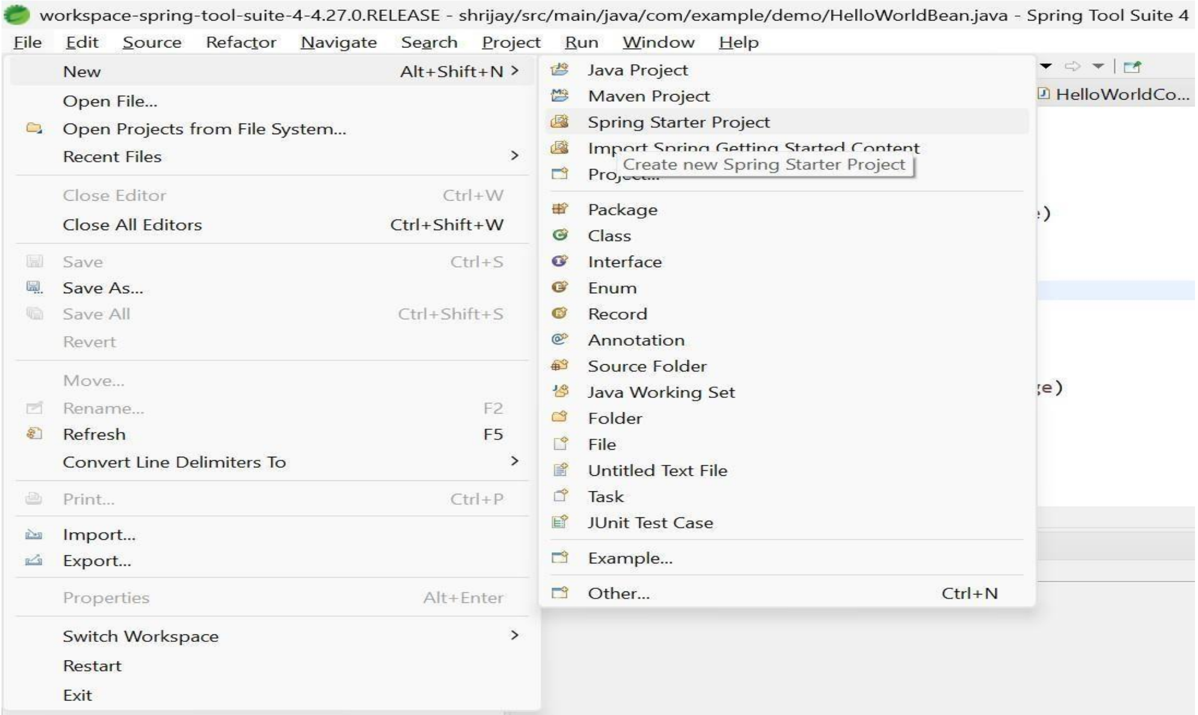
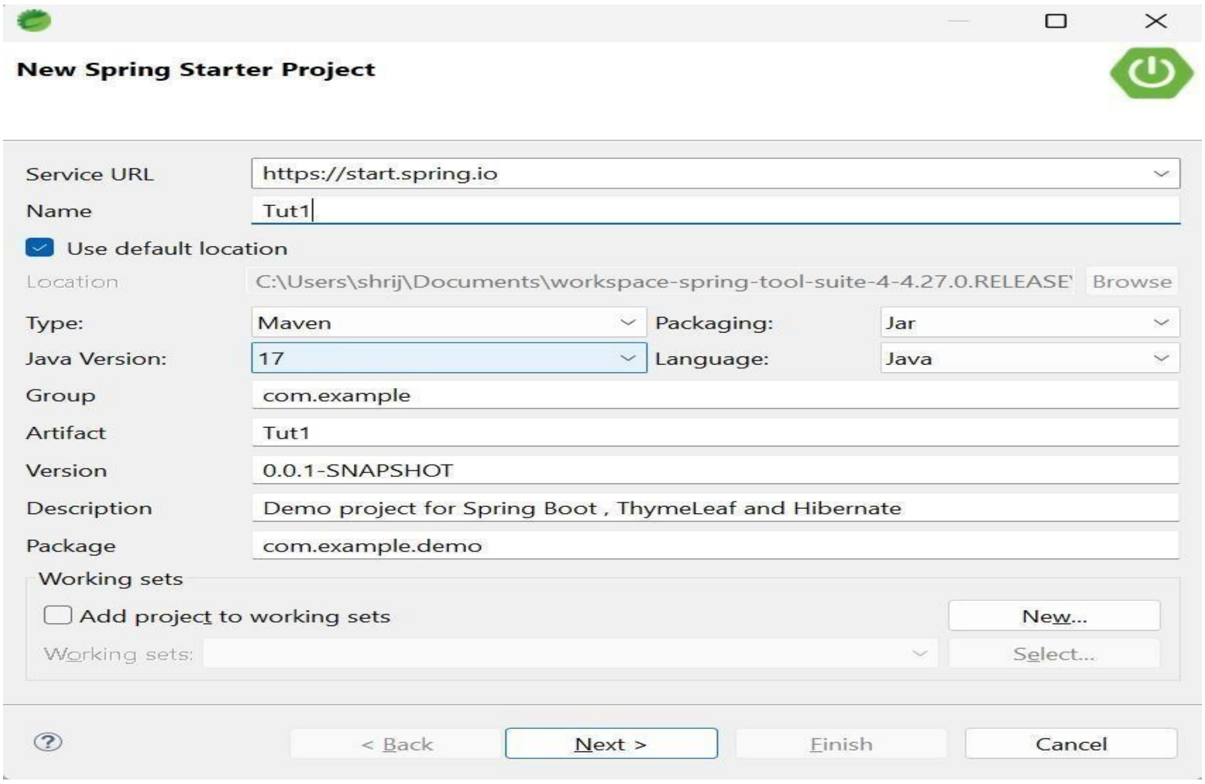
Spring Boot and RESTful Web Services

Steps to create a spring boot project :

New File ❼ Spring Starter Project

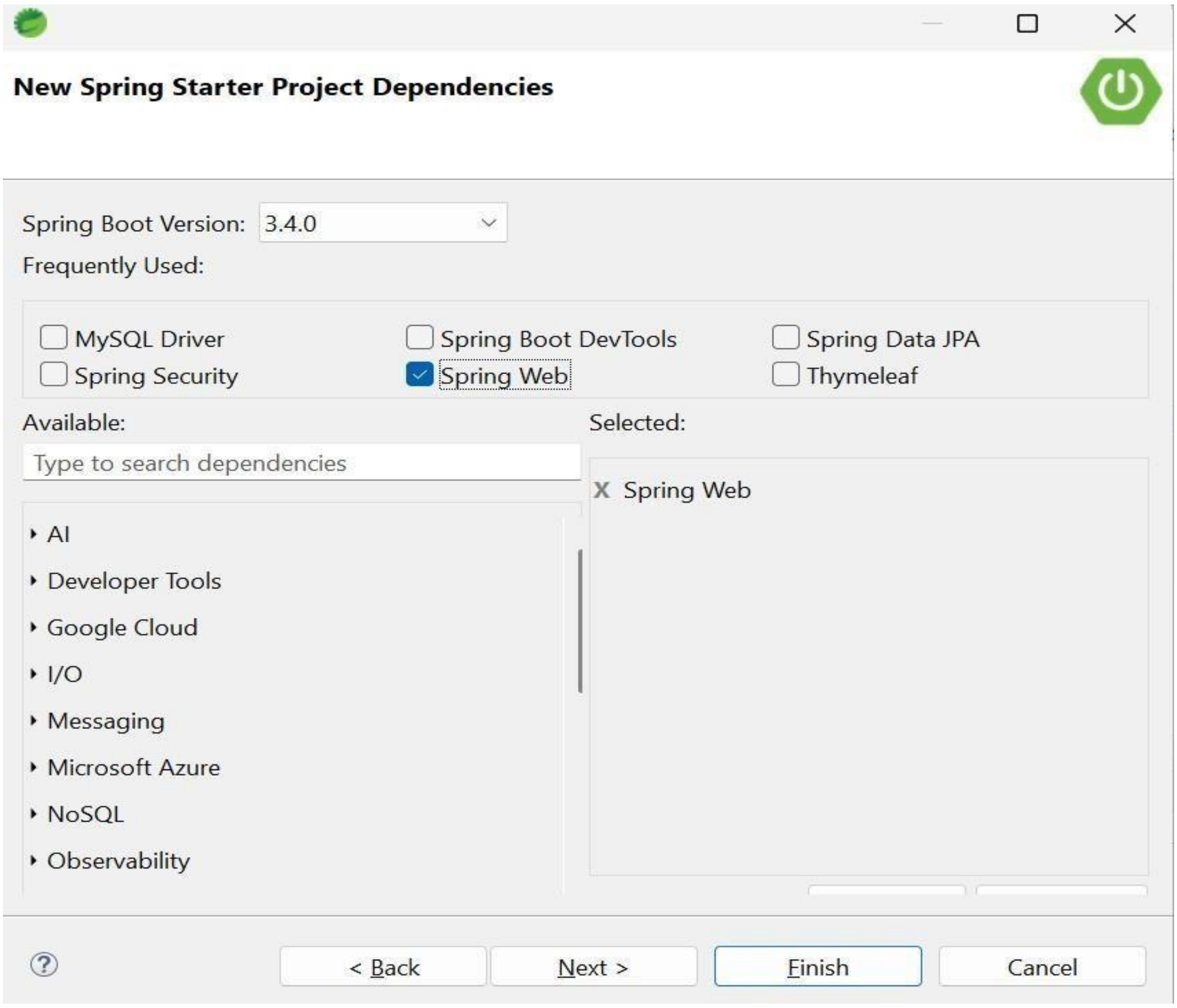


Poject name ❼next



Service URL: https://start.spring.io
Name: Tut1
☑ Use default location
Location: C:\Users\shrij\Documents\workspace-spring-tool-suite-4-4.27.0.RELEASE'
Type: Maven
Packaging: Jar
Java Version: 17
Language: Java
Group: com.example
Artifact: Tut1
Version: 0.0.1-SNAPSHOT
Description: Demo project for Spring Boot , ThymeLeaf and Hibernate
Package: com.example.demo
Working sets
☐ Add project to working sets
Working sets:

Select spring web ❼finish



New Spring Starter Project Dependencies

Spring Boot Version: 3.4.0

Frequently Used:

☐ MySQL Driver ☐ Spring Boot DevTools ☐ Spring Data JPA
☐ Spring Security ☑ Spring Web ☐ Thymeleaf

Available:

Selected:

X Spring Web

To run the application right click on main class run as then run as java application.

Q.1) Write a program to create a simple Spring Boot application that prints a message.

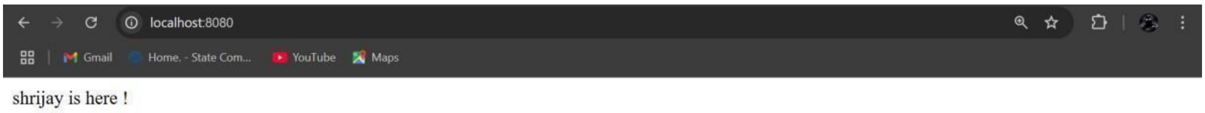
HelloController.java

```
package com.example.demo;


import org.springframework.web.bind.annotation.RequestMapping; import
org.springframework.web.bind.annotation.RestController;
@RestController
public class HelloController {
        @RequestMapping("/")
        public String hello()
        {
        return "shrijay is here !";
        }
}
```

Tut1Application.java package

```
com.example.demo;


import org.springframework.boot.SpringApplication; import
org.springframework.boot.autoconfigure.SpringBootApplication;


@SpringBootApplication
public class Tut1Application {


        public static void main(String[] args) {
                SpringApplication.run(Tut1Application.class, args);
        }

}
```

Output :



shrijay is here !

Q.2) Write a program to demonstrate RESTful Web Services with spring boot.

HelloWorldBean.java package com.example.demo;

```
public class HelloWorldBean {
        public String message;            public
HelloWorldBean(String message)
        {
        this.message=message;
        }
        public String getMessage()
        {
        return message;
        }
        public void setMessage(String message)
        {
        this.message = message;
        }
        @Override
        public String toString()
        {
        return String.format ("HelloWorldBean [message=%s]", message);
        }
}
```

HelloWorldController.java package com.example.demo;

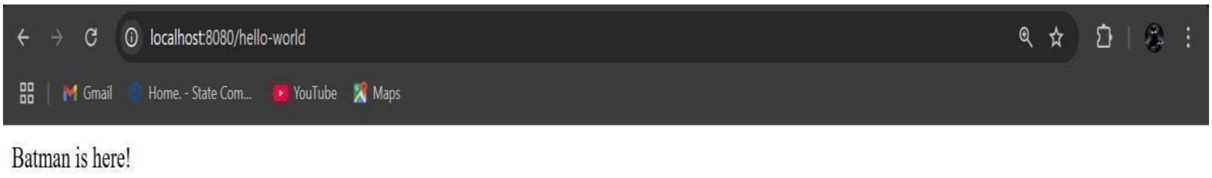
```
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;
```

```
@RestController public class
HelloWorldController {
        @GetMapping(path="/hello-world")
public String helloWorld()
        {
        return "Batman is here!";
        }
        @GetMapping(path="/hello-world-bean")
public HelloWorldBean helloWorldBean()
        {
        return new HelloWorldBean("How are you fella? xD");  }
}
```

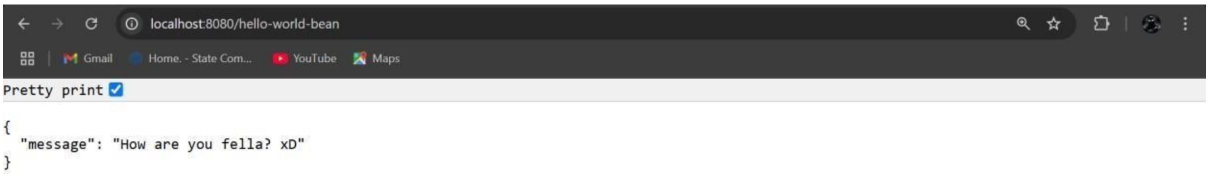
ShrijayApplication.java

```
package com.example.demo;


import org.springframework.boot.SpringApplication; import
org.springframework.boot.autoconfigure.SpringBootApplication;
@SpringBootApplication
public class ShrijayApplication {


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


}
```

Output :



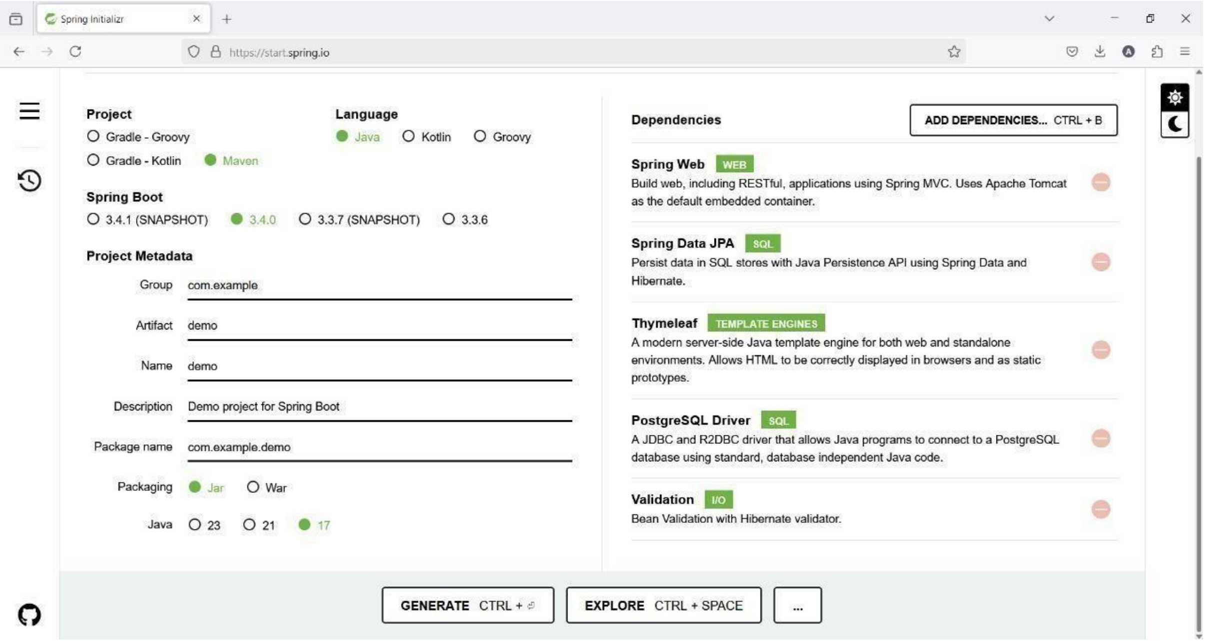
Batman is here!



```
{
  "message": "How are you fella? xD"
}
```

Q.3) Write a program to demonstrate database access with spring boot.

1) Go To spring initializr For Creating Springboot Project :



Project
○ Gradle - Groovy
○ Gradle - Kotlin
● Maven

Language
● Java ○ Kotlin ○ Groovy

Spring Boot
○ 3.4.1 (SNAPSHOT) ● 3.4.0 ○ 3.3.7 (SNAPSHOT) ○ 3.3.6

Project Metadata

Group: com.example
Artifact: demo
Name: demo
Description: Demo project for Spring Boot
Package name: com.example.demo
Packaging: ● Jar ○ War
Java: ○ 23 ○ 21 ● 17

Dependencies

Spring Web WEB
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

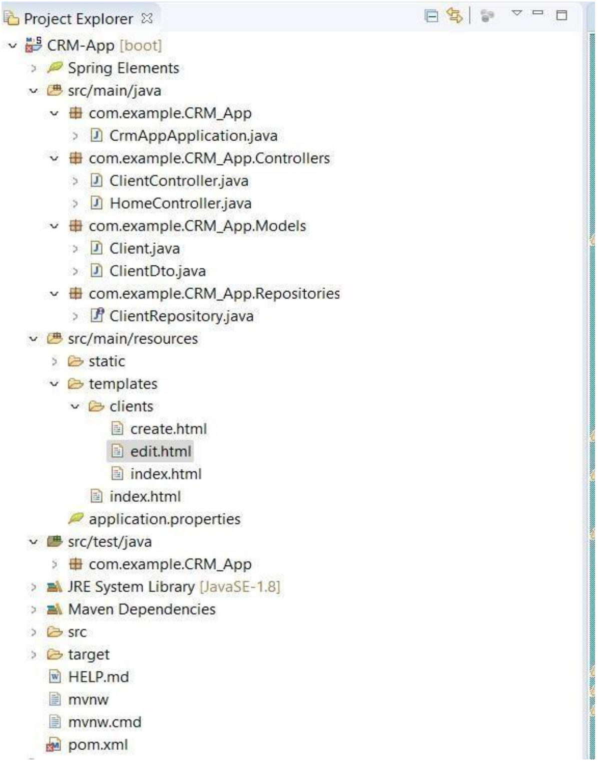
Spring Data JPA SQL
Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.

Thymeleaf TEMPLATE ENGINES
A modern server-side Java template engine for both web and standalone environments. Allows HTML to be correctly displayed in browsers and as static prototypes.

PostgreSQL Driver SQL
A JDBC and R2DBC driver that allows Java programs to connect to a PostgreSQL database using standard, database independent Java code.

Validation I/O
Bean Validation with Hibernate validator.

- **Project Explorer**

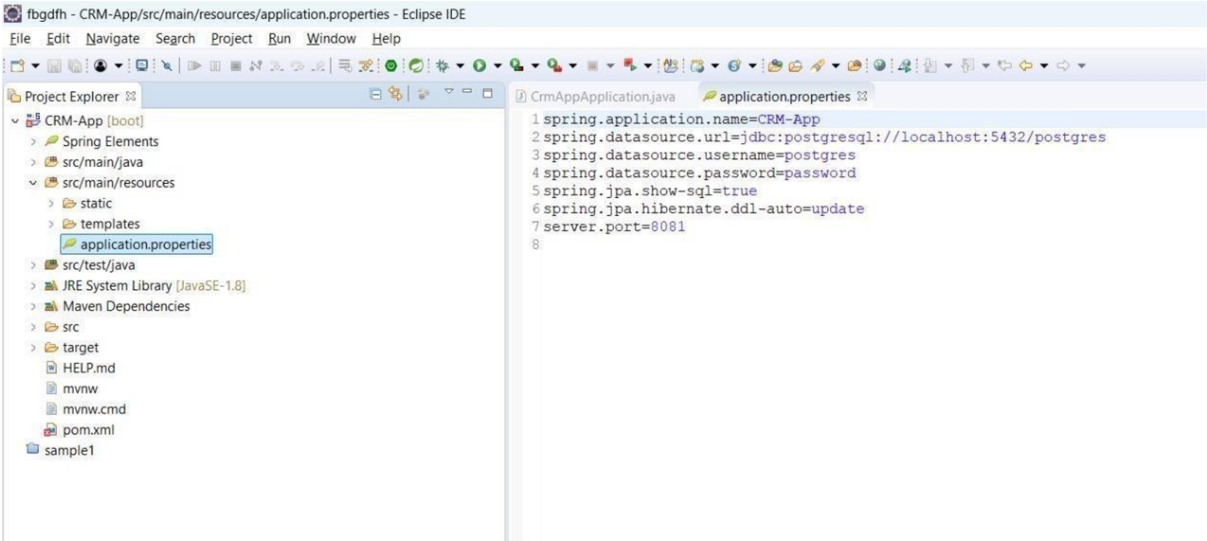


```
CRM-App [boot]
  Spring Elements
  src/main/java
    com.example.CRM_App
      CrmAppApplication.java
    com.example.CRM_App.Controllers
      ClientController.java
      HomeController.java
    com.example.CRM_App.Models
      Client.java
      ClientDto.java
    com.example.CRM_App.Repositories
      ClientRepository.java
  src/main/resources
    static
    templates
      clients
        create.html
        edit.html
        index.html
      index.html
    application.properties
  src/test/java
    com.example.CRM_App
  JRE System Library [JavaSE-1.8]
  Maven Dependencies
  src
  target
  HELP.md
  mvnw
  mvnw.cmd
  pom.xml
```

2) Create a database and table (Mysql / Posgress)

```
CREATE TABLE IF NOT EXISTS
clients (    id SERIAL PRIMARY
KEY,    address
VARCHAR(255),    create_at
TIMESTAMP(6),
    email VARCHAR(255) NOT NULL UNIQUE,
    first_name
VARCHAR(255),     last_name
VARCHAR(255),     phone
VARCHAR(255),
    status VARCHAR(255)
);
```

3) Application.properties



```
spring.application.name=CRM-App
spring.datasource.url=jdbc:postgresql://localhost:5432/postgres
spring.datasource.username=postgres
spring.datasource.password=password
spring.jpa.show-sql=true
spring.jpa.hibernate.ddl-auto=update
server.port=8081
```

4) Pom.xml

```
5) <?xml version="1.0" encoding="UTF-8"?>
6) <project
   xmlns="http://maven.apache.org/POM/4.0.0"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance"
7) xsi:schemaLocation="http://maven.apache.org/PO
   M/4.0.0 https://maven.apache.org/xsd/maven-
   4.0.0.xsd">
8) <modelVersion>4.0.0</modelVersion>
9) <parent>
10) <groupId>org.springframework.boot</groupId>
11) <artifactId>spring-boot-starter-
   parent</artifactId>
12) <version>3.4.0</version>
13) <relativePath/> <!-- lookup parent from
   repository
-->
14) </parent>
15) <groupId>com.example</groupId>
```

```xml
<artifactId>CRM-App</artifactId>
<version>0.0.1-SNAPSHOT</version>
<name>CRM-App</name>
<description>Demo project for Spring Boot</description>
<url/>
<licenses>
<license/>
</licenses>
<developers>
<developer/>
</developers>
<scm>
<connection/>
<developerConnection/>
<tag/>
<url/>
</scm>
<properties>
<java.version>17</java.version>
</properties>
<dependencies>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-datajpa</artifactId>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starterthymeleaf</artifactId>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-startervalidation</artifactId>
</dependency>
<dependency>
```

```
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-starterweb</artifactId>
        </dependency>

        <dependency>
                <groupId>org.postgresql</groupId>
                <artifactId>postgresql</artifactId>
                <scope>runtime</scope>
        </dependency>
        <dependency>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-startertest</artifactId>
                <scope>test</scope>
        </dependency>
        </dependencies>

        <build>
                <plugins>
                        <plugin>
 <groupId>org.springframework.boot</groupId>
                                <artifactId>spring-boot-mavenplugin</artifactId>
                        </plugin>
                </plugins>
        </build>

</project>
```

5) Entity

```
package com.example.CRM_App.Models;
import java.util.Date;

import jakarta.persistence.Column;
import jakarta.persistence.Entity;
import
jakarta.persistence.GeneratedValue;
```

```
    import
    jakarta.persistence.GenerationType;
    import jakarta.persistence.Id;
    import jakarta.persistence.Table;

    @Entity
    @Table(name="clients")
    public class Client {

    @Id

    @GeneratedValue(strategy=GenerationType.IDENTITY)
    private int id;          private String firstName;
            private String
    lastName;

    @Column(unique=true,nullable=false)
    private   String   email;   private   String
    phone;  private String address;
    private String status;
            @Column(name =
    "createAt") private Date
    createAt;      public int getId()
    {                  return id;
    }
public void setId(int 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 getPhone() {
             return
    phone;       }
         public void setPhone(String phone) {
    this.phone = phone;
    }
public String getAddress() {
             return
    address;     }
public void setAddress(String address) {
     this.address = address;
    }
public String getStatus() {
             return
    status;      }
         public void setStatus(String status) {
    this.status = status;
    }
         public Date getCreateAt() {
         return createAt;
    }
public void setCreateAt(Date createAt) {
     this.createAt = createAt;
         }

    }
```

B) Repository

```
package com.example.CRM_App.Repositories; import
org.springframework.data.jpa.repository.JpaRepository;

import com.example.CRM_App.Models.Client;

public interface ClientRepository extends
JpaRepository<Client,Integer> {

public Client findByEmail(String email);
}
```

C) **Controller-1** package

```
com.example.CRM_App.Controllers;

import java.util.Date;
import java.util.List;

import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Sort; 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
org.springframework.web.bind.annotation.RequestParam;

import com.example.CRM_App.Models.Client;
import
com.example.CRM_App.Models.ClientDto;
import com.example.CRM_App.Repositories.ClientRepository;
```

```
     import jakarta.validation.Valid;

     import org.springframework.ui.Model; import
     org.springframework.validation.BindingResult;
     import
     org.springframework.validation.FieldError;
     @Controller
     @RequestMapping("/clients")
     public class ClientController {

@Autowired
private ClientRepository clientRepo;

           @GetMapping({"", "/"})
           public String getClient(Model model) {
             List<Client> clients =
     clientRepo.findAll(Sort.by(Sort.Direction.DESC,
     "id"));          model.addAttribute("clients",
     clients);        return "clients/index";
     }

           @GetMapping({"/create"})      public
     String createClient(Model model) {
     ClientDto clientDto = new ClientDto();
     model.addAttribute("clientDto", clientDto);
       return "clients/create";
     }

           @PostMapping("/create")
     public String createClient(
  @Valid @ModelAttribute ClientDto clientDto,
  BindingResult result
    ) {

             if (clientRepo.findByEmail(clientDto.getEmail()) != null)
     {           result.addError(           new
     FieldError("clientDto", "email", clientDto.getEmail(),
     false, null, null, "Email address is already used")
```

```
        );
       }


                if (result.hasErrors()) {
           return "clients/create";
       }

                Client client = new Client();
      client.setFirstName(clientDto.getFirstName());
      client.setLastName(clientDto.getLastName());
      client.setEmail(clientDto.getEmail());
      client.setPhone(clientDto.getPhone());
      client.setAddress(clientDto.getAddress());
      client.setStatus(clientDto.getStatus());
      client.setCreateAt(new Date());

  clientRepo.save(client);

                return "redirect:/clients";
             }

@GetMapping("/edit")
public String editClient(Model model, @RequestParam int id) {
  Client client = clientRepo.findById(id).orElse(null);

                if (client == null) {
           return "redirect:/clients";
       }

                ClientDto clientDto = new ClientDto();
      clientDto.setFirstName(client.getFirstName());
      clientDto.setLastName(client.getLastName());
      clientDto.setEmail(client.getEmail());
      clientDto.setPhone(client.getPhone());
      clientDto.setAddress(client.getAddress());
  clientDto.setStatus(client.getStatus());
```

```
   model.addAttribute("client", client);
   model.addAttribute("clientDto", clientDto);

   return "clients/edit";
       }

                 @PostMapping("/edit")
        public String editClient(
        Model model,
   @RequestParam int id,
   @Valid @ModelAttribute ClientDto clientDto,
   BindingResult result
       ) {

   if (result.hasErrors()) {
      return "clients/edit";
         }

   Client client = clientRepo.findById(id).orElse(null);

                    if (client == null) {
              return "redirect:/clients";
         }

                    client.setFirstName(clientDto.getFirstName());
        client.setLastName(clientDto.getLastName());
        client.setEmail(clientDto.getEmail());
        client.setPhone(clientDto.getPhone());
                    client.setAddress(clientDto.getAddress());
        client.setStatus(clientDto.getStatus());

   clientRepo.save(client);

   return "redirect:/clients";
       }

@GetMapping("/delete")
```

```
            public String deleteClient(@RequestParam int id) {
        Client client = clientRepo.findById(id).orElse(null);

             if (client != null) {
      clientRepo.delete(client);
       }

return "redirect:/clients";
      }
      }
```

C) **Controller-2** package

```
com.example.CRM_App.Controllers;

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

@Controller
public class HomeController {

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

D) **validation** package

```
com.example.CRM_App.Models;
import
jakarta.validation.constraints.Email;
import
jakarta.validation.constraints.NotEmp
ty; public class ClientDto {
   @NotEmpty(message = "First Name is required")
   private String firstName;
```

```
    @NotEmpty(message = "Last Name is required")
    private String lastName;

    @Email(message = "Valid email is required")
@NotEmpty(message = "Email is required")
    private String email;

    private String phone;
    private String address;

    @NotEmpty(message = "Status is required")
    private String status;

    // Getters and Setters
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 getPhone() {
        return phone;
    }

    public void setPhone(String phone) {
        this.phone = phone;
    }

    public String getAddress() {
return address;
    }

    public void setAddress(String address) {
        this.address = address;
    }

    public String getStatus() {
return status;
    }

    public void setStatus(String status) {
        this.status = status;
    }
}
```

E) Template File: Ensure client/index.html exists in the src/main/resources/templates/ directory.

1. src/main/resources/templates/index.html

```
<!doctype html>
<html lang="en">
  <head>

    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <!-- Bootstrap CSS -->
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC
" crossorigin="anonymous">
    <link rel="stylesheet" href="style.css">
    <title>Clients Manager</title>
  </head>
  <body>
    <div class="container my-3">
    <h1 class="text-center mb-3" style="color:white;font-size:60px">Welcome to our
Website</h1>
    <center><a class="btn btn-primary" href="/clients">Clients</a></center>
</div>



    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.
js" integrity="sha384-
MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
crossorigin="anonymous"></script>
  </body>
</html>
```

2.src/main/resources/templates/clients/index.html

```
<!doctype html>
<html lang="en" xmlns:th="http://www.thymeleaf.org">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <!-- Bootstrap CSS -->
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
```

```
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC
" crossorigin="anonymous">
  <title>Clients Manager</title>
 </head>
 <body>
  <div class="container my-4">
   <h1 class="text-center mb-3">List of Clients</h1>
   <a class="btn btn-primary" href="/clients/create">New Clients</a>

   <table class="table">
    <thead>
     <tr>
      <th>ID</th>
      <th>Name</th>
      <th>Email</th>
      <th>Phone</th>
      <th>Address</th>
      <th>Status</th>
      <th>Created At</th>
      <th>Action</th>
     </tr>
    </thead>
    <tbody class="table-group-divider">
    <tr th:each="Client : ${clients}">
     <td th:text="${Client.id}"></td>
     <td th:text="${Client.firstName + ' ' + Client.lastName}"></td>
     <td th:text="${Client.email}"></td>
     <td th:text="${Client.phone}"></td>
     <td th:text="${Client.address}"></td>
     <td th:text="${Client.status}"></td>
     <td th:text="${Client.createAt != null ? #dates.format(Client.createAt,
'yyyyMM-dd HH:mm:ss') : 'N/A'}"></td>
     <td style="white-space:nowrap">
<a class="btn btn-primary btn-sm"
th:href="@{/clients/edit(id=${Client.id})}">E
dit</a>
      <a class="btn btn-danger btn-sm" th:href="@{/clients/delete(id=${Client.id})}"
onclick="return confirm('Are you sure?')">Delete</a>
     </td>
    </tr>
   </tbody>
   </table>
  </div>
```

```
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.
js" integrity="sha384-
MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
crossorigin="anonymous"></script>
  </body>
</html>
```

3.src/main/resources/templates/clients/create.html

```
<!doctype html>
<html lang="en">

<head>

  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">

  <!-- Bootstrap CSS -->
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.
min.css" rel="stylesheet"     integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC
" crossorigin="anonymous">

  <title>Clients Manager</title>
</head>

<body>
  <div class="container my-4">
    <div class="row">
      <div class="col-md-8 mx-auto rounded border p-4">
        <h2 class="text-center mb-5">New Client</h2>
        <form method="post" th:object="${clientDto}">
          <div class="row mb-3">
            <label class="col-sm-4 col-form-label">First Name</label>
            <div class="col-sm-8">
              <input class="form-control" th:field="${clientDto.firstName}"
/>              <p th:if="${#fields.hasErrors('firstName')}"
th:errorclass="text-danger"
th:errors="${clientDto.firstName}"></p>
            </div>
          </div>
    <div class="row mb-3">
      <label class="col-sm-4 col-form-label">Last Name</label>
      <div class="col-sm-8">
        <input class="form-control" th:field="${clientDto.lastName}">
        <p th:if="${#fields.hasErrors('lastName')}" th:errorclass="text-danger"
th:errors="${clientDto.lastName}">
        </p>
      </div>
    </div>

    <div class="row mb-3">
      <label class="col-sm-4 col-form-label">Email</label>
      <div class="col-sm-8">
        <input class="form-control" th:field="${clientDto.email}">
        <p th:if="${#fields.hasErrors('email')}" th:errorclass="text-danger"
th:errors="${clientDto.email}"></p>
      </div>
    </div>



    <div class="row mb-3">
      <label class="col-sm-4 col-form-label">Phone</label>
      <div class="col-sm-8">
        <input class="form-control" th:field="${clientDto.phone}">
        <p th:if="${#fields.hasErrors('phone')}" th:errorclass="text-danger"
th:errors="${clientDto.phone}"></p>
      </div>
    </div>
    <div class="row mb-3">
      <label class="col-sm-4 col-form-label">Address</label>
      <div class="col-sm-8">
        <input class="form-control" th:field="${clientDto.address}">
        <p th:if="${#fields.hasErrors('address')}" th:errorclass="text-danger"
th:errors="${clientDto.address}">
        </p>
      </div>
    </div>

    <div class="row mb-3">
      <label class="col-sm-4 col-form-label">Status</label>
      <div class="col-sm-8">
        <select class="form-select" th:field="${clientDto.status}">
        <option value="New">New</option>
        <option value="Permanent">Permanent</option>
        <option value="Lead">Lead</option>
        <option value="Occasional">Occasional</option>
        <option value="Inactive">Inactive</option>
       </select>
       <p th:if="${#fields.hasErrors('status')}" th:errorclass="text-danger"
th:errors="${clientDto.status}"></p>
      </div>
     </div>
     <div class="row">
      <div class="offset-sm-4 col-sm-4 d-grid">
       <button type="submit" class="btn btn-primary">Submit</button>
      </div>
      <div class="col-sm-4 d-grid">
       <a class="btn btn-outline-primary" href="/clients">Cancel</a>
      </div>
     </div>


    </form>

   </div>
  </div>
 </div>



 <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.
js"   integrity="sha384-
MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
crossorigin="anonymous"></script>
</body>

</html>
```

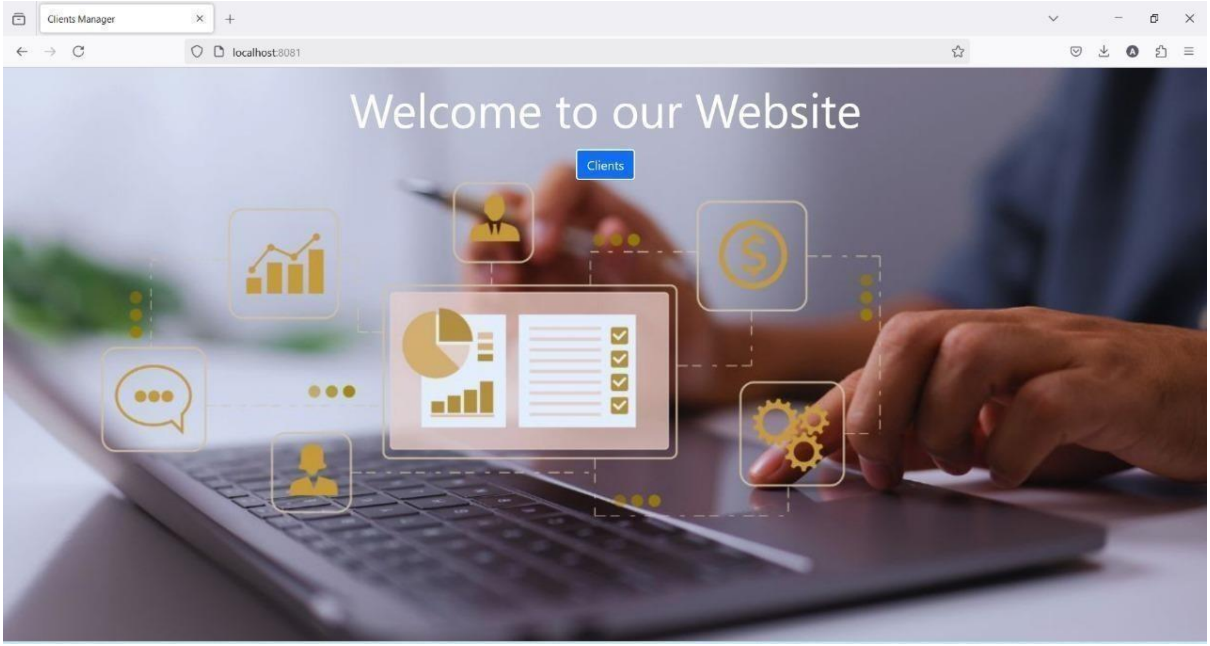
4.src/main/resources/templates/clients/edit.html

```
<!doctype html>
<html lang="en">

<head>
```

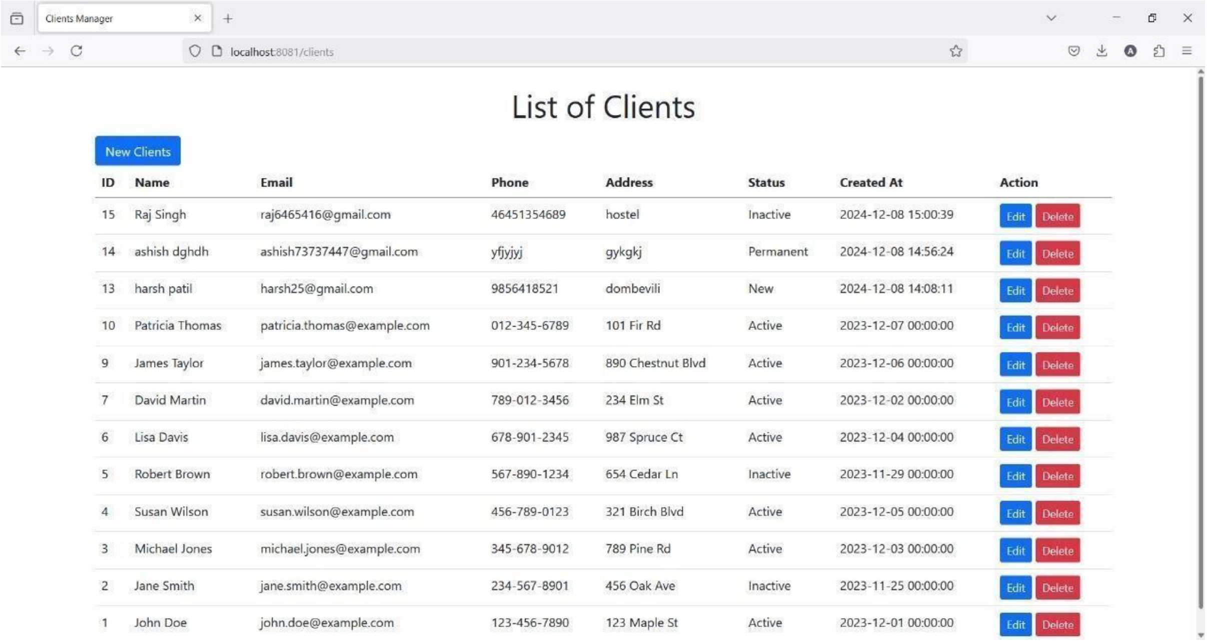
```
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">

  <!-- Bootstrap CSS -->
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.
min.css" rel="stylesheet"    integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC
" crossorigin="anonymous">

  <title>Clients Manager</title>
</head>

<body>
  <div class="container my-4">
    <div class="row">
      <div class="col-md-8 mx-auto rounded border p-4">
        <h2 class="text-center mb-5">Edit Client</h2>
        <form method="post" th:object="${clientDto}">

        <div class="row mb-3">
            <label class="col-sm-4 col-form-label">Client Id</label>
            <div class="col-sm-8">
              <input class="form-control-plaintext" readonly th:value="${client.id}" />

            </div>

          <div class="row mb-3">
            <label class="col-sm-4 col-form-label">First Name</label>
            <div class="col-sm-8">
              <input class="form-control" th:field="${clientDto.firstName}" required />
<p th:if="${#fields.hasErrors('firstName')}" th:errorclass="text-danger"
th:errors="${clientDto.firstName}"></p>
            </div>
          </div>
          <div class="row mb-3">
            <label class="col-sm-4 col-form-label">Last Name</label>
            <div class="col-sm-8">
              <input    class="form-control"    th:field="${clientDto.lastName}"
required>                          <p  th:if="${#fields.hasErrors('lastName')}"
th:errorclass="text-danger" th:errors="${clientDto.lastName}">
              </p>
            </div>
          </div>

          <div class="row mb-3">
            <label class="col-sm-4 col-form-label">Email</label>
            <div class="col-sm-8">
              <input class="form-control" th:field="${clientDto.email}"
required>             <p th:if="${#fields.hasErrors('email')}"
th:errorclass="text-danger" th:errors="${clientDto.email}"></p>
            </div>
          </div>



          <div class="row mb-3">
            <label class="col-sm-4 col-form-label">Phone</label>
            <div class="col-sm-8">
              <input class="form-control" th:field="${clientDto.phone}"
required>             <p th:if="${#fields.hasErrors('phone')}"
th:errorclass="text-danger" th:errors="${clientDto.phone}"></p>
            </div>
          </div>
          <div class="row mb-3">
            <label class="col-sm-4 col-form-
label">Address</label>           <div class="col-sm-8">
              <input    class="form-control"    th:field="${clientDto.address}"
required>                            <p   th:if="${#fields.hasErrors('address')}"
th:errorclass="text-danger" th:errors="${clientDto.address}">
              </p>
            </div>
          </div>

          <div class="row mb-3">
            <label class="col-sm-4 col-form-label">Status</label>
            <div class="col-sm-8">
              <select class="form-select" th:field="${clientDto.status}">
                <option value="New">New</option>
                <option value="Permanent">Permanent</option>
                <option value="Lead">Lead</option>
                <option value="Occasional">Occasional</option>
                <option value="Inactive">Inactive</option>
              </select>
```

```
       <p th:if="${#fields.hasErrors('status')}" th:errorclass="text-danger"
th:errors="${clientDto.status}"></p>
      </div>
     </div>

     <div class="row mb-3">
      <label class="col-sm-4 col-form-label">Created At</label>
      <div class="col-sm-8">
       <input          class="form-control-plaintext"          readonly
th:value="${client.CreateAt}" required/>
      </div>
     <div class="row">
      <div class="offset-sm-4 col-sm-4 d-grid">
       <button type="submit" class="btn btn-primary">Update</button>
      </div>
      <div class="col-sm-4 d-grid">
       <a class="btn btn-outline-primary" href="/clients">Cancel</a>
      </div>
     </div>


    </form>

   </div>
  </div>
 </div>

 <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.
js"   integrity="sha384-
MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
crossorigin="anonymous"></script>
</body>

</html>
```

✞ Output:-

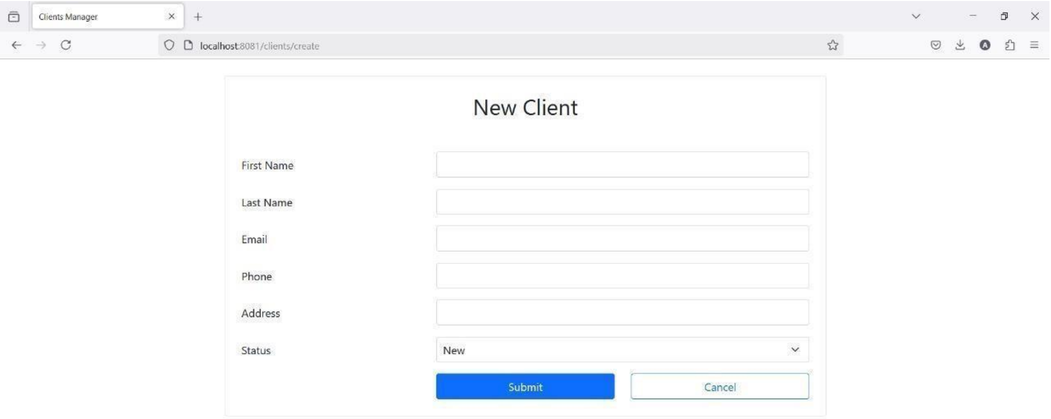


Welcome to our Website



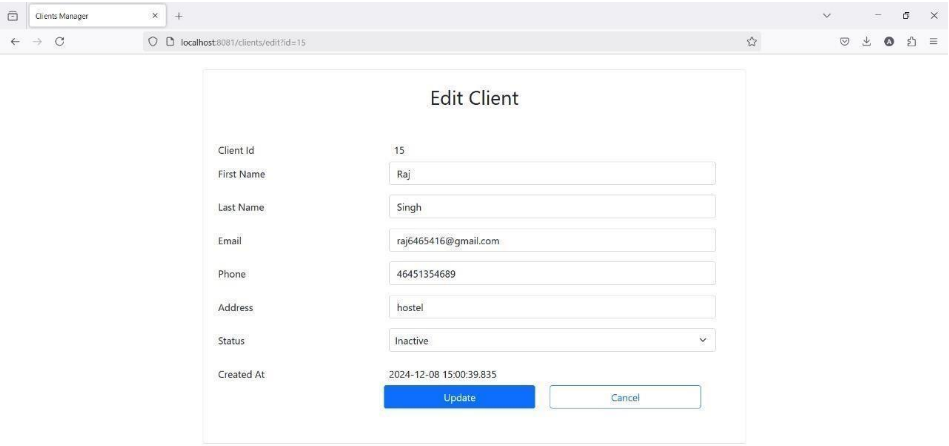
List of Clients

ID	Name	Email	Phone	Address	Status	Created At	Action
15	Raj Singh	raj6465416@gmail.com	46451354689	hostel	Inactive	2024-12-08 15:00:39	Edit Delete
14	ashish dghdh	ashish73737447@gmail.com	yfjyjyj	gykgkj	Permanent	2024-12-08 14:56:24	Edit Delete
13	harsh patil	harsh25@gmail.com	9856418521	dombevili	New	2024-12-08 14:08:11	Edit Delete
10	Patricia Thomas	patricia.thomas@example.com	012-345-6789	101 Fir Rd	Active	2023-12-07 00:00:00	Edit Delete
9	James Taylor	james.taylor@example.com	901-234-5678	890 Chestnut Blvd	Active	2023-12-06 00:00:00	Edit Delete
7	David Martin	david.martin@example.com	789-012-3456	234 Elm St	Active	2023-12-02 00:00:00	Edit Delete
6	Lisa Davis	lisa.davis@example.com	678-901-2345	987 Spruce Ct	Active	2023-12-04 00:00:00	Edit Delete
5	Robert Brown	robert.brown@example.com	567-890-1234	654 Cedar Ln	Inactive	2023-11-29 00:00:00	Edit Delete
4	Susan Wilson	susan.wilson@example.com	456-789-0123	321 Birch Blvd	Active	2023-12-05 00:00:00	Edit Delete
3	Michael Jones	michael.jones@example.com	345-678-9012	789 Pine Rd	Active	2023-12-03 00:00:00	Edit Delete
2	Jane Smith	jane.smith@example.com	234-567-8901	456 Oak Ave	Inactive	2023-11-25 00:00:00	Edit Delete
1	John Doe	john.doe@example.com	123-456-7890	123 Maple St	Active	2023-12-01 00:00:00	Edit Delete



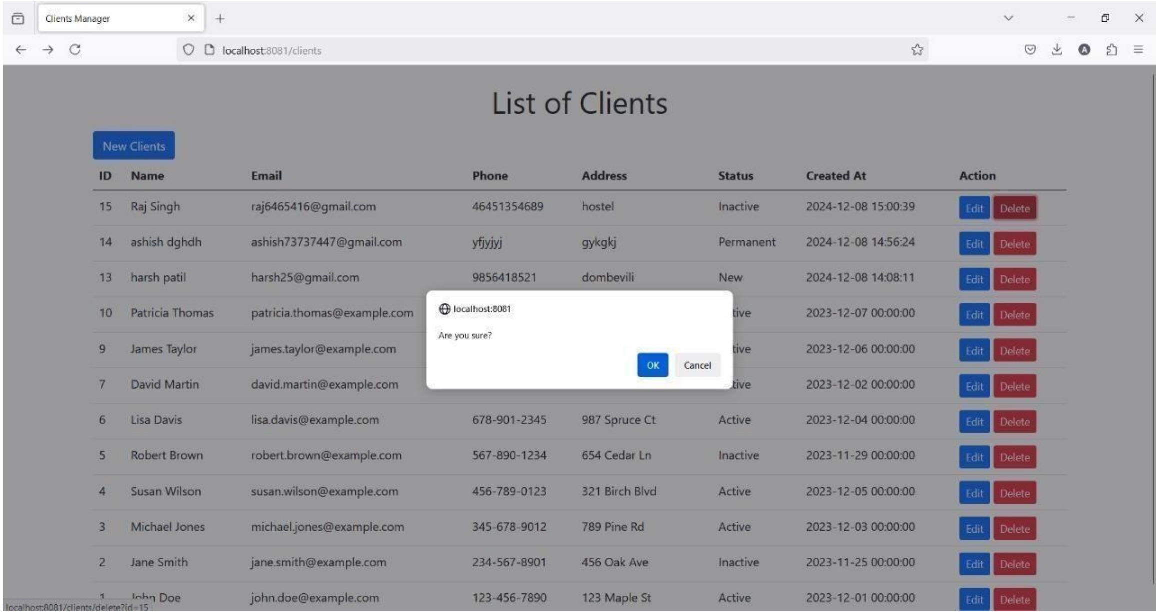
New Client

First Name	
Last Name	
Email	
Phone	
Address	
Status	New



Edit Client

Client Id	15
First Name	Raj
Last Name	Singh
Email	raj6465416@gmail.com
Phone	46451354689
Address	hostel
Status	Inactive
Created At	2024-12-08 15:00:39.835



List of Clients

ID	Name	Email	Phone	Address	Status	Created At	Action
15	Raj Singh	raj6465416@gmail.com	46451354689	hostel	Inactive	2024-12-08 15:00:39	Edit Delete
14	ashish dghdh	ashish73737447@gmail.com	yfjyjyj	gykgkj	Permanent	2024-12-08 14:56:24	Edit Delete
13	harsh patil	harsh25@gmail.com	9856418521	dombevili	New	2024-12-08 14:08:11	Edit Delete
10	Patricia Thomas	patricia.thomas@example.com			tive	2023-12-07 00:00:00	Edit Delete
9	James Taylor	james.taylor@example.com			tive	2023-12-06 00:00:00	Edit Delete
7	David Martin	david.martin@example.com			tive	2023-12-02 00:00:00	Edit Delete
6	Lisa Davis	lisa.davis@example.com	678-901-2345	987 Spruce Ct	Active	2023-12-04 00:00:00	Edit Delete
5	Robert Brown	robert.brown@example.com	567-890-1234	654 Cedar Ln	Inactive	2023-11-29 00:00:00	Edit Delete
4	Susan Wilson	susan.wilson@example.com	456-789-0123	321 Birch Blvd	Active	2023-12-05 00:00:00	Edit Delete
3	Michael Jones	michael.jones@example.com	345-678-9012	789 Pine Rd	Active	2023-12-03 00:00:00	Edit Delete
2	Jane Smith	jane.smith@example.com	234-567-8901	456 Oak Ave	Inactive	2023-11-25 00:00:00	Edit Delete
1	John Doe	john.doe@example.com	123-456-7890	123 Maple St	Active	2023-12-01 00:00:00	Edit Delete

localhost:8081

Are you sure?