Code

index.html

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">

<head>
<link
      href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootst
rap.min.css"
      rel="stylesheet">
<link rel="stylesheet"
      href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.css">
<link rel="shortcut icon"
      href="https://cdn-icons-png.flaticon.com/512/470/470199.png">
<meta charset="UTF-8">
<title>EMS App</title>
</head>

<style>
body {
      overflow-x: hidden;
      scroll-behavior: smooth;
}

nav {
      background-color: navy !important;
}

.navbar-brand {
      color: whitesmoke !important;
      font-size: 30px !important;
      font-weight: bold;
}

.card {

box-shadow: rgba(14, 30, 37, 0.12) 1px 2px 2px 0px,
            rgba(14, 30, 37, 0.32) 1px 2px 2px 0px;
}
.card-body {
      text-align: center;
}
```

```
.table-responsive {
      box-shadow: rgba(60, 64, 67, 0.3) 0px 1px 2px 0px,
            rgba(60, 64, 67, 0.15) 0px 2px 6px 2px;
}
th {
      text-align: center;
      font-size: 18px !important;
}

td {
      text-align: center;
      font-size: 15px;
      font-weight: 500;
}

label {
      font-weight: 500;
}
</style>

<body>

      <nav class="navbar navbar-expand-sm navbar-light mb-5">
            <div class="container">
                  <a class="navbar-brand text-bold" href="#"><i
                        class="fa fa-align-center" aria-
hidden="true">EMS</i></a>
            </div>
      </nav>

      <main>

<div class="container p-4">
                  <div class="head_section">
                        <div class="row row-cols-1 row-cols-md-4 g-3">
                              <div class="col">
                                    <a href="#exampleModalToggle1" data-bs-
toggle="modal"
                                          role="button" style="text-
decoration: none; color: white;">
                                          <div class="card h-100 bg-
success">
                                                <div class="card-body">
                                                      <h5 class="text-light">
                                                            <i class="fa fa-
plus"></i>Add Employee
```

```
</h5>
</div>
      </div>
      </a>
      </div>
<div class="col">
<a href="#exampleModalToggle2" data-bs-toggle="modal"
                                        role="button" style="text-
decoration: none; color: white;">
<div class="card h-100 bg-primary">
<div class="card-body">
<h5 class="text-light">
<i class="fa fa-area-chart" aria-hidden="true"></i>Update
                                                          Employee
                </h5>
           </div>
      </div>
</a>
</div>
<div class="col">
<a href="#exampleModalToggle3" data-bs-toggle="modal"
role="button" style="text-decoration: none; color: white;">

<div class="card h-100 bg-danger">
      <div class="card-body">
      <h5 class="text-light">
      <i class="fa fa-trash"></i>Delete Employee
           </h5>
                </div>
      </div>
      </a>
           </div>
      <div class="col">
<a href="#deleteAllModal4" data-bs-toggle="modal" role="button"
      style="text-decoration: none; color: white;">    <div class="card
h-100 bg-dark">
      <div class="card-body">
      <h5 class="text-light">
<i class="fa fa-warning"></i>Delete All
      </h5> </div>     </div>
      </a>
      </div>
      </div>
      </div>
<br>
<div class="items_table mt-5 mb-4">
<div class="table-responsive p-2">
```

```
<h4 class="text-center p-2 class=" p-1" mt-2" style="font-family: 'Times
New Roman', Times, serif; font-weight: bold;">Employee Management
System</h4>
<table class="table table-bordered table-hover mt-5">

          </thead>
     <tbody>
<tr th:each="employee, index : ${employees}">
<td th:text="${index.index + 1}"></td>
<td th:text="${employee.id}"></td>
<td th:text="${employee.employeeName}"></td>
<td th:text="${employee.employeeEmail}"></td>
<td th:text="${employee.employeePhone}"></td>
<td th:text="${employee.employeeGender}"></td>
<td th:text="${employee.employeeSalary}"></td>
<td th:text="${employee.employeeRole+' Developer'}"></td>
</tr>
     </tbody>
          </table>
          </div>
          </div>
</div>
</main>

     <!-- model for create-->
<div class="modal fade" id="exampleModalToggle1" aria-hidden="true"
aria-labelledby="exampleModalToggleLabel" tabindex="-1">
<div class="modal-dialog modal-dialog-centered">
     <div class="modal-content">
          <div class="modal-body">
<div th:if="${success}" class="alert alert-success" role="alert">
<p th:text="${success}"></p>
</div>
<form class="p-2" th:action="@{/create}" th:object="${employee}"
method="post">
<center>
<h4 style="font-family: 'Times New Roman', Times, serif;">Add
Employee</h4>
</center>
     <label class="p-1" for="employeeEmail">Email</label> <input
type="text" th:field="*{employeeEmail}" class="form-control"
     placeholder="email address" required>
</div>
<div class="row p-2">
     <label class="p-1" for="employeePhone">Phone</label> <input
type="tel" th:field="*{employeePhone}" class="form-control"
     placeholder="phone number" required>
```

```
</div>
<div class="row p-2">
                            <label class="p-1">Gender</label> <select
th:field="*{employeeGender}" class="form-select" required>
          <option value="" selected>select option</option>
               <option value="Male">Male</option>
               <option value="Female">Female</option>
</select>
</div>
          <div class="row p-2">
     <label class="p-1" for="employeeSalary">Salary</label> <input
     type="number" th:field="*{employeeSalary}" class="form-control"
placeholder="salary" required>
     </div>
     <div class="row p-2">
<label class="p-1" for="employeeRole">Employee Role</label> <select
th:field="*{employeeRole}" class="form-select" required>
          <option value="" selected>select option</option>
     <option value="Java">Java Developer</option>
     <option value="Python">Python Developer</option>
     <option value="Web">Web Developer</option>
     <option value="Android">Android Developer</option>
     <option value="UI">UI Developer</option>

</select>
</div>
<button type="submit" class="btn btn-success mt-3 mb-2">Add
          Employee</button>
</form></div>
</div>
</div>

     <!-- model for update-->
<div class="modal fade" id="exampleModalToggle2" aria-hidden="true"
          aria-labelledby="exampleModalToggleLabel" tabindex="-1">
     <div class="modal-dialog modal-dialog-centered">
     <div class="modal-content">
     <div class="modal-body">
     <form class="p-2" th:action="@{/update}" th:object="${employee}"
method="post">  <!-- Check if errorMessage is present in the model and
display it -->
<div th:if="${errorMessage}" class="alert alert-danger role="alert">
<p th:text="${errorMessage}"></p>
</div>
<center>
<div class="row p-1">
          <label class="p-1" for="id">Employee ID</label> <input
```

```
type="text" th:field="*{id}" class="form-control"
          placeholder="employee id" required>
                                    </div>
     <div class="row p-1">
<label class="p-1" for="employeeName">Employee Name</label> <input
type="text" th:field="*{employeeName}" class="form-control"
                                placeholder="employee name" required>
          </div>
          <div class="row p-1">
     <label class="p-1" for="employeeEmail">Email</label>

<input type="text" th:field="*{employeeEmail}" class="form-
control"   placeholder="email address" required>
               </div>
<div class="row p-1">
     <label class="p-1" for="employeePhone">Phone</label> <input
type="tel" th:field="*{employeePhone}" class="form-control"
     placeholder="phone number" required>
          </div>

          <div class="row p-1">
<label class="p-1">Gender</label> <select
th:field="*{employeeGender}" class="form-select" required>
     <option value="" selected>select option</option>
               <option value="Male">Male</option>
               <option value="Female">Female</option>
          </select>
     </div>
<div class="row p-1">
     <label class="p-1" for="employeeSalary">Salary</label> <input
type="number" th:field="*{employeeSalary}" class="form-control"
placeholder="salary" required>
                                    </div>
                         <div class="row p-1">
<label class="p-1" for="employeeRole">Employee Role</label> <select
th:field="*{employeeRole}" class="form-select" required>
          <option value="" selected>select option</option>
          <option value="Java">Java Developer</option>
          <option value="Python">Python Developer</option>
          <option value="Web">Web Developer</option>
          <option value="Android">Android Developer</option>
          <option value="UI">UI Developer</option>
</select>
<button type="submit" class="btn btn-primary mt-3 mb-2">Update
Employee</button>
          </form>
</div>
```

```
<div class="row p-2">
          <label class="p-1" for="id">Employee ID</label> <input
type="text" th:field="*{id}" class="form-control"
     placeholder="employee id" required>
     </div>
<button type="submit" class="btn btn-danger mt-3 mb-2">Delete
                Employee</button>
     </form>
</div>
</div>

     <!-- Delete All employees-->
     <div class="modal fade" id="deleteAllModal4" aria-hidden="true"
          aria-labelledby="deleteAllModalLabe4" tabindex="-1">
          <div class="modal-dialog modal-dialog-centered">
               <div class="modal-content">

                    <div class="modal-body">
                         <form class="p-2" th:action="@{/remove/all}"
                              th:object="${confirmationForm}"
method="post">
<center><h4 style="font-family: 'Times New Roman', Times, serif;">Delete
     All Employees</h4></center>

<div class="row p-2">
<label class="p-3 text-warning" for="confirmation">Type
                              'Yes' For Confirmation</label> <input
type="text" th:field="*{confirmation}" class="form-control"
     placeholder="confirmation" required>
          </div>
<button type="submit" class="btn btn-dark mt-3 mb-2">Delete
                                   All Employees</button>
                         </form>
                    </div>
               </div>
          </div>
     </div>

     <!-- bootstrap js -->
     <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bund
le.min.js"></script>
</body>

</html>
```

New_employee.html

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org">
<head>
<meta charset="ISO-8859-1">
<title>Employee Management System</title>
<link rel="stylesheet"
	href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css"
	integrity="sha384-
MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO"
	crossorigin="anonymous">
</head>
<body>
	<div class="container">
		<h1>Employee Management System</h1>
		<hr>
		<h2>Save Employee</h2>

		<form action="#" th:action="@{/saveEmployee}" th:object="${employee}"
			method="POST">
			<input type="text" th:field="*{firstName}"
				placeholder="Employee First Name" class="form-control mb-4 col-
4">
				<input type="text" th:field="*{lastName}"

placeholder="Employee Last Name" class="form-control mb-4 col-4">

				<input type="text" th:field="*{email}"
				placeholder="Employee Email" class="form-control mb-4 col-4">

				<button type="submit" class="btn btn-info col-2"> Save
Employee</button>
</form>

<hr>		<a th:href = "@{/}"> Back to Employee List</a>
	</div>
</body>
</html>
```

addEmployee.html

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org">
<head>
<meta charset="ISO-8859-1">
<title>Employee Management System</title>
```

```
<link rel="stylesheet"
	href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css">
</head>
<body>
	<div class="container">
		<h1>Employee Management System</h1>
		<hr>
		<h2>Update Employee</h2>

		<form action="#" th:action="@{/saveEmployee}" th:object="${employee}"
			method="POST">
				<!-- Add hidden form field to handle update -->
			<input type="hidden" th:field="*{id}" />

			<input type="text" th:field="*{firstName}" class="form-control mb-4 col-
4">

				<input type="text" th:field="*{lastName}" class="form-control mb-
4 col-4">

<input type="text" th:field="*{email}" class="form-control mb-4 col-4">


<button type="submit" class="btn btn-info col-2"> Update Employee</button>
</form>
	<hr>
		<a th:href = "@{/}"> Back to Employee List</a>
	</div>
</body>
</html>
```

registration.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Registration</title>
<link rel="stylesheet"
	href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"
	integrity="sha384-
BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"
	crossorigin="anonymous">
</head>
<body>
	<!-- create navigation bar ( header) -->
	<nav class="navbar navbar-inverse navbar-fixed-top">
	<div class="container">
```

```
<div class="navbar-header">
<button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-
target="#navbar" aria-expanded="false" aria-controls="navbar">
<span class="sr-only">Toggle navigation</span> <span class="icon-bar"></span> <span
class="icon-bar"></span> <span  class="icon-bar"></span></button>
<a class="navbar-brand" href="#" th:href="@{/}">Employee Management System</a>

                    </div> </div>
        </nav><br><br>
        <!-- Create HTML registration form -->
        <div class="container">
                <div class="row">
                        <div class="col-md-6 col-md-offset-3">

                                <!-- success message -->
                                <div th:if="${param.success}">
                                        <div class="alert alert-info">You've successfully registered
                                                to our awesome app!</div>
                                </div>

                                <h1>Registration</h1>

                                <form th:action="@{/registration}" method="post"
th:object="${user}">
                                        <div class="form-group">
                                                <label class="control-label" for="firstName"> First
Name </label>
                                                <input id="firstName" class="form-control"
th:field="*{firstName}"         required autofocus="autofocus" />
                                        </div>

                                        <div class="form-group">
                                                <label class="control-label" for="lastName"> Last
Name </label> <input
                                                        id="lastName" class="form-control"
th:field="*{lastName}"
                                                        required autofocus="autofocus" />
                                        </div>

<div class="form-group">
<label class="control-label" for="email"> Email </label>
<input id="email" class="form-control" th:field="*{email}" required
        autofocus="autofocus" />
        </div>
        <div class="form-group">
        <label class="control-label" for="password"> Password </label>
<input id="password" class="form-control" type="password"
        th:field="*{password}" required autofocus="autofocus" />
```

login.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Registration and Login App</title>

<link rel="stylesheet"
	href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"
	integrity="sha384-
BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"
	crossorigin="anonymous">

</head><body>
	<!-- create navigation bar ( header) -->
	<nav class="navbar navbar-inverse navbar-fixed-top">
		<div class="container">
			<div class="navbar-header">
				<button type="button" class="navbar-toggle collapsed"
					data-toggle="collapse" data-target="#navbar" aria-
expanded="false"	aria-controls="navbar">
<span class="sr-only">Toggle navigation</span> <span

					class="icon-bar"></span> <span class="icon-bar"></span>
<span
						class="icon-bar"></span>
				</button>
				<a class="navbar-brand" href="#" th:href="@{/}">Employee
Management System</a>
			</div>		</div>
</nav>
	<br>	<br>
	<div class = "container">
		<div class = "row">
			<div class = "col-md-6 col-md-offset-3">

				<h1> Sign-in </h1>
				<form th:action="@{/login}" method="post">

					<!-- error message -->
					<div th:if="${param.error}">
						<div class="alert alert-danger">Invalid username or
							password.</div>
					</div>
```

```
                        <!-- logout message -->
                                <div th:if="${param.logout}">
                                        <div class="alert alert-info">You have been logged
out.</div>
                                </div>

                                <div class = "form-group">
                                        <label for ="username"> Username </label> :
                                <input type="text" class = "form-control" id ="username"
name = "username"
placeholder="Enter Email ID" autofocus="autofocus">
                                </div>

                                <div class="form-group">
<label for="password">Password</label>: <input type="password"          id="password"
name="password" class="form-control"
placeholder="Enter Password" /></div>

<div class="form-group">
        <div class="row">
        <div class="col-sm-6 col-sm-offset-3">
        <input type="submit" name="login-submit" id="login-submit"
                class="form-control btn btn-primary" value="Log In" />
                                                        </div> </div> </div> </form>
                <div class="form-group"><span>New user? <a href="/"
th:href="@{/registration}">Register here</a></span>
                        </div> </div></div></div></body>
</html>
```

pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
     <modelVersion>4.0.0</modelVersion>

     <parent>
          <groupId>org.springframework.boot</groupId>
          <artifactId>spring-boot-starter-parent</artifactId>
          <version>2.6.2</version>
          <relativePath/> <!-- lookup parent from repository -->
     </parent>
     <groupId>com.example</groupId>
```

```
<artifactId>thymeleaf</artifactId>
	<version>0.0.1-SNAPSHOT</version>
	<name>thymeleaf</name>
	<description>Demo project for Spring Boot</description>
	<properties>
		<java.version>17</java.version>
	</properties>
	<dependencies>
		<dependency>

	<groupId>org.springframework.boot</groupId>
	<artifactId>spring-boot-starter-data-jpa</artifactId>
		</dependency>

		<dependency>
	<groupId>org.springframework.boot</groupId>
	<artifactId>spring-boot-starter-thymeleaf</artifactId>
		</dependency>
		<dependency>

	<groupId>org.springframework.boot</groupId>
		<artifactId>spring-boot-starter-web</artifactId>
		</dependency>

		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-devtools</artifactId>
			<scope>runtime</scope>
			<optional>true</optional>
		</dependency>

		<dependency>
			<groupId>mysql</groupId>
			<artifactId>mysql-connector-java</artifactId>
			<scope>runtime</scope>
		</dependency>

		<dependency>
		<groupId>org.springframework.boot</groupId>
		<artifactId>spring-boot-starter test</artifactId>
	<scope>test</scope>
		</dependency>
	</dependencies>
	<build>
		<plugins>
			<plugin>
```

```
<groupId>org.springframework.boot</groupId>
          <artifactId>spring-boot-maven-plugin</artifactId>

               </plugin>
          </plugins>
     </build>
</project>
```

Java Code:

EmployeeManagementSystemProjectApplication.java

```
package in.Keshav;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class EmployeeManagementSystemProjectApplication {

        public static void main(String[] args) {

        SpringApplication.run(EmployeeManagementSystemProjectApplication.class, args);
        }

}
```

ServletInitializer.java

```
package in.Keshav;

import org.springframework.boot.builder.SpringApplicationBuilder;
import org.springframework.boot.web.servlet.support.SpringBootServletInitializer;
public class ServletInitializer extends SpringBootServletInitializer {

        @Override
        protected SpringApplicationBuilder configure(SpringApplicationBuilder application) {
                return

 application.sources(EmployeeManagementSystemProjectApplication.class);
        }

}
```

EmployeeController.java

```
package in.Keshav.controller;

import java.util.List;
```

```
import java.util.Optional;
import java.util.Random;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;
import in.Keshav.entity.ConfirmationForm;
import in.Keshav.entity.Employee;
import in.Keshav.repository.EmployeeRepo;

@Controller
public class EmployeeController {
	//https://www.javaguides.net/2021/07/spring-boot-tutorial-build-employee-management-project.html
		@Autowired
		private EmployeeRepo employeeRepo;
		// display the html page
		@GetMapping("/")

		public String getIndex(Model model) {
			List<Employee> employeeList = employeeRepo.findAll();
			model.addAttribute("employees", employeeList);
			model.addAttribute("employee", new Employee());
			model.addAttribute("confirmationForm", new ConfirmationForm());
			return "index";
		}

		// Insert employee data

		@PostMapping("/create")

		public String newEmployee(Employee employee, Model model) {
			model.addAttribute("employee", new Employee());

			// creating dynamic Employee ID
			String empId = "EMP";
			Random random = new Random();
			long randomNumber = 1000 + random.nextInt(9000);
			empId = empId + randomNumber;
			employee.setId(empId);

			// save the employee
			employeeRepo.save(employee);

			return "redirect:/";
		}
```

```
// update the existing employee
		@PostMapping("/update")
		public String updateEmployee(@ModelAttribute Employee employee, Model
model) {
			model.addAttribute("employee", new Employee());
			Optional<Employee> existingEmployee =
employeeRepo.findById(employee.getId());
			// checking employee exist or not
			if (existingEmployee.isPresent()) {

				employeeRepo.save(employee);
			} else {
				model.addAttribute("errorMessage", "Employee with ID " +
employee.getId() + " not found.");
			}
			return "redirect:/";
		}

		// delete an employee by id
		@PostMapping("/remove")

public String removeEmployee(Employee employee, Model model) {
			model.addAttribute("employee", new Employee());
			Optional<Employee> existingEmployee =
employeeRepo.findById(employee.getId());
			if (existingEmployee.isPresent()) {
				employeeRepo.deleteById(employee.getId());
			}
			return "redirect:/";
		}

		// delete all employees data by confromation
		@PostMapping("/remove/all")
		public String removeAll(@ModelAttribute ConfirmationForm confirmationForm,
Model model) {
			String confirmation = confirmationForm.getConfirmation();
			if ("Yes".equalsIgnoreCase(confirmation)) {
				employeeRepo.deleteAll();
			} else {
				return "redirect:/";
			}
			return "redirect:/";
		}

	}
```

ConfirmationForm.java

```
package in.Keshav.entity;
public class ConfirmationForm {

	private String confirmation;

	public ConfirmationForm() {
		super();
	}

	public ConfirmationForm(String confirmation) {
		super();
		this.confirmation = confirmation;
	}

	public String getConfirmation() {
		return confirmation;
	}

	public void setConfirmation(String confirmation) {
		this.confirmation = confirmation;
	}
	@Override
public String toString() {return "ConfirmationForm [confirmation=" +
confirmation +"]";


	}

}
```

Employee.java

```
package in.Keshav.entity;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table (name="employee_system")
public class Employee {
	@Id
	private String id;
   private String employeeName;
   private String employeeEmail;
   private Long employeePhone;
```

```
 private String employeeGender;
   private String employeeSalary;
   private String employeeRole;


        public Employee() {
                super();

        }
        public Employee(String id, String employeeName, String employeeEmail, Long
employeePhone, String employeeGender,
                        String employeeSalary, String employeeRole) {
                super();
                this.id = id;
                this.employeeName = employeeName;
                this.employeeEmail = employeeEmail;
                this.employeePhone = employeePhone;
                this.employeeGender = employeeGender;
                this.employeeSalary = employeeSalary;
                this.employeeRole = employeeRole;
        }
        public String getId() {
                return id;
        }
        public void setId(String id) {
                this.id = id;
        }
        public String getEmployeeName() {
                return employeeName;
        }
        public void setEmployeeName(String employeeName) {
                this.employeeName = employeeName;
        }
        public String getEmployeeEmail() {
                return employeeEmail;
        }
        public void setEmployeeEmail(String employeeEmail) {
                this.employeeEmail = employeeEmail;
        }
        public Long getEmployeePhone() {
                return employeePhone;
        }
        public void setEmployeePhone(Long employeePhone) {
                this.employeePhone = employeePhone;
        }
        public String getEmployeeGender() {
                return employeeGender;
```

```
}
	public void setEmployeeGender(String employeeGender) {
		this.employeeGender = employeeGender;
	}
	public String getEmployeeSalary() {
		return employeeSalary;
	}
	public void setEmployeeSalary(String employeeSalary) {
		this.employeeSalary = employeeSalary;
	}

	public String getEmployeeRole() {
		return employeeRole;
	}

	public void setEmployeeRole(String employeeRole) {

		this.employeeRole = employeeRole;
	}
	@Override
	public String toString() {
		return "Employee [id=" + id + ", employeeName=" + employeeName + ",
employeeEmail=" + employeeEmail
				+ ", employeePhone=" + employeePhone + ", employeeGender=" +
employeeGender + ", employeeSalary="
				+ employeeSalary + ", employeeRole=" + employeeRole + "]";
	}

}
```

EmployeeRepo.java

```
package in.Keshav.repository;

import org.springframework.data.jpa.repository.JpaRepository;
import in.Keshav.entity.Employee;
public interface EmployeeRepo extends JpaRepository<Employee, String>{

}
```

UserServiceImpl.java

```
import java.util.Arrays;
import java.util.Collection;
import java.util.stream.Collectors;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.core.GrantedAuthority;
import org.springframework.security.core.authority.SimpleGrantedAuthority;
```

```
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.security.core.userdetails.UsernameNotFoundException;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.stereotype.Service;
import net.javaguides.springboot.dto.UserRegistrationDto;
import net.javaguides.springboot.model.Role;
import net.javaguides.springboot.model.User;
import net.javaguides.springboot.repository.UserRepository;

@Service
public class UserServiceImpl implements UserService{

	private UserRepository userRepository;
	@Autowired
	private BCryptPasswordEncoder passwordEncoder;

	public UserServiceImpl(UserRepository userRepository) {
		super();
		this.userRepository = userRepository;
	}

	@Override
	public User save(UserRegistrationDto registrationDto) {
		User user = new User(registrationDto.getFirstName(),
	registrationDto.getLastName(), registrationDto.getEmail(),
	passwordEncoder.encode(registrationDto.getPassword()),
 Arrays.asList(new Role("ROLE_USER")));

		return userRepository.save(user);
	}

	@Override
	public UserDetails loadUserByUsername(String username) throws
UsernameNotFoundException {

		User user = userRepository.findByEmail(username);
		if(user == null) {
			throw new UsernameNotFoundException("Invalid username or password.");
		}
		return new org.springframework.security.core.userdetails.User(user.getEmail(),
user.getPassword(), mapRolesToAuthorities(user.getRoles()));
	}
	private Collection<? extends GrantedAuthority>
mapRolesToAuthorities(Collection<Role> roles){
		return roles.stream().map(role -> new
SimpleGrantedAuthority(role.getName())).collect(Collectors.toList());
	}
}
```

```
package in.Keshav.config;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.authentication.dao.DaoAuthenticationProvider;
import
org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBui
lder;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
import
org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapte
r;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.security.web.util.matcher.AntPathRequestMatcher;
```

SecurityConfiguration.java

```
import in.service.UserService;
@Configuration
@EnableWebSecurity
public class SecurityConfiguration extends WebSecurityConfigurerAdapter {

        @Autowired
        private UserService userService;

        @Bean
   public BCryptPasswordEncoder passwordEncoder() {
      return new BCryptPasswordEncoder();
   }

@Bean
   public DaoAuthenticationProvider authenticationProvider() {
      DaoAuthenticationProvider auth = new DaoAuthenticationProvider();
      auth.setUserDetailsService(userService);
      auth.setPasswordEncoder(passwordEncoder());
      return auth;
   }
        @Override
   protected void configure(AuthenticationManagerBuilder auth) throws Exception {
      auth.authenticationProvider(authenticationProvider());
   }
        @Override
        protected void configure(HttpSecurity http) throws Exception {
                http.authorizeRequests().antMatchers(
                                "/registration**",
```

```
 "/js/**",  "/css/**",  "/img/**").permitAll().anyRequest().authenticated()
                .and()
                .formLogin()
                .loginPage("/login")

                .permitAll()
                .and()
                .logout()
                .invalidateHttpSession(true)
                .clearAuthentication(true)
                .logoutRequestMatcher(new AntPathRequestMatcher("/logout"))
                .logoutSuccessUrl("/login?logout")
                .permitAll();
        }

}
```

MainController.java

```
package in.Keshav.controller;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.GetMapping;

@Controller

public class MainController {

      @GetMapping("/login")

      public String login() {

            return "login";

      }     /*

       * @GetMapping("/") public String home() { return "index"; }

       */}
```

UserRegistrationController.java

```
package in.Keshav.controller;
import org.springframework.stereotype.Controller;
```

```
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import net.javaguides.springboot.dto.UserRegistrationDto;
import net.javaguides.springboot.service.UserService;
@Controller
@RequestMapping("/registration")
public class UserRegistrationController {
private UserService userService;
      public UserRegistrationController(UserService userService) {
            super();
            this.userService = userService;
      }
      @ModelAttribute("user")
    public UserRegistrationDto userRegistrationDto() {
        return new UserRegistrationDto();
    }
      @GetMapping
      public String showRegistrationForm() {
            return "registration";
      }
      @PostMapping
```

```
public String registerUserAccount(@ModelAttribute("user")
UserRegistrationDto registrationDto) {

          userService.save(registrationDto);

return "redirect:/registration?success";

     }

}
```

EmployeeService.java

```
import java.util.List;

import org.springframework.data.domain.Page;

import net.javaguides.springboot.model.Employee;

public interface EmployeeService {

     List<Employee> getAllEmployees();

     void saveEmployee(Employee employee);

     Employee getEmployeeById(long id);void deleteEmployeeById(long
id);

     Page<Employee> findPaginated(int pageNo, int pageSize, String
sortField, String sortDirection);

}
```

Role.java

```
import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

@Entity
```

```
@Table(name = "role")
public class Role {
	@Id
	@GeneratedValue(strategy = GenerationType.IDENTITY)
	private Long id;
	private String name;
	public Role() {
	}
	public Role(String name) {
		super();
		this.name = name;
	}
	public Long getId() {
		return id;
	}
	public void setId(Long id) {
		this.id = id;
	}
	public String getName() {
		return name;
	public void setName(String name) {
		this.name = name;
	}
}
```

User.java

```
import java.util.Collection;
import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinTable;
import javax.persistence.ManyToMany;
import javax.persistence.Table;
import javax.persistence.UniqueConstraint;
import javax.persistence.JoinColumn;

@Entity

@Table(name =  "user", uniqueConstraints = @UniqueConstraint(columnNames
= "email"))

public class User {

      @Id

      @GeneratedValue(strategy =  GenerationType.IDENTITY)

      private Long id;

      @Column(name = "first_name")

      private String firstName;

      @Column(name = "last_name")

      private String lastName;

      private String email;

      private String password;

      @ManyToMany(fetch = FetchType.EAGER, cascade = CascadeType.ALL)

      @JoinTable(     name = "users_roles",

            joinColumns = @JoinColumn(
```

```
 name = "user_id", referencedColumnName = "id"),
inverseJoinColumns = @JoinColumn( name = "role_id",
      referencedColumnName = "id"))
private Collection<Role> roles;
      public User() {
      }
      public User(String firstName, String lastName, String email,
String password, Collection<Role> roles) {
            super();
            this.firstName = firstName;
            this.lastName = lastName;
            this.email = email;
            this.password = password;
            this.roles = roles;
      }
      public Long getId() {
            return id;
      }
public void setId(Long id) {
            this.id = id    }
      public String getFirstName() {
            return firstName;
      }
      public void setFirstName(String firstName) {
```

```
this.firstName = firstName;
      }


public String getLastName() {
            return lastName;
      }
      public void setLastName(String lastName) {
            this.lastName = lastName;
      }
      public String getEmail() {
            return email;
      }     public void setEmail(String email) {
            this.email = email;
      }
      public String getPassword() {
            return password;
      }
      public void setPassword(String password) {
            this.password = password;
      }
      public Collection<Role> getRoles() {
            return roles;
      }     public void setRoles(Collection<Role> roles) {
```

```
this.roles = roles;

      }

}
```