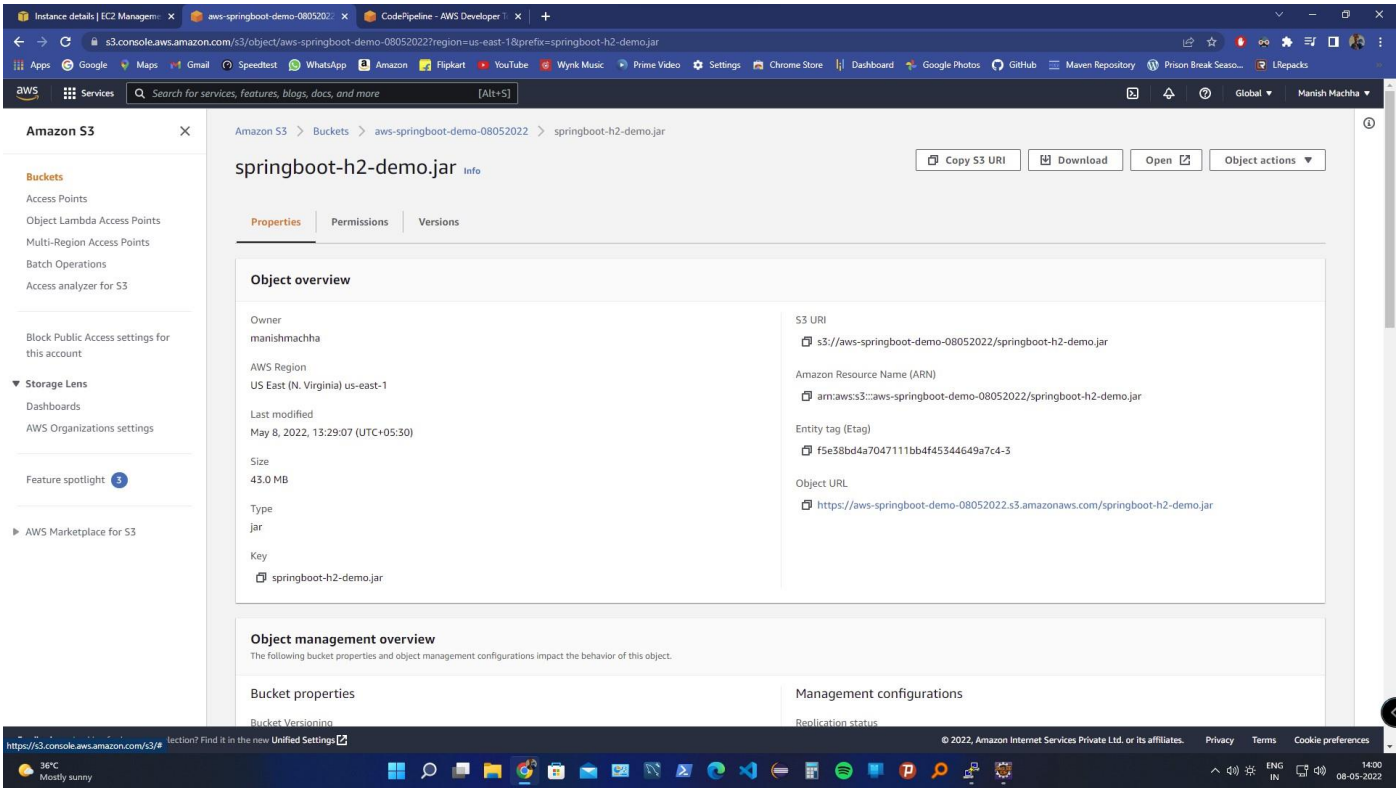
SCREENSHOTS

1. EC2 Instance

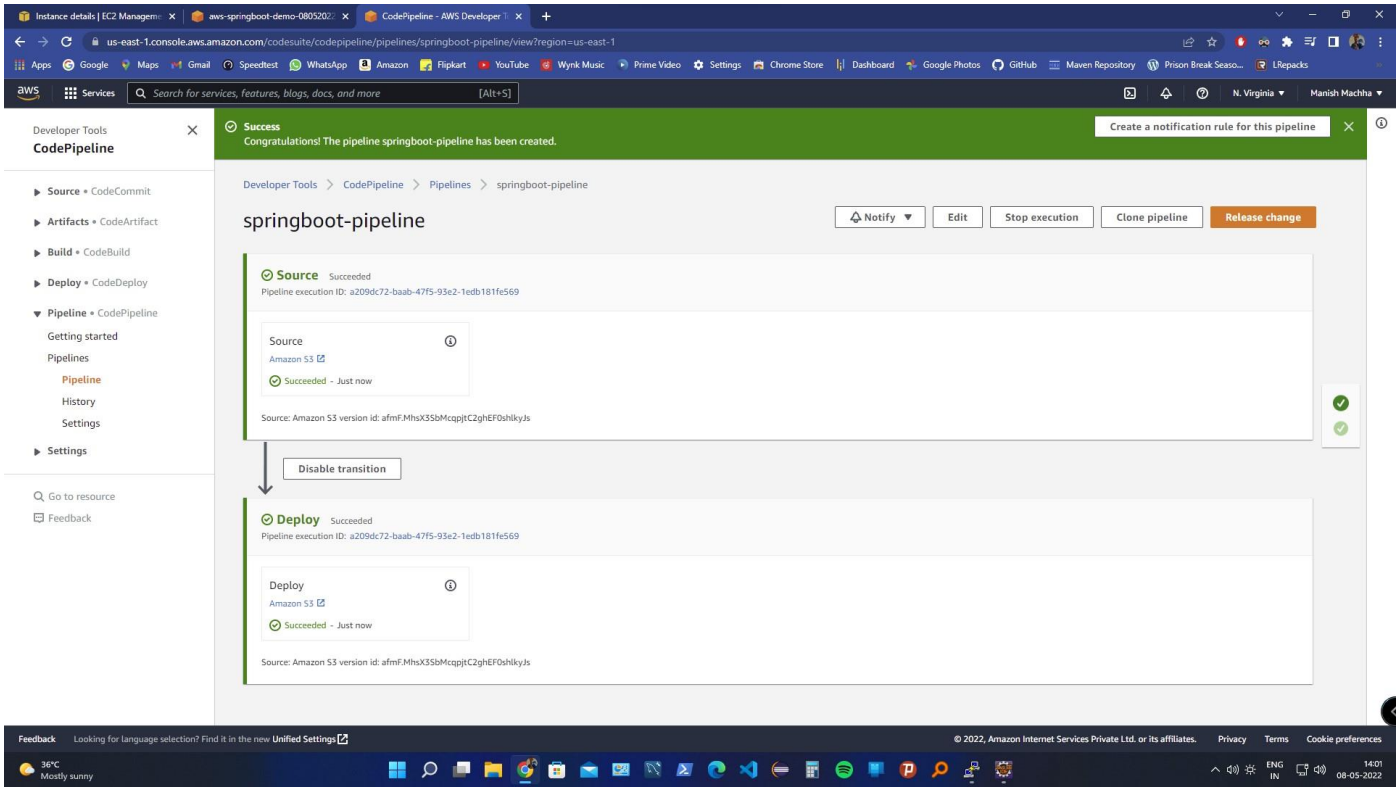




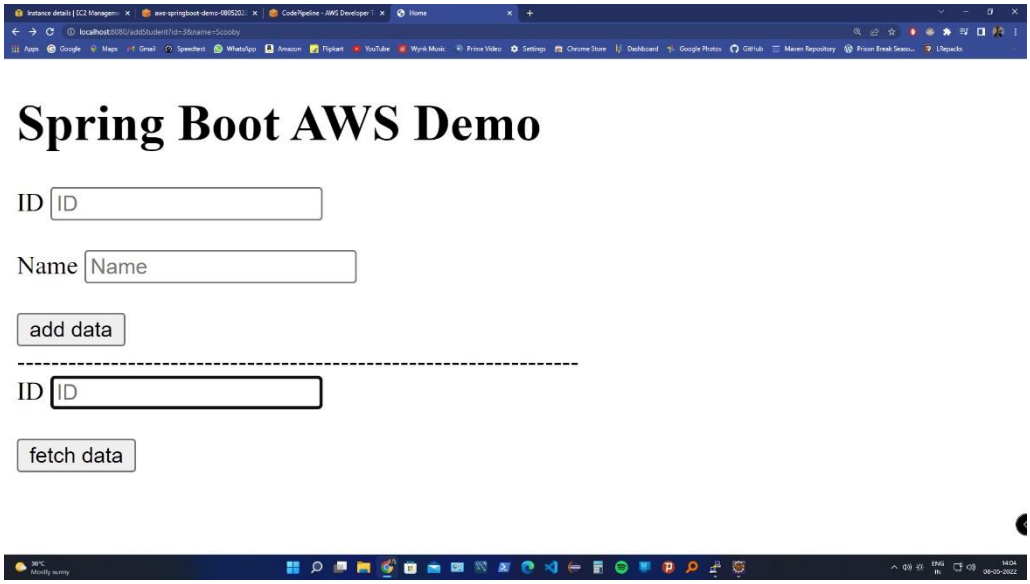
2. S3 Bucket

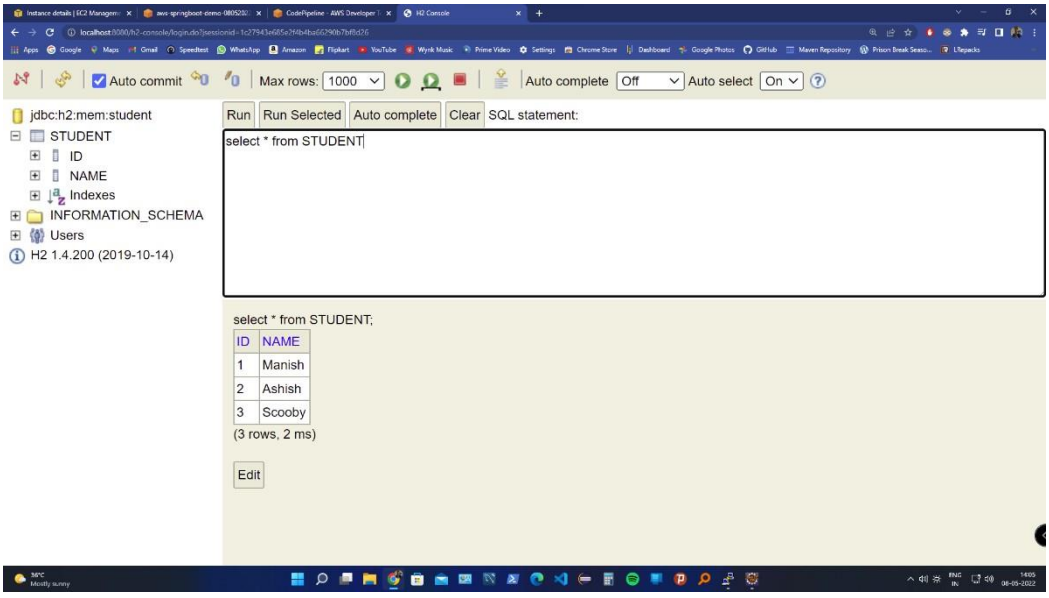


3. CI CD pipeline



4. Application Pages


Spring Boot AWS Demo
ID
ID
Name
Name
add data
ID
ID
fetch data


Auto commit
Max rows: 1000
Auto complete Off
Auto select On
jdbc:h2:mem:student
STUDENT
ID
NAME
Indexes
INFORMATION_SCHEMA
Users
H2 1.4.200 (2019-10-14)
Run
Run Selected
Auto complete
Clear
SQL statement:
select * from STUDENT
select * from STUDENT;
ID NAME
1 Manish
2 Ashish
3 Scooby
(3 rows, 2 ms)
Edit