

Java Full Stack Development

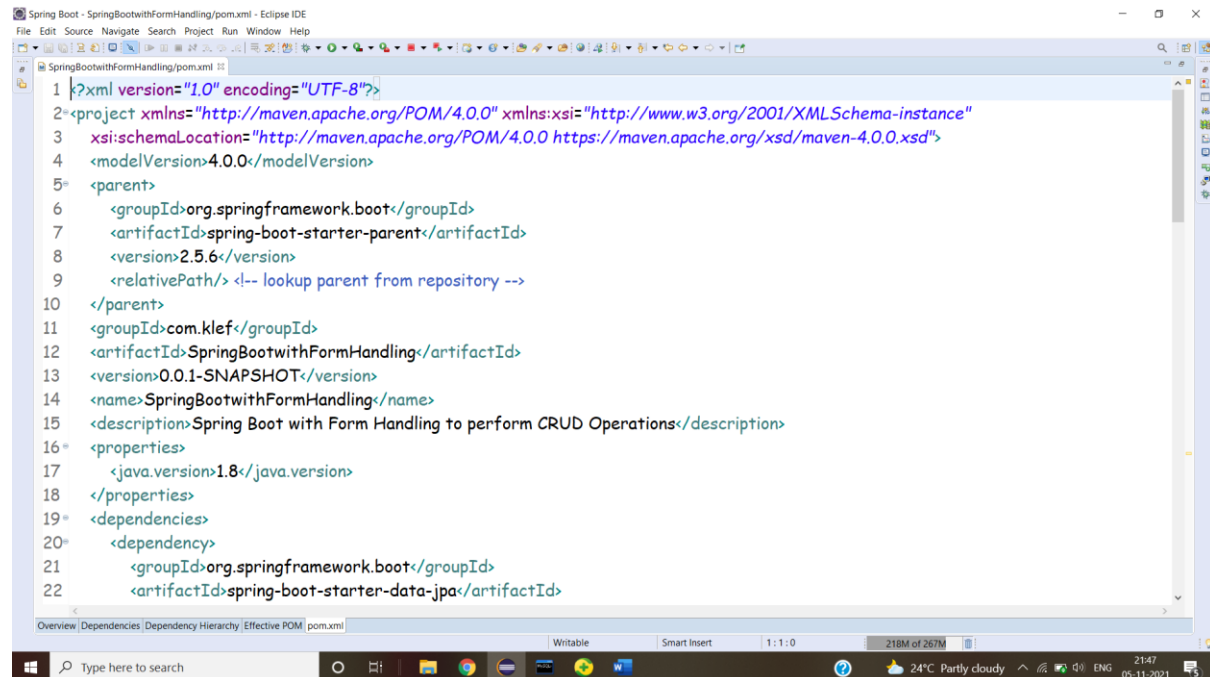
Skilling-10

ID NO: 190030677

Name: K. Sravani

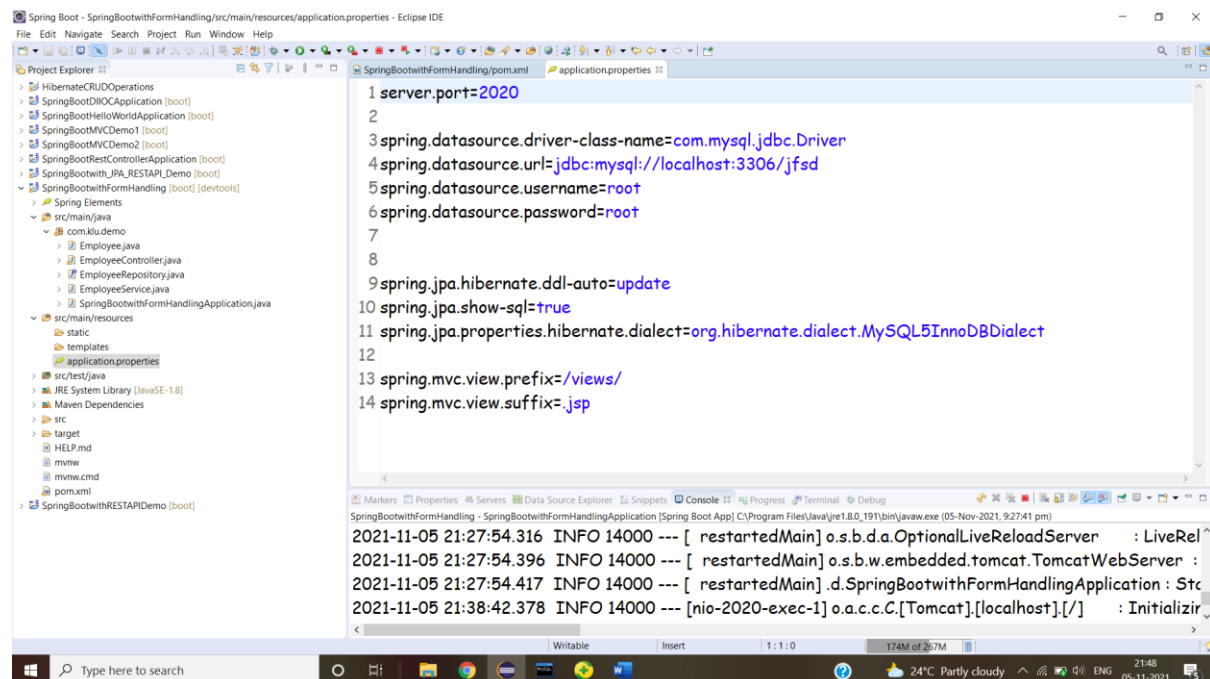
Spring Boot with Form Handling

Pom.xml



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
4   <modelVersion>4.0.0</modelVersion>
5   <parent>
6     <groupId>org.springframework.boot</groupId>
7     <artifactId>spring-boot-starter-parent</artifactId>
8     <version>2.5.6</version>
9     <relativePath/> <!-- lookup parent from repository -->
10  </parent>
11  <groupId>com.klef</groupId>
12  <artifactId>SpringBootwithFormHandling</artifactId>
13  <version>0.0.1-SNAPSHOT</version>
14  <name>SpringBootwithFormHandling</name>
15  <description>Spring Boot with Form Handling to perform CRUD Operations</description>
16  <properties>
17    <java.version>1.8</java.version>
18  </properties>
19  <dependencies>
20    <dependency>
21      <groupId>org.springframework.boot</groupId>
22      <artifactId>spring-boot-starter-data-jpa</artifactId>
```

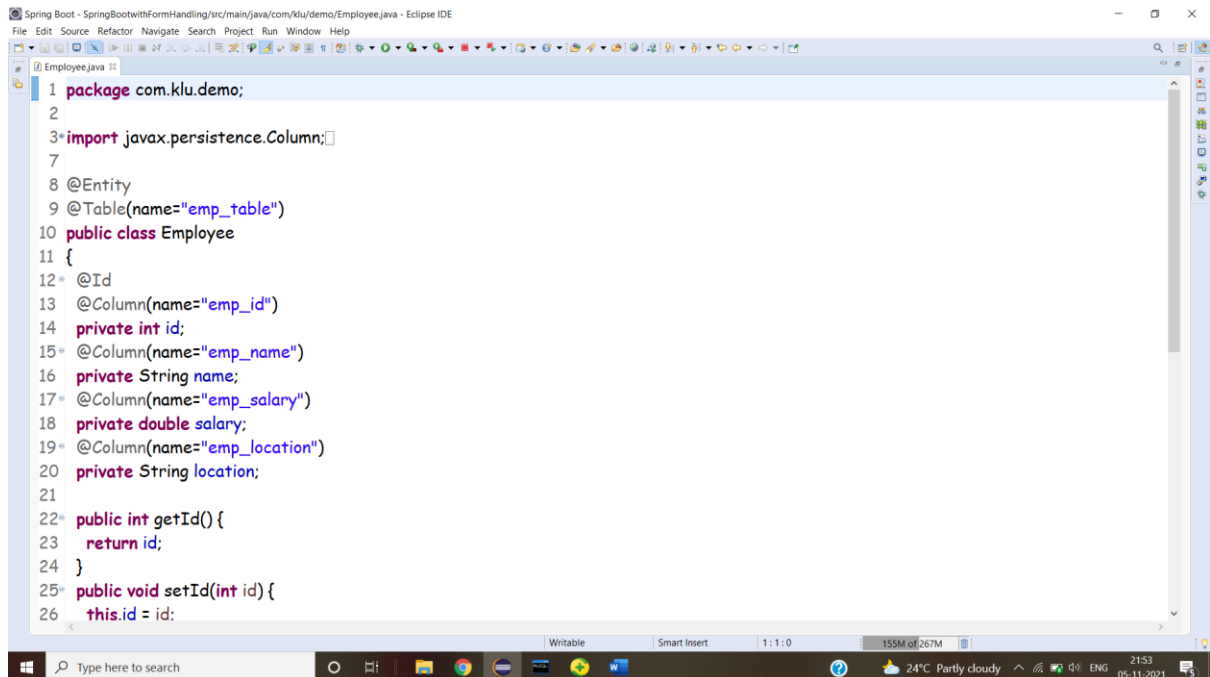
application.properties



```
1 server.port=2020
2
3 spring.datasource.driver-class-name=com.mysql.jdbc.Driver
4 spring.datasource.url=jdbc:mysql://localhost:3306/jfsd
5 spring.datasource.username=root
6 spring.datasource.password=root
7
8
9 spring.jpa.hibernate.ddl-auto=update
10 spring.jpa.show-sql=true
11 spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5InnoDBDialect
12
13 spring.mvc.view.prefix=/views/
14 spring.mvc.view.suffix=.jsp
```

2021-11-05 21:27:54.316 INFO 14000 --- [restartedMain] o.s.b.a.OptionalLiveReloadServer : LiveRel
2021-11-05 21:27:54.396 INFO 14000 --- [restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer :
2021-11-05 21:27:54.417 INFO 14000 --- [restartedMain] .d.SpringBootwithFormHandlingApplication : Stc
2021-11-05 21:38:42.378 INFO 14000 --- [nio-2020-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializ

Employee.java

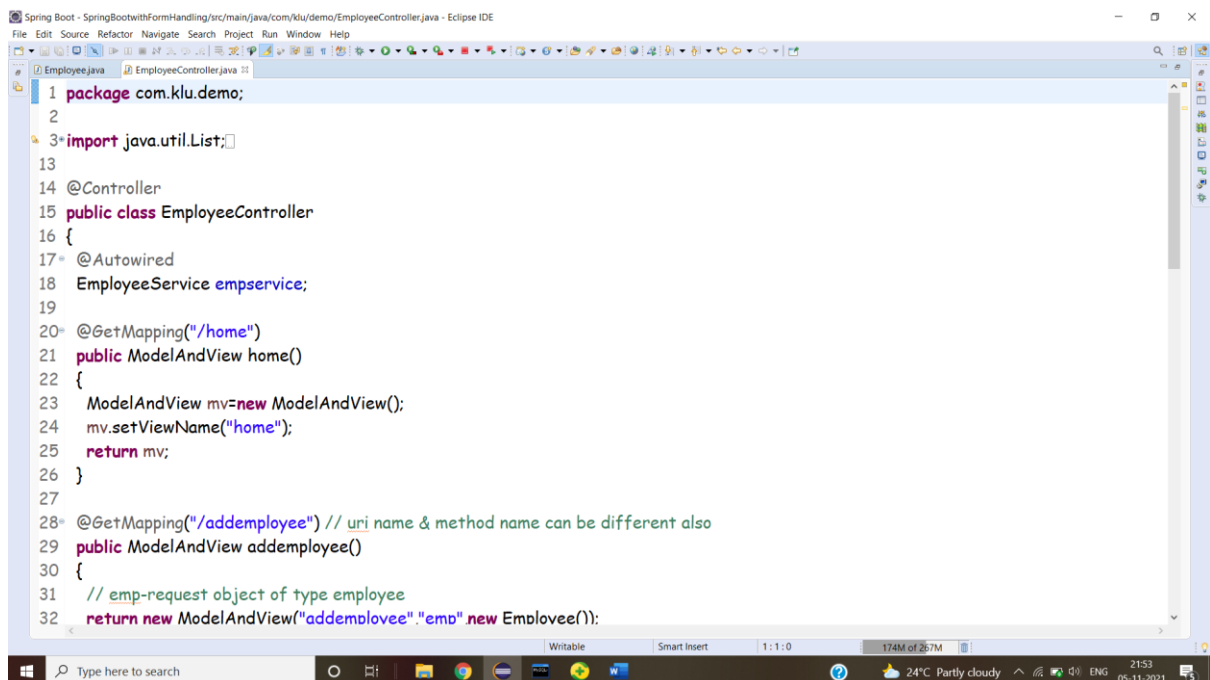


The screenshot shows the Eclipse IDE with the file 'Employee.java' open. The code is as follows:

```
1 package com.klu.demo;
2
3 import javax.persistence.Column;
4
5 @Entity
6 @Table(name="emp_table")
7 public class Employee
8 {
9     @Id
10    @Column(name="emp_id")
11    private int id;
12    @Column(name="emp_name")
13    private String name;
14    @Column(name="emp_salary")
15    private double salary;
16    @Column(name="emp_location")
17    private String location;
18
19    public int getId() {
20        return id;
21    }
22    public void setId(int id) {
23        this.id = id;
24    }
25 }
```

The IDE interface includes a menu bar (File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help), a toolbar, and a status bar at the bottom showing system information like temperature and time.

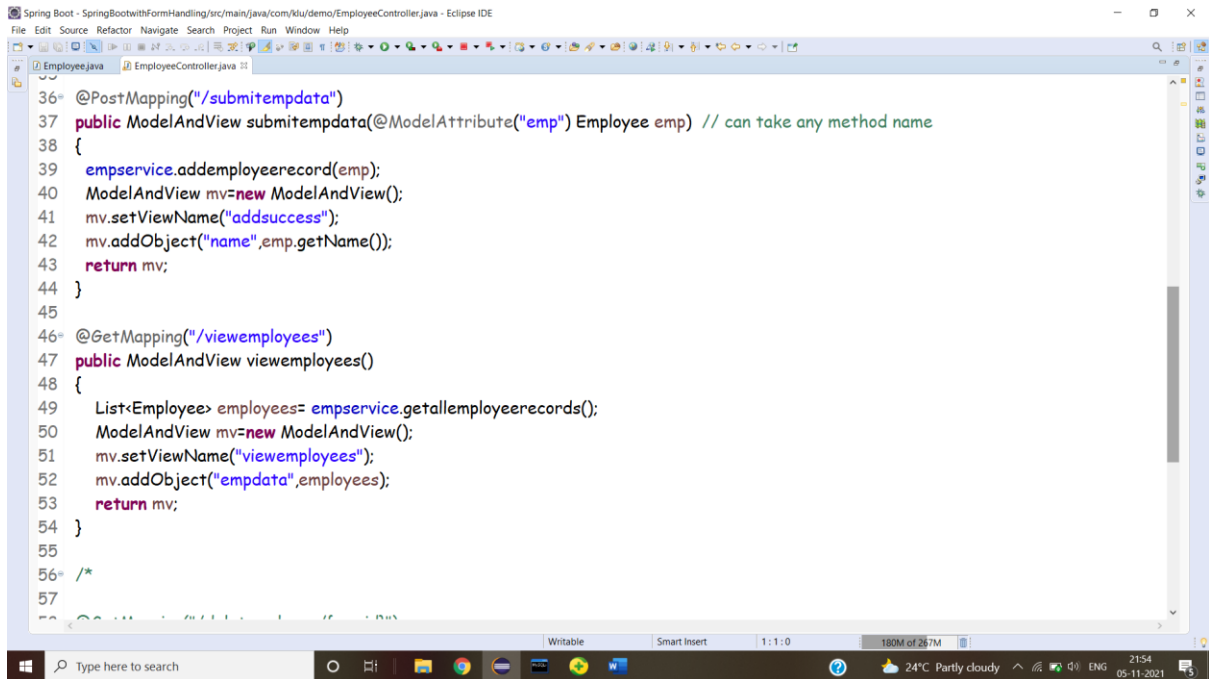
EmployeeController.java



The screenshot shows the Eclipse IDE with the file 'EmployeeController.java' open. The code is as follows:

```
1 package com.klu.demo;
2
3 import java.util.List;
4
5 @Controller
6 public class EmployeeController
7 {
8     @Autowired
9     EmployeeService empService;
10
11     @GetMapping("/home")
12     public ModelAndView home()
13     {
14         ModelAndView mv=new ModelAndView();
15         mv.setViewName("home");
16         return mv;
17     }
18
19     @GetMapping("/addemployee") // uri name & method name can be different also
20     public ModelAndView addemployee()
21     {
22         // emp-request object of type employee
23         return new ModelAndView("addemployee".emp new Employee());
24     }
25 }
```

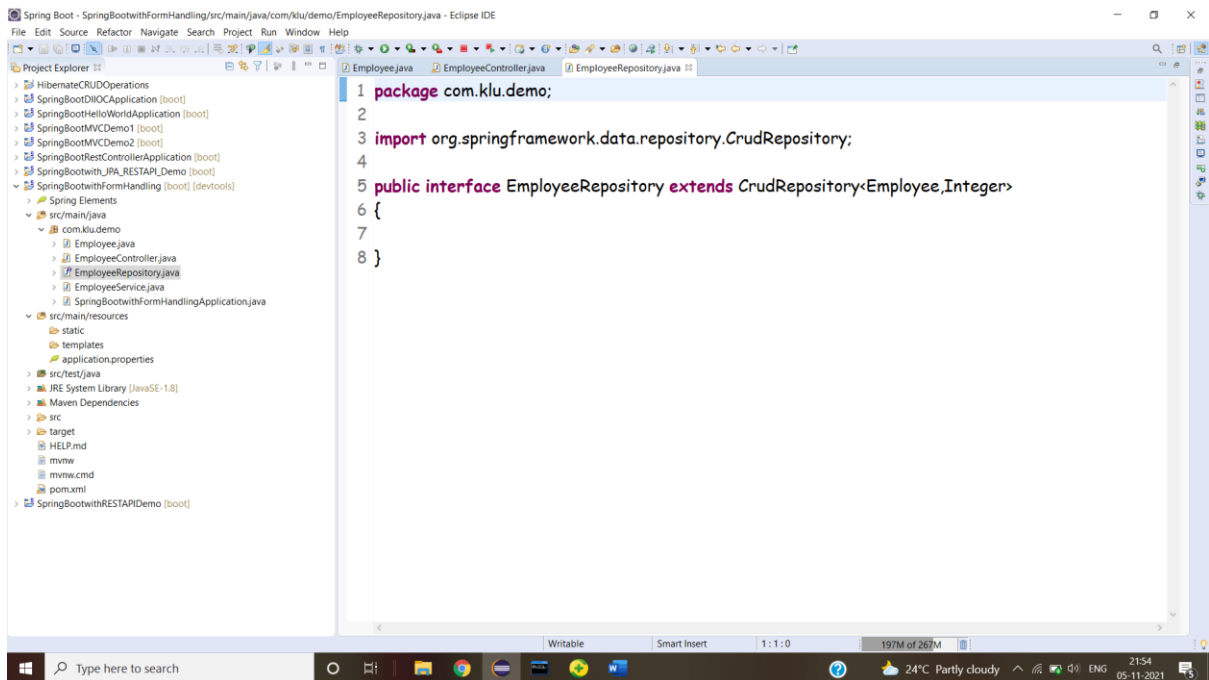
The IDE interface is similar to the previous screenshot, showing the same menu bar and toolbar, with the status bar at the bottom.



The screenshot shows the Eclipse IDE with the `EmployeeController.java` file open. The code defines two REST endpoints: `submitempdata` (POST) and `viewemployees` (GET). The `submitempdata` method takes an `Employee` object and returns a `ModelAndView` with the view name `addsuccess` and the employee's name. The `viewemployees` method returns a `ModelAndView` with the view name `viewemployees` and a list of all employees. The IDE interface includes a menu bar, a toolbar, and a status bar at the bottom showing system information like temperature and date.

```
36 @PostMapping("/submitempdata")
37 public ModelAndView submitempdata(@ModelAttribute("emp") Employee emp) // can take any method name
38 {
39     empService.addemployeerecord(emp);
40     ModelAndView mv=new ModelAndView();
41     mv.setViewName("addsuccess");
42     mv.addObject("name",emp.getName());
43     return mv;
44 }
45
46 @GetMapping("/viewemployees")
47 public ModelAndView viewemployees()
48 {
49     List<Employee> employees= empService.getallemployeerecords();
50     ModelAndView mv=new ModelAndView();
51     mv.setViewName("viewemployees");
52     mv.addObject("empdata",employees);
53     return mv;
54 }
55
56 /*
57
```

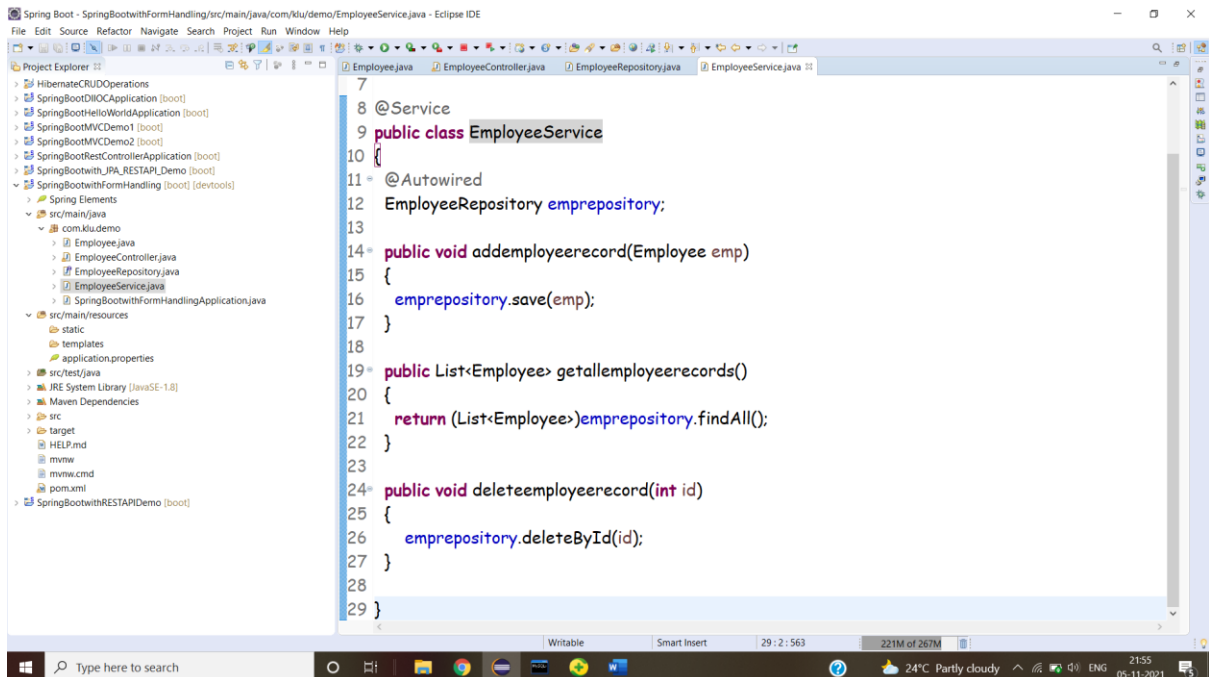
EmploteeRepository.java



The screenshot shows the Eclipse IDE with the `EmployeeRepository.java` file open. The code defines a `CrudRepository` for `Employee` entities. The IDE interface includes a Project Explorer on the left showing the project structure, a menu bar, a toolbar, and a status bar at the bottom showing system information like temperature and date.

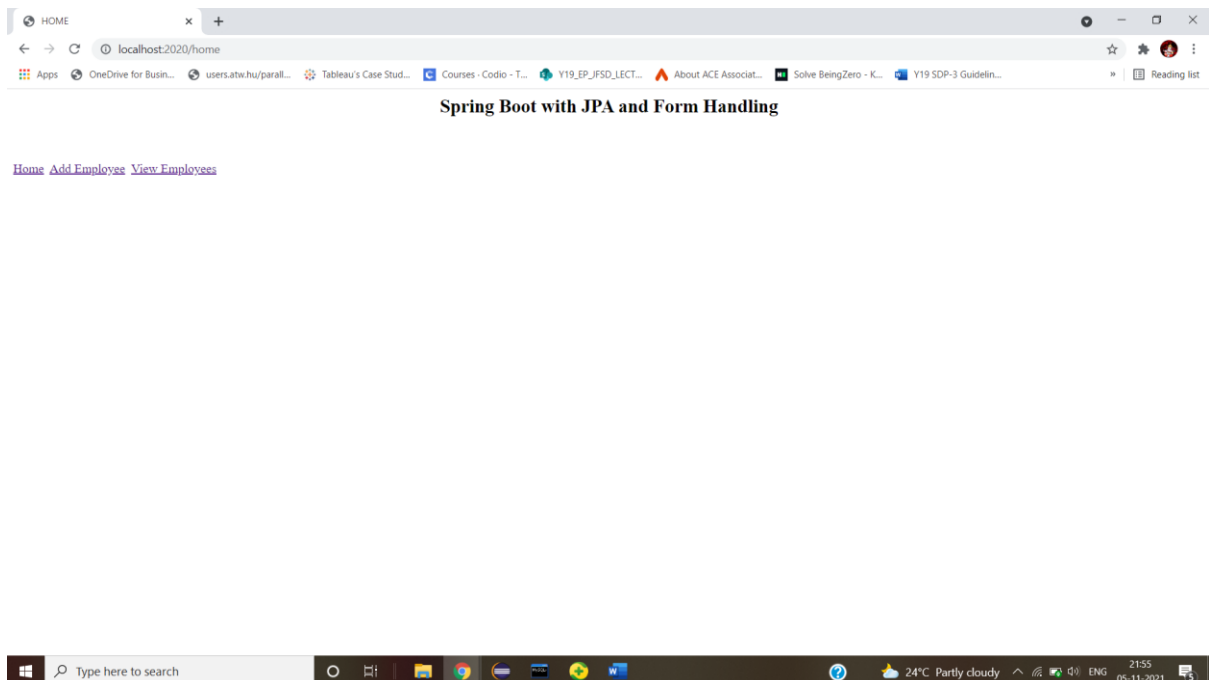
```
1 package com.klu.demo;
2
3 import org.springframework.data.repository.CrudRepository;
4
5 public interface EmployeeRepository extends CrudRepository<Employee,Integer>
6 {
7 }
8
```

EmployeeService.java



```
7
8 @Service
9 public class EmployeeService
10 {
11     @Autowired
12     EmployeeRepository empRepository;
13
14     public void addEmployeeRecord(Employee emp)
15     {
16         empRepository.save(emp);
17     }
18
19     public List<Employee> getAllEmployeeRecords()
20     {
21         return (List<Employee>)empRepository.findAll();
22     }
23
24     public void deleteEmployeeRecord(int id)
25     {
26         empRepository.deleteById(id);
27     }
28
29 }
```

Output Screenshots:





Spring Boot with JPA and Form Handling

[Home](#) [Add Employee](#) [View Employees](#)

Add Employee Record

Enter Employee ID

Enter Employee Name

Enter Employee Salary

Enter Employee Location



Hey Siri, Employee Record Added Successfully

To add another record [click here](#)



```
MySQL Command Line Client
1 row in set (0.00 sec)

mysql> select * from emp_table;
+-----+-----+-----+-----+
| emp_id | emp_location | emp_name | emp_salary |
+-----+-----+-----+-----+
| 30 | Tenali | Siri | 75000 |
+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql>
```



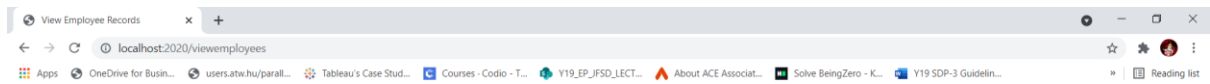
Spring Boot with JPA and Form Handling

[Home](#) [Add Employee](#) [View Employees](#)

View All Employee Records

ID	Name	Salary	Location	Action
30	Siri	75000.0	Tenali	Delete





Spring Boot with JPA and Form Handling

[Home](#) [Add Employee](#) [View Employees](#)

View All Employee Records

ID	Name	Salary	Location	Action
----	------	--------	----------	--------

