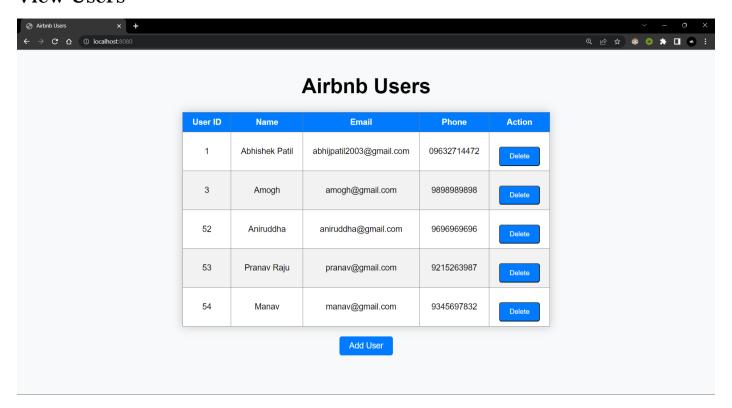
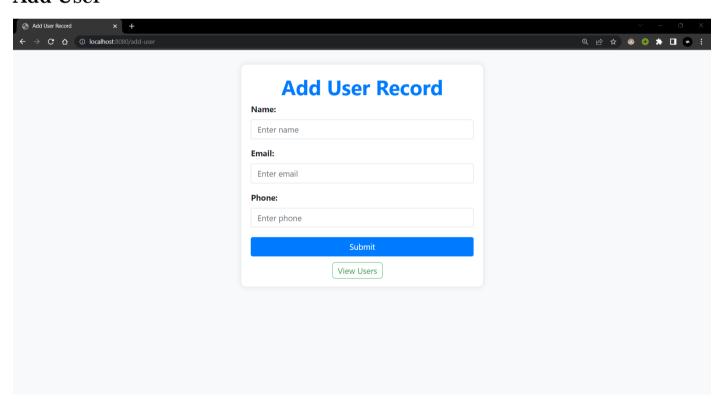
CRUD Web Application with Spring Boot, MySQL and Thymeleaf

- MVC (Model-View-Controller) is a popular architecture pattern used in software development to separate an application's concerns into three interconnected components:
 - o Model
 - o View
 - o Controller
- The Model represents the data and logic of the application
- The View represents the user interface
- The Controller handles user input and manages the interaction between the Model and View.
- This separation of concerns allows for better code organization, maintainability, and scalability.
- Spring Boot is a popular Java framework that provides a simplified way of building and deploying web applications.
- Features
 - o auto-configuration
 - o embedded web server (**Tomcat** server)
 - o robust set of libraries to build web applications faster
- Spring Boot follows the MVC architecture pattern
 - o Model is represented by domain objects or entities
 - View is represented by templates or user interfaces
 - o Controller is represented by the Spring Framework Controller
- Spring Boot provides several features to make MVC development easier
 - o annotations-based configuration
 - o data binding
 - o error handling.
- Tools Used
 - o Java language
 - o **Spring Boot** Java Web Framework (follows MVC architecture)
 - o **Thymeleaf** to render HTML pages
 - o MySQL Database
 - o IntelliJ Ide

View Users



Add User



MvcApplication.java

```
package dev.abhishek.mvc;

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

@SpringBootApplication
public class MvcApplication {

public static void main(String[] args) {

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

SpringApplication.run(MvcApplication.class, args);
}
```

User.java

```
4 import jakarta.persistence.GeneratedValue;
 5 import jakarta.persistence.GenerationType;
6 import jakarta.persistence.Id;
 7 import lombok.AllArgsConstructor;
8 import Lombok Data;
   import Lombok.NoArgsConstructor;
11 @Entity
12 @Data
13 @AllArgsConstructor
14 @NoArgsConstructor
15 public class User{
     @Id
@GeneratedValue(strategy= GenerationType.AUTO)
      private Integer userid;
      private String name;
      private String email;
       private String phone;
```

application.properties

```
spring.jpa.hibernate.ddl-auto=update
spring.datasource.url=jdbc:mysql://${MYSQL_HOST:localhost}:3306/mvc
spring.datasource.username=root
spring.datasource.password=abhi-patil
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
```

UserRepository.java

```
package dev.abhishek.mvc.repositories;

import org.springframework.data.repository.CrudRepository;
import dev.abhishek.mvc.models.User;
public interface UserRepository extends CrudRepository<User, Integer> {
}
```

MainController.java

```
package dev.abhishek.mvc.controllers;
4 import dev.abhishek.mvc.repositories.UserRepository;
  import org.springframework.beans.factory.annotation.Autowired;
6 import org.springframework.stereotype.Controller;
   import org.springframework.ui.Model;
   import org.springframework.web.bind.annotation.*;
  @Controller
  public class MainController {
     @Autowired
      private UserRepository userRepository;
     @GetMapping(path="/")
      public String viewUser(Model model){
           List<User> users = (List<User>) userRepository.findAll();
           model.addAttribute("users", users);
           return "view-user";
     @GetMapping(path="/add-user")
       public String returnAddUserPage(Model model) {
          User user = new User();
           model.addAttribute("user", user);
           return "add-user";
       @PostMapping(path="/add-user")
       public String addNewUser(@ModelAttribute("user") User user) {
           userRepository.save(user);
           return "redirect:/add-user";
     @PostMapping("/delete-user")
      public String deleteUser(@RequestParam("id") Integer id, Model model) {
           userRepository.deleteById(id);
           model.addAttribute("users", userRepository.findAll());
           return "redirect:/";
```

```
<!DOCTYPE html>
    <html xmLns:th="http://www.w3.org/1999/xhtml" Lang="en">
       <title>Add User Record</title>
       <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
               background-color: #f8f9fa;
               max-width: 500px;
               margin: 0 auto;
               margin-top: 30px;
              background-color: #ffffff;
               padding: 20px;
               border-radius: 10px;
               box-shadow: 0px 0px 10px rgba(0,0,0,0.1);
               text-align: center;
               font-weight: bold;
               color: #007bff;
               font-weight: bold;
           button[type="submit"] {
               margin-top: 20px;
                background-color: #007bff;
               border-color: #007bff;
           button[type="submit"]:hover {
               background-color: #0069d9;
               border-color: #0062cc;
               color: #28a745 !important;
               margin-right: 20px;
              border: 1px solid #28a745;
padding: 5px 10px;
               border-radius: 5px:
            color: #ffffff !important;
             background-color: #28a745;
             border-color: #28a745;
             text-decoration: none;
       <h1>Add User Record</h1>
       <form action="#" th:action="@{/add-user}" method="POST" th:object="${user}">
           <div class="form-group";</pre>
              <label>Name:</label>
               <input type="text" th:field = "*{name}" class="form-control" id="name" placeholder="Enter name">
           <div class="form-group">
              <label>Email:</label>
               <input type="email" th:field = "*{email}" class="form-control" id="email" placeholder="Enter email">
           <div class="form-group">
               <label>Phone:</label>
               <input type="text" th:field = "*{phone}" class="form-control" id="phone" placeholder="Enter phone">
           <button type="submit" class="btn btn-primary btn-block">Submit</button>
           <a href="/">View Users</a>
```

```
1 <!DOCTYPE html>
  <html lang="en" xmlns:th="http://www.w3.org/1999/xhtml">
    <meta charset="UTF-8">
    <title>Airbnb Users</title>
    <style>
               font-family: Arial, sans-serif;
               background-color: #f8f9fa;
               text-align: center;
               font-size: 2.5rem;
               margin-top: 50px;
               margin-bottom: 30px;
              margin: 0 auto;
               border-collapse: collapse;
               background-color: #fff;
               box-shadow: 0 0 20px rgba(0,0,0,0.2);
               padding: 10px 20px;
               text-align: center;
               background-color: #007bff;
               color: #fff;
           tbody tr:nth-child(2n) {
               background-color: #f2f2f2;
           .button {
               display: inline-block;
               background-color: #007bff;
               color: #fff;
               padding: 10px 20px;
               margin-top: 20px;
               border-radius: 5px;
               text-decoration: none;
               transition: background-color 0.3s ease;
```

```
.button:hover {
            background-color: #0056b3;
         .delete-button {
            display inline-block;
            background-color: #dc3545;
            color: #fff;
            padding: 5px 10px;
            margin-left: 10px;
            border-radius: 5px;
            text-decoration: none;
            transition: background-color 0.3s ease;
         .delete-button:hover {
            background-color: #c82333;
      </style>
   <h1>Airbnb Users</h1>
  User ID
      Name
    Email
    Phone
     Action

       <form method="post" th:action="@{/delete-user}">
         <input type="hidden" name="id" th:value="${user.userid}" />
         <button type="submit" class="button">Delete/button>
       </form>
      <div style="text-align: center;">
    <a href="/add-user" class="button">Add User</a>
50 </div>
  </body>
```