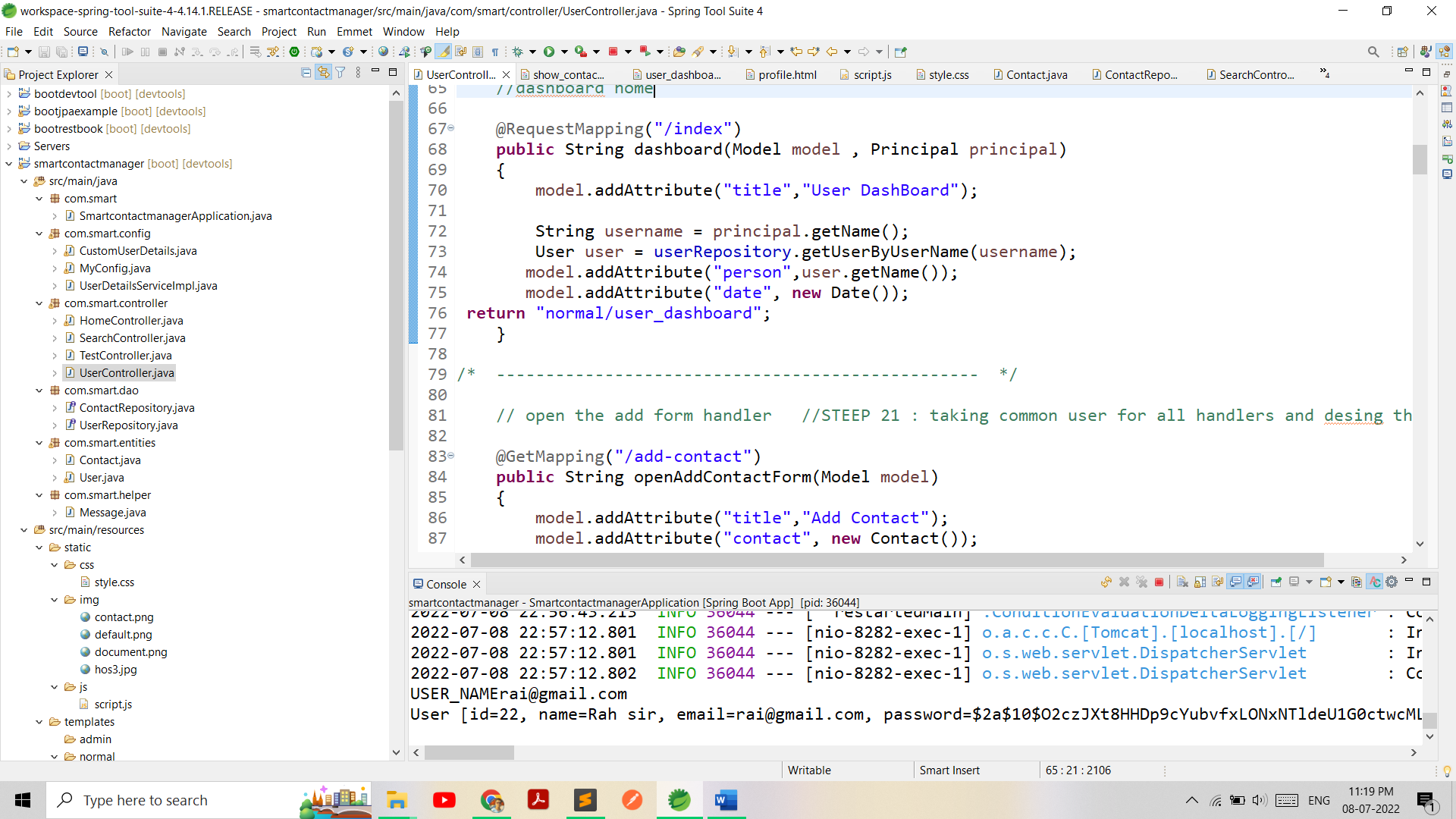
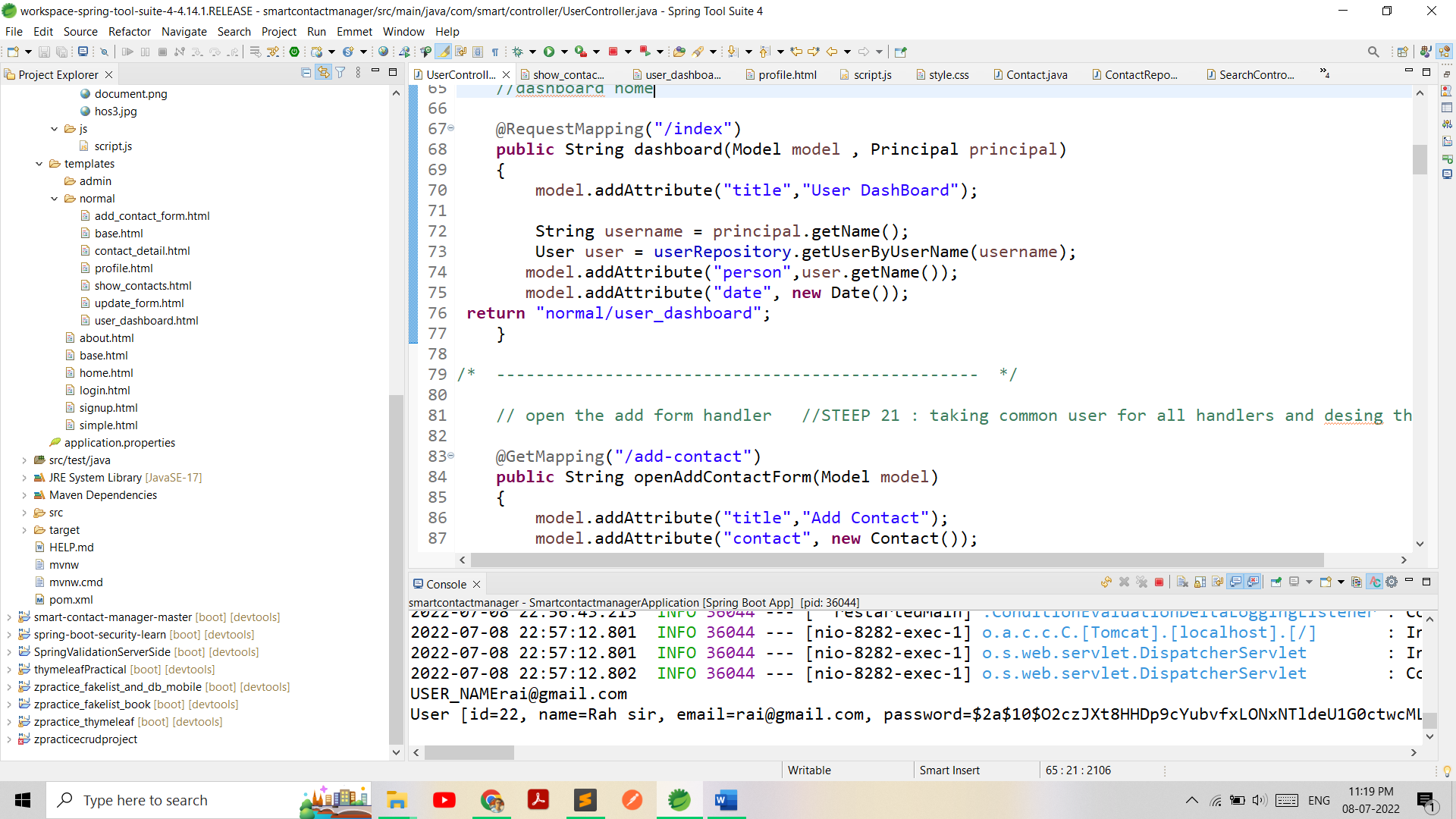
**: : PHONEBOOK CLOUD BY SHIVAM RAI : :**





* **Refer images for the folder structure in spring tool suite.**

***CODE IN JAVA (SPRING BOOT , HTML , CSS , JS )***

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**0. pom.xml file**

<?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>2.7.0</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<groupId>com.smart</groupId>

<artifactId>smartcontactmanager</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>smartcontactmanager</name>

<description>this spring boot project </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>

<!-- https://mvnrepository.com/artifact/javax.validation/validation-api -->

<dependency>

<groupId>javax.validation</groupId>

<artifactId>validation-api</artifactId>

</dependency>

<!-- https://mvnrepository.com/artifact/org.hibernate/hibernate-validator -->

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-validator</artifactId>

<version>6.0.17.Final</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.springframework.boot/spring-boot-starter-security -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-security</artifactId>

</dependency>

<!-- Just trying the sending mail for the sent OTP for password reset -->

<!-- https://mvnrepository.com/artifact/com.sun.mail/javax.mail -->

<dependency>

<groupId>com.sun.mail</groupId>

<artifactId>javax.mail</artifactId>

<version>1.6.2</version>

</dependency>

<!-- STEP 52 : RAZOR PAY PAYMENT INTEGARATION , put the latest versionm by copy code from github link there, -->

<dependency>

<groupId>com.razorpay</groupId>

<artifactId>razorpay-java</artifactId>

<version>1.3.9</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**1.application.properties**

server.port=8282

spring.datasource.name=phonebook

spring.datasource.url=jdbc:mysql://localhost:3306/phonebook

spring.datasource.username=root

spring.datasource.password=root

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL55Dialect

spring.jpa.hibernate.ddl-auto=update

#if error comes in uploading file size image the use these

spring.servlet.multipart.max-file-size=-1

spring.servlet.multipart.max-request-size=-1

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

2. SmartcontactmanagerApplication.java

package com.smart;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SmartcontactmanagerApplication {

public static void main(String[] args)

{

SpringApplication.run(SmartcontactmanagerApplication.class, args);

}

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

3. CustomUserDetails.java

package com.smart.config;

import java.util.Collection;

import java.util.List;

import org.springframework.security.core.GrantedAuthority;

import org.springframework.security.core.authority.SimpleGrantedAuthority;

import org.springframework.security.core.userdetails.UserDetails;

import com.smart.entities.User;

public class CustomUserDetails implements UserDetails

{

private User user;

public CustomUserDetails(User user) {

super();

this.user = user;

}

@Override

public Collection<? extends GrantedAuthority> getAuthorities() {

SimpleGrantedAuthority simpleGrantedAuthority = new SimpleGrantedAuthority(user.getRole());

return List.of(simpleGrantedAuthority);

}

@Override

public String getPassword() {

return user.getPassword();

}

@Override

public String getUsername() {

return user.getEmail(); //email used as the username

}

@Override

public boolean isAccountNonExpired() {

return true;

}

@Override

public boolean isAccountNonLocked() {

return true;

}

@Override

public boolean isCredentialsNonExpired() {

return true;

}

@Override

public boolean isEnabled() {

return true;

}

}

**4.** MyConfig.java

package com.smart.config;

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.AuthenticationManagerBuilder;

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.WebSecurityConfigurerAdapter;

import org.springframework.security.core.userdetails.UserDetailsService;

import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

@Configuration

@EnableWebSecurity

public class MyConfig extends WebSecurityConfigurerAdapter {

@Bean

public UserDetailsService getUserDetailService()

{

return new UserDetailsServiceImpl();

}

@Bean

public BCryptPasswordEncoder passwordEncoder()

{

return new BCryptPasswordEncoder();

}

@Bean

public DaoAuthenticationProvider authenticationProvider()

{

DaoAuthenticationProvider daoAuthenticationProvider= new DaoAuthenticationProvider();

daoAuthenticationProvider.setUserDetailsService(this.getUserDetailService());;

daoAuthenticationProvider.setPasswordEncoder(passwordEncoder()); ///////////////

return daoAuthenticationProvider;

}

//configure method

@Override

protected void configure(AuthenticationManagerBuilder auth) throws Exception {

auth.authenticationProvider(authenticationProvider());

}

@Override

protected void configure(HttpSecurity http) throws Exception {

http

.authorizeHttpRequests()

.antMatchers("/admin/\*\*")

.hasRole("ADMIN")

.antMatchers("/user/\*\*")

.hasRole("USER")

.antMatchers("/\*\*")

.permitAll()

.and()

.formLogin() //form based login system is used , which has feature of Logout

.loginPage("/signin") //our own login page designed

.loginProcessingUrl("/dologin") //On it the form need to be submitter, if success, it shows the user/index page , if fail the redirect to same page //after this the default page of login will not come, instead our designed willcome .

.defaultSuccessUrl("/user/index")

//.failureUrl("/login-fail") //can desing this page externally

.and()

.csrf().disable();

}

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

5. UserDetailsServiceImpl.java

package com.smart.config;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.security.core.userdetails.UserDetails;

import org.springframework.security.core.userdetails.UserDetailsService;

import org.springframework.security.core.userdetails.UsernameNotFoundException;

import com.smart.dao.UserRepository;

import com.smart.entities.User;

public class UserDetailsServiceImpl implements UserDetailsService {

@Autowired

private UserRepository userRepository;

@Override

public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {

//fetching data from database

User user = userRepository.getUserByUserName(username); //username is email only

if(user==null)

{throw new UsernameNotFoundException("Could not found user !!");

}

CustomUserDetails customUserDetails= new CustomUserDetails(user);

return customUserDetails;

}

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

6. EmailController.java

package com.smart.controller;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

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

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

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

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

import com.smart.entities.EmailRequest;

import com.smart.service.EmailService;

@RestController

public class EmailController {

@Autowired

private EmailService emailService;

@RequestMapping("/welcome")

public String welcome()

{

return " Namaste ! this is my email.api ";

}

// HAndler to send Email

@RequestMapping(value="/sendemail" ,method=RequestMethod.POST)

public ResponseEntity<?> sendEmail(@RequestBody EmailRequest request)

{

System.out.println(request);

boolean results=this.emailService.sendEmail(request.getSubject(),request.getMessage(),request.getTo());

if(results) {

return ResponseEntity.ok("Email is sent successfully... :: ");

}

else {

return ResponseEntity.status(HttpStatus.INTERNAL\_SERVER\_ERROR).body("Email not sent");

}

//System.out.println(request);

//return ResponseEntity.ok("dOne :: ");

//rdgjwaxsbwejxsiz

}

}

**7.** ForgotController.java :

package com.smart.controller;

import java.util.Random;

import javax.servlet.http.HttpSession;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

import org.springframework.stereotype.Controller;

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

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

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

import com.smart.dao.UserRepository;

import com.smart.entities.User;

import com.smart.service.EmailService;

@Controller

public class ForgotController {

@Autowired

EmailService emailService; //step : 45 sending the mail

@Autowired //need to verify the user , upon validation of otp

UserRepository userRepository;

@Autowired

BCryptPasswordEncoder bCryptPasswordEncoder; //to encode new password

//STEP 44:

@RequestMapping("/forgot")

public String openEmailForm()

{

return "forgot\_email\_form";

}

//STEP : 45

//handle reset password form submission

@PostMapping("/send-otp")

public String sendOtp(@RequestParam("email") String email, HttpSession session )

{

System.out.println("Email is : "+email);

//Generating the random no and send its as otp , also store that random at otp at session, to validate it with the user send otp:

Random random= new Random();

int otp = random.nextInt(999, 999999);

System.out.println(otp);

// STEP 46 : OTP SEND CODE USING EMAIL SERVICE

//Done with includeing the email service class, and importing the dependency

String subject=" Reset Password-OTP from PhoneBook Cloud";

String to=email;

//to enable html here change in Emailserive , s.sentContext(message,"text/html");

String message=""

+ "<div style='border:1px solid #e2e2e2; padding:20px'>"

+"<h1>"

+"OTP is "

+"<b>"

+ otp

+ "</b>"

+"</h1>"

+"</div>";

boolean flag = this.emailService.sendEmail(subject, message, email);

if(flag) //means no errors OTP Send successfully so

{ //Can also store this in database

session.setAttribute("dotp",otp); //STORING it in session , to verify it when user submit its otp

session.setAttribute("email", email);

return "verify\_otp";

}

else {

//when email not sent , then error displayed on the forgot email page.

session.setAttribute("message","Check your email id ");

return "forgot\_email\_form";

}

// page to allow user to put the send otp to the mail

}

//\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//STEP : 46 : VERFIy the OTP submitted by user

@PostMapping("/verify-otp")

public String verifyOtp(@RequestParam("otp") int otp , HttpSession session)

{

//getting the original random generated otp using the session : and veriy it with the OTP SEND by user , ie recieved in the @requestParam otp

int dotp=(int)session.getAttribute("dotp");

String email=(String)session.getAttribute("email");

System.out.println("DATAbase otp " + dotp);

System.out.println("USer otp "+ otp);

System.out.println("USer email "+ email);

if(dotp==otp)

{

User user = this.userRepository.getUserByUserName(email);

System.out.println(" CUrrent user "+ user);

if(user==null) //user does not exists

{

//TIME 16:05

session.setAttribute("message","User does not exists :");

return "forgot\_email\_form";

}

else {

//valid person , allow to set new password.

return "change\_password";

}

}

else {

session.setAttribute("message", "You have entered Wrong Otp ");

return "verify\_otp";

}

}

//--------------------------------------------------------

// STEP : 47

//Handler for change password :

@PostMapping("/change-password")

public String changPassword(@RequestParam("new\_password") String newpassword ,HttpSession session)

{

//fetch the email id from sessopm and set this new user password;

String email = (String) session.getAttribute("email");

User user = this.userRepository.getUserByUserName(email);

user.setPassword(bCryptPasswordEncoder.encode(newpassword));

this.userRepository.save(user);

System.out.println("Final stage : password Changed");

// session.setAttribute("message", " Password Changed SuccessFully ");

// return "verify\_otp";

return "redirect:/signin?change=password changed successfully"; //apply if cond to show this on login page

}

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

8. HomeController.java

package com.smart.controller;

import javax.servlet.http.HttpSession;

import javax.validation.Valid;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.validation.BindingResult;

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.smart.dao.UserRepository;

import com.smart.entities.User;

import com.smart.helper.Message;

@Controller

public class HomeController {

@Autowired

private BCryptPasswordEncoder passwordEncoder;

@Autowired

private UserRepository userRepository ;

@RequestMapping("/")

public String home(Model m)

{

m.addAttribute("title","Home -PhoneBook Cloud");

return "home";

}

@RequestMapping("/about")

public String about(Model m)

{

m.addAttribute("title","About -PhoneBook Cloud");

return "about";

}

@RequestMapping("/signup")

public String signup(Model m)

{

m.addAttribute("title","Register Here -PhoneBook Cloud");

m.addAttribute("user",new User());

return "signup";

}

//Handler for handle form / register user

@PostMapping("/do\_register")

public String registerUser(@Valid @ModelAttribute("user") User user, BindingResult result1 , @RequestParam(value="agreement", defaultValue = "false") Boolean agreement,Model model ,HttpSession session)

{

try {

if(!agreement )

{

System.out.println("U have not agreed the terms and conditions");

throw new Exception("U have not agreed the terms and conditions");

}

if(result1.hasErrors())

{

System.out.println("ERROR "+ result1.toString());

model.addAttribute("user",user);

return "signup";

}

user.setRole("ROLE\_USER");

user.setEnabled(true);

user.setImageUrl("default.png");

user.setPassword(passwordEncoder.encode(user.getPassword()));

System.out.println("Agreement "+ agreement );

System.out.println("USER "+ user);

//save data to database

User result = this.userRepository.save(user);

model.addAttribute("user",new User());

session.setAttribute("message", new Message("SuccessFully Registered !!" ,"alert-success"));

return "signup";

}catch (Exception e) {

e.printStackTrace();

model.addAttribute("user",user);

session.setAttribute("message", new Message("Something went woring !! "+ e.getMessage(), "alert-danger "));

return "signup";

}

}

@GetMapping("/signin")

public String customLogin(Model m)

{

m.addAttribute("title","The Login Page");

return "login";

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**9.** SearchController.java

package com.smart.controller;

import java.security.Principal;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.ResponseEntity;

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

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

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

import com.smart.dao.ContactRepository;

import com.smart.dao.UserRepository;

import com.smart.entities.Contact;

import com.smart.entities.User;

@RestController //its return the json / string return directly, not as a view

public class SearchController {

@Autowired

private UserRepository userRepository;

@Autowired

private ContactRepository contactRepository;

//search handler

@GetMapping("/search/{query}")

public ResponseEntity<?> search(@PathVariable("query") String query , Principal principal)

{

System.out.println(query);

User user = this.userRepository.getUserByUserName(principal.getName());

List<Contact> contacts = this.contactRepository.findByNameContainingAndUser(query, user);

return ResponseEntity.ok(contacts);

}

}

/\*

//TO CHECK THE DATABASE CONNECTIVITY OVER HERE

@Autowired

private UserRepository userRepository;

@GetMapping("/test")

@ResponseBody

public String test()

{

User user= new User();

user.setName("Shivam");

user.setEmail("sjh14@gmail.com");

userRepository.save(user); //THIS WAS THE ERROR I WAS MISSING

return "Working DAta may to added in database";

}

\*/

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

10. TestController.java

package com.smart.controller;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

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

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

import com.smart.dao.UserRepository;

import com.smart.entities.User;

@Controller

public class TestController {

@Autowired

UserRepository userRepository;

@GetMapping("/test")

@ResponseBody

public String test()

{

User user= new User();

user.setEmail("Shivam@Gami.com");

user.setId(1);

user.setName("Shivam");

user.setPassword("Shivam");

user.setRole("ROLE\_NORMAL");

userRepository.save(user);

return "This is just for testing";

}

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

11. UserController.java

package com.smart.controller;

import java.io.File;

import java.nio.file.Files;

import java.nio.file.Path;

import java.nio.file.Paths;

import java.nio.file.StandardCopyOption;

import java.security.Principal;

import java.util.Date;

import java.util.Map;

import java.util.Optional;

import javax.servlet.http.HttpSession;

import org.json.JSONObject;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.core.io.ClassPathResource;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.PageRequest;

import org.springframework.data.domain.Pageable;

import org.springframework.http.ResponseEntity;

import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

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.PathVariable;

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

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.multipart.MultipartFile;

import com.razorpay.Order;

import com.razorpay.RazorpayClient;

import com.smart.dao.ContactRepository;

import com.smart.dao.MyOrderRepository;

import com.smart.dao.UserRepository;

import com.smart.entities.Contact;

import com.smart.entities.MyOrder;

import com.smart.entities.User;

import com.smart.helper.Message;

@Controller

@RequestMapping("/user")

public class UserController {

@Autowired //to save the order id generated in PAYMENT INTEGRATION (RAZORY PAY)

private MyOrderRepository myOrderRepository;

@Autowired

BCryptPasswordEncoder bCryptPasswordEncoder; //used for change password module

@Autowired

private UserRepository userRepository;

@Autowired

private ContactRepository contactRepository;

/\* ------------------------------------------------- \*/

// adding the common hander to assign the logged in : user to all the handler to use it;

//STEPP : 20

@ModelAttribute

public void addCommonData(Model model, Principal principal)

{

String username = principal.getName();

System.out.println("USER\_NAME"+username);

User user= userRepository.getUserByUserName(username);

model.addAttribute("user",user);

System.out.println(user);

}

/\* ------------------------------------------------- \*/

//dashboard home

@RequestMapping("/index")

public String dashboard(Model model , Principal principal)

{

model.addAttribute("title","User DashBoard");

String username = principal.getName();

User user = userRepository.getUserByUserName(username);

model.addAttribute("person",user.getName());

model.addAttribute("date", new Date());

return "normal/user\_dashboard";

}

/\* ------------------------------------------------- \*/

// open the add form handler //STEEP 21 : taking common user for all handlers and desing the form .

@GetMapping("/add-contact")

public String openAddContactForm(Model model)

{

model.addAttribute("title","Add Contact");

model.addAttribute("contact", new Contact());

return "normal/add\_contact\_form";

}

/\* ------------------------------------------------- \*/

//STEP 23 : Submit Contact Form

//handle the submission of Contact Form of action : /user/process-contact : user already included, so handler the process-contact only

@PostMapping("/process-contact")

public String processContact(@ModelAttribute Contact contact ,

Principal principal,

@RequestParam("profileImage") MultipartFile file,

HttpSession session

)

{

System.out.println(file.getOriginalFilename());

try {

String name=principal.getName();

User user = this.userRepository.getUserByUserName(name);

//to test the error cond we can throw error ef if(3>2) throw new Exception();

//STEPP 24 : ADDING CONTACT IMAGE

// profile image processing and uploading

// The file will be addded to target folder / its folder after the project built

if(file.isEmpty())

{

System.out.println("File is empty");

//set default photo

contact.setImage("contact.png");

}

else{

//set the file to folder ,and put in contact field of image.

contact.setImage(file.getOriginalFilename());

File saveFile = new ClassPathResource("static/img").getFile();

Path path=Paths.get(saveFile.getAbsolutePath()+File.separator+file.getOriginalFilename());

Files.copy(file.getInputStream(),path,StandardCopyOption.REPLACE\_EXISTING);

System.out.println("File uploaded successfully");

}

contact.setUser(user); // saved as bi directional mapping exists

user.getContacts().add(contact); // contact added to user

this.userRepository.save(user);

System.out.println("DATA"+ contact);

System.out.println("Contact added to database");

//message success added successfully

session.setAttribute("message", new Message("Your Contact is Added !! Add More","success")); //use the helper class to send the content and type with msg

} catch (Exception e) {

System.out.println("ERROR : "+ e.getMessage());

e.printStackTrace();

//message error : user not added

session.setAttribute("message", new Message("Some Thing went worong !!Try Again","danger")); //use the helper class to send the content and type with msg

}

return "normal/add\_contact\_form";

}

/\* ----------------------------- ------------------- \*/

//STEP 24

//Show contacts Handler

@GetMapping("/show-contacts/{page}")

public String showContacts(@PathVariable("page") Integer page,Model m,Principal principal)

{

m.addAttribute("title","Show User Contacts");

//STEP 26

//send the list of contact from database

//make Seperate Contact Repository -> helpful in creating the pagenateion Concepts

String userName = principal.getName();

User user = this.userRepository.getUserByUserName(userName);

Pageable pageable = PageRequest.of(page, 5); //Pageable is parent interface

Page<Contact> contacts= this.contactRepository.findContactByUser(user.getId(),pageable);

m.addAttribute("contacts",contacts);

m.addAttribute("currentPage",page);

m.addAttribute("totalPages",contacts.getTotalPages());

//STEP 27 PAGE-NATION

//per page-5 contacts (n)

//current page- 0 (page)

//Use path varaible to get the current page. ALso return the page from the ContactRepository.

//Change return type to page type and addAttibutes of current page etc

return "normal/show\_contacts";

}

/\*----------------------------- ------------------- \*/

// STEPP 30 : Handler for the email click to show user profile

@GetMapping("/{cid}/contact")

public String showContactDetail(@PathVariable("cid") Integer cid , Model model,Principal principal)

{

//System.out.println("CID "+cid);

Optional<Contact> contactOptional = this.contactRepository.findById(cid);

Contact contact = contactOptional.get();

String username = principal.getName();

User user = this.userRepository.getUserByUserName(username);

//STEPP 31: clearing bug of random contact-access

if(user.getId() == contact.getUser().getId()) //send the valid contactss to users onlu

{

model.addAttribute("contact",contact);

model.addAttribute("title",contact.getName());

}

return "normal/contact\_detail";

}

/\*----------------------------- ------------------- \*/

//STEPP : 32

@GetMapping("/delete/{cid}")

public String deleteContact(@PathVariable("cid") Integer cid ,Model m , HttpSession session ,Principal principal)

{

System.out.println("CID : "+ cid);

Contact contact = this.contactRepository.findById(cid).get();

//apply check that , person deleteing the contact should be actual logged in user.

//unlink the user , problem due to cascade can be resolved

//STEP :35 REMOVING THE CONTACT FROM DATA BASE USE the

User user = this.userRepository.getUserByUserName(principal.getName());

user.getContacts().remove(contact); //TO remove the current contact of Delete button , JVM interanlly uses the equals method of COntact class , so override it.

this.userRepository.save(user);

session.setAttribute("message", new Message("Contact deleted Successfully" , "success"));

return "redirect:/user/show-contacts/0";

}

/\*----------------------------- ------------------- \*/

//STEPP 33

//open update form folder

//post is secure mapping , so safety is ensured.

@PostMapping("/update-contact/{cid}")

public String updateForm(@PathVariable("cid") Integer cid, Model m)

{

m.addAttribute("title","Update Contact");

Contact contact = this.contactRepository.findById(cid).get();

m.addAttribute("contact",contact);

return "/normal/update\_form"; //Will Show the data on new form for the update opeitonss

}

/\*----------------------------- ------------------- \*/

//STEPP :33

// the update form submission

@PostMapping("/process-update")

public String updateHandler(@ModelAttribute Contact contact, @RequestParam("profileImage") MultipartFile file, Model m , HttpSession session,Principal principal)

{

try {

//fetch old contact details

Contact oldContactDetail = this.contactRepository.findById(contact.getCid()).get();

//if new image chosen

if(!file.isEmpty())

{

//process file - rewrite

//1 detlete the old photo from folder : get the path

File deletefile = new ClassPathResource("static/img").getFile();

File file1= new File(deletefile,oldContactDetail.getImage());

file1.delete();

//2.update the new Photo in file

File savefile = new ClassPathResource("static/img").getFile();

Path path = Paths.get(savefile.getAbsolutePath()+File.separator+ file.getOriginalFilename());

Files.copy(file.getInputStream(),path,StandardCopyOption.REPLACE\_EXISTING);

contact.setImage(file.getOriginalFilename()); //saving the name in database;

} //file is empty only : same as old

else {

contact.setImage(oldContactDetail.getImage());

}

//saving the current user, to reflect bacj the changes of updated contact.

User user = this.userRepository.getUserByUserName(principal.getName());

contact.setUser(user);

this.contactRepository.save(contact);

session.setAttribute("message",new Message("Your contact updated ","success") );

} catch (Exception e) {

}

System.out.println("CONTACT"+contact.getName());

return "redirect:/user/"+contact.getCid() + "/contact";

}

/\*----------------------------- ------------------- \*/

//STEP 37 :

// your profile handler

@GetMapping("/profile") // /user/profile : it will be actually like this

public String yourProfile(Model m)

{

m.addAttribute("title","Profile Page");

return "normal/profile";

}

/\*----------------------------- ------------------- \*/

//STEP 41 :

// your settings handler

@GetMapping("/settings") // /user/profile : it will be actually like this

public String settings(Model m)

{

m.addAttribute("title","Settings Page");

return "normal/settings";

}

/\*----------------------------- ------------------- \*/

//STEP 42 :

// change-password handler : to accept the change password form

//already user included at top

@PostMapping("/change-password")

public String changepassword(@RequestParam("oldPassword") String oldPassword , @RequestParam("newPassword") String newPassword , Principal principal ,HttpSession session)

{

// System.out.println("OLd Password "+ oldPassword);

//System.out.println("New Password "+ newPassword);

//Checking the old password is correct or not

String userName = principal.getName();

User currentUser = this.userRepository.getUserByUserName(userName);

//System.out.println(currentUser.getPassword());

if(this.bCryptPasswordEncoder.matches(oldPassword, currentUser.getPassword()))

{

//valid : change with ne

currentUser.setPassword(bCryptPasswordEncoder.encode(newPassword));

this.userRepository.save(currentUser);

//To send the message to user take Http SEssion

session.setAttribute( "message", new Message(" Password Changed Successfully","success"));

return "redirect:/user/index" ; //return to home view after pass change

}

else{

session.setAttribute( "message", new Message(" Incorrect Old Password !!","danger"));

return "redirect:/user/settings" ;

}

}

/\*----------------------------- ------------------- \*/

//STEP 53 RAZOR PAY : creating order for the payment.

//ajax fun has url : user/create\_order

@PostMapping("/create\_order")

@ResponseBody

public String createOrder(@RequestBody Map<String, Object> data , Principal principal) throws Exception

{

System.out.println("create order function executed .");

//receiving the amount value in map object ()

System.out.println(data);

//take out the amount

int amt = Integer.parseInt(data.get("amount").toString());

System.out.println( amt);

var client=new RazorpayClient("rzp\_test\_8TFsUYjZM9VXJ5","9OIqnSVEJhunoCh67JJmGdIG");

//order generate by the razor pay server

JSONObject ob= new JSONObject();

ob.put("amount",amt\*100);

ob.put("currency","INR");

ob.put("receipt","txn\_23354");

//creating new order by the r pay server

Order order = client.Orders.create(ob); //Error came in

System.out.println(order);

//SAVE THE INFO OF order IN DATABASE FOR FUTURE REFRENCE.

//link the MyOrderRepository to save the data-

//STEP 55 :

MyOrder myOrder = new MyOrder();

myOrder.setAmount(order.get("amount")+"");

myOrder.setOrderId(order.get("id"));

myOrder.setPaymentId(null);

myOrder.setStatus("created");

myOrder.setReceipt(order.get("receipt"));

myOrder.setUser(this.userRepository.getUserByUserName(principal.getName()));

this.myOrderRepository.save(myOrder);

// Look the transaction in Razory Pay , Orders , it will be refleected their

// Now we will return this details of the order generated to the client, as next step

// so return order.toString(); , next Step client will directly request / send money to the Razor pay

return order.toString(); //JSON is sent back to client . on console of browser

//Next step 53 : to show the Razor pay form in js file. to make payment.

//return "done";

}

/\*----------------------------- ------------------- \*/

//STEP : 57

//Handler the request to save the Order id Razor Pay payement details .

@PostMapping("/update\_order" ) //handle the ajax request to handle the form update for order of RAzor Pay Payment.

public ResponseEntity<?> updateOrder(@RequestBody Map<String,Object> data)

{

//STEP : 58 Make a cutomfindMethod in MyOrderREepository Inteface .

MyOrder myOrder = this.myOrderRepository.findByOrderId(data.get("order\_id").toString());

myOrder.setPaymentId(data.get("payment\_id").toString());

myOrder.setStatus(data.get("status").toString());

this.myOrderRepository.save(myOrder);

System.out.println( "Final data :"+data);

System.out.println("Final MyOrder "+ myOrder);

return ResponseEntity.ok(Map.of("msg","updated"));

}

/\*----------------------------- ------------------- \*/

/\*----------------------------- ------------------- \*/

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

12. ContactRepository .java

package com.smart.dao;

import java.util.List;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.Pageable;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Query;

import org.springframework.data.repository.query.Param;

import com.smart.entities.Contact;

import com.smart.entities.User;

//STEP 26

public interface ContactRepository extends JpaRepository<Contact, Integer>

{

//Helpful in pagenation Concepts

//writing manual query to fetch the contacts of LOgged in User

@Query("from Contact as c where c.user.id=:userId")

public Page<Contact> findContactByUser(@Param("userId") int userId , Pageable pageable);

//STEP 28 :

//changing the return type of Func to Page<Contact> from list type. ans also addding parameter of Pageble

//Pageble has (current page, contact per page)

//STEP 35

//SEARCH BAR

//mathc the Contact , which matches with current User

public List<Contact> findByNameContainingAndUser(String name,User user); //springboot itslef gives the defination of this funtion.

}

/\*

\*

//writing manual query to fetch the contacts of LOgged in User

@Query("from Contact as c where c.user.id=:userId")

public List<Contact> findContactByUser(@Param("userId") int userId);

\*

\*

\*/

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

13. MyOrderRepository.java

package com.smart.dao;

import org.springframework.data.jpa.repository.JpaRepository;

import com.smart.entities.MyOrder;

public interface MyOrderRepository extends JpaRepository<MyOrder,Long >

{

//STEP : 57

public MyOrder findByOrderId(String orderId);

//THe field name should follow Camel case eg private orderId , then can use method as findByOrderId(String orderId);

//Springboot will internally resolve this method and give the required results.

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

14. UserRepository.java

package com.smart.dao;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Query;

import org.springframework.data.repository.query.Param;

import com.smart.entities.User;

//No need to use @service , as its inside out main smart package.

public interface UserRepository extends JpaRepository<User,Integer>{

//write jpa query

@Query("select u from User u where u.email=:email")

public User getUserByUserName(@Param("email") String email);

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

15. Contact.java

package com.smart.entities;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.ManyToOne;

import javax.persistence.Table;

import com.fasterxml.jackson.annotation.JsonIgnore;

@Entity

@Table(name="CONTACT")

public class Contact {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private int cid;

private String name;

private String secondName;

private String work;

private String email;

private String phone;

private String image;

@Column(length=1000)

private String description;

@ManyToOne()

@JsonIgnore //it will not serrialize , othewise cycle : Search-Controller

private User user;

public Contact() {

super();

// TODO Auto-generated constructor stub

}

public Contact(int cid, String name, String secondName, String work, String email, String phone, String image,

String description) {

super();

this.cid = cid;

this.name = name;

this.secondName = secondName;

this.work = work;

this.email = email;

this.phone = phone;

this.image = image;

this.description = description;

}

public int getCid() {

return cid;

}

public void setCid(int cid) {

this.cid = cid;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getSecondName() {

return secondName;

}

public void setSecondName(String secondName) {

this.secondName = secondName;

}

public String getWork() {

return work;

}

public void setWork(String work) {

this.work = work;

}

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 getImage() {

return image;

}

public void setImage(String image) {

this.image = image;

}

public String getDescription() {

return description;

}

public void setDescription(String description) {

this.description = description;

}

public User getUser() {

return user;

}

public void setUser(User user) {

this.user = user;

}

/\*

\* @Override public String toString() { return "Contact [cid=" + cid + ", name="

\* + name + ", secondName=" + secondName + ", work=" + work + ", email=" + email

\* + ", phone=" + phone + ", image=" + image + ", description=" + description +

\* "]"; }

\*/

//Over ridded to enable the delete functionality of Contacts from database

@Override

public boolean equals(Object obj)

{

return this.cid==((Contact)obj).getCid();

}

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

16. EmailRequest.java

**package** com.smart.entities;

**public** **class** EmailRequest {

**private** String to;

**private** String subject;

**private** String message;

**public** EmailRequest() {

**super**();

// **TODO** Auto-generated constructor stub

}

**public** EmailRequest(String to, String subject, String message) {

**super**();

**this**.to = to;

**this**.subject = subject;

**this**.message = message;

}

**public** String getTo() {

**return** to;

}

**public** **void** setTo(String to) {

**this**.to = to;

}

**public** String getSubject() {

**return** subject;

}

**public** **void** setSubject(String subject) {

**this**.subject = subject;

}

**public** String getMessage() {

**return** message;

}

**public** **void** setMessage(String message) {

**this**.message = message;

}

@Override

**public** String toString() {

**return** "EmailRequest [to=" + to + ", subject=" + subject + ", message=" + message + "]";

}

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**17. MyOrder.java**

package com.smart.entities;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.ManyToOne;

import javax.persistence.Table;

@Entity

@Table(name="orders")

public class MyOrder

{

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private long myorderId;

private String orderId;

private String amount;

private String receipt;

private String status;

@ManyToOne // ie is one user can, have the mutilple payments.

private User user;

private String paymentId; //to store the payment status:

public long getMyorderId() {

return myorderId;

}

public void setMyorderId(long myorderId) {

this.myorderId = myorderId;

}

public String getOrderId() {

return orderId;

}

public void setOrderId(String orderId) {

this.orderId = orderId;

}

public String getAmount() {

return amount;

}

public void setAmount(String amount) {

this.amount = amount;

}

public String getReceipt() {

return receipt;

}

public void setReceipt(String receipt) {

this.receipt = receipt;

}

public String getStatus() {

return status;

}

public void setStatus(String status) {

this.status = status;

}

public User getUser() {

return user;

}

public void setUser(User user) {

this.user = user;

}

public String getPaymentId() {

return paymentId;

}

public void setPaymentId(String paymentId) {

this.paymentId = paymentId;

}

@Override

public String toString() {

return "MyOrder [myorderId=" + myorderId + ", orderId=" + orderId + ", amount=" + amount + ", receipt="

+ receipt + ", status=" + status + ", user=" + user + ", paymentId=" + paymentId + "]";

}

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

18. User.java

package com.smart.entities;

import java.util.ArrayList;

import java.util.List;

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.OneToMany;

import javax.persistence.Table;

import javax.validation.constraints.NotBlank;

import javax.validation.constraints.Size;

@Entity

@Table(name="USER")

public class User {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private int id;

//@NotBlank(message="Name is required")

//@Size(min=2,message="min 2 and max 20 char are allowed!!")

private String name;

// @Column(unique=true)

private String email;

private String password;

private String role;

private boolean enabled;

private String imageUrl;

//@Column(length= 500 )

private String about;

// GIVIGNG mapping

//mappedBy is used , so that no extra table is created of user\_contact, only user manages the whole //\*NI foreignkey managegment will be done through userid field on Contact table.

//@OneToMany(cascade = CascadeType.ALL , fetch = FetchType.LAZY , mappedBy = "user",orphanRemoval = true)

@OneToMany(cascade = CascadeType.ALL , mappedBy = "user",orphanRemoval = true)

private List<Contact> contacts= new ArrayList<>();

////////////////////////////////////////

public User(int id, String name, String email, String password, String role, boolean enabled, String imageUrl,

String about, List<Contact> contacts) {

super();

this.id = id;

this.name = name;

this.email = email;

this.password = password;

this.role = role;

this.enabled = enabled;

this.imageUrl = imageUrl;

this.about = about;

this.contacts = contacts;

}

public User() {

super();

}

public List<Contact> getContacts() {

return contacts;

}

public void setContacts(List<Contact> contacts) {

this.contacts = contacts;

}

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;

}

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 String getRole() {

return role;

}

public void setRole(String role) {

this.role = role;

}

public boolean isEnabled() {

return enabled;

}

public void setEnabled(boolean enabled) {

this.enabled = enabled;

}

public String getImageUrl() {

return imageUrl;

}

public void setImageUrl(String imageUrl) {

this.imageUrl = imageUrl;

}

public String getAbout() {

return about;

}

public void setAbout(String about) {

this.about = about;

}

@Override

public String toString() {

return "User [id=" + id + ", name=" + name + ", email=" + email + ", password=" + password + ", role=" + role

+ ", enabled=" + enabled + ", imageUrl=" + imageUrl + ", about=" + about + ", contacts=" + contacts

+ "]";

}

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**19. Message.java**

**package** com.smart.helper;

**public** **class** Message {

**private** String content;

**private** String type;

**public** Message() {

**super**();

// **TODO** Auto-generated constructor stub

}

**public** Message(String content, String type) {

**super**();

**this**.content = content;

**this**.type = type;

}

**public** String getContent() {

**return** content;

}

**public** **void** setContent(String content) {

**this**.content = content;

}

**public** String getType() {

**return** type;

}

**public** **void** setType(String type) {

**this**.type = type;

}

@Override

**public** String toString() {

**return** "Message [content=" + content + ", type=" + type + "]";

}

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**20. EmailService.java**

package com.smart.service;

import java.util.Properties;

import javax.mail.Authenticator;

import javax.mail.Message;

import javax.mail.PasswordAuthentication;

import javax.mail.Session;

import javax.mail.Transport;

import javax.mail.internet.InternetAddress;

import javax.mail.internet.MimeMessage;

import org.springframework.stereotype.Service;

@Service //or use it @Component

public class EmailService {

public boolean sendEmail(String subject,String message,String to)

{

//copy paste the code from sending mail in java video

boolean flag=false;

String from="shivamraiias@gmail.com";

//variable for gmail.

String host="smtp.gmail.com";

//get the system properties

Properties properties=System.getProperties();

System.out.println("PROPERTIES"+properties);

//setting imp info to properties object.

//host set

properties.put("mail.smtp.host",host);

properties.put("mail.smtp.port","465");

properties.put("mail.smtp.ssl.enable","true");

properties.put("mail.smtp.auth","true");

//STEP 1 : to gt the sesssion object:

Session session=Session.getInstance(properties,new Authenticator()

{

@Override

protected PasswordAuthentication getPasswordAuthentication()

{

return new PasswordAuthentication("shivamraiias@gmail.com", "rdgjwaxsbwejxsiz"); //This was creating the error , hence random password of two factor authentication is generated and then put here. Refer Stack OverFlow.

}

});

session.setDebug(true);

//Step 2 : compare the message [text , multi media]

MimeMessage m = new MimeMessage(session);

try {

//from email

m.setFrom(from);

//adding recipient to message

m.addRecipient(Message.RecipientType.TO,new InternetAddress(to));

//adding subject to message

m.setSubject(subject);

//adding text to message

//m.setText(message);

//To send the Html in message

m.setContent(message,"text/html");

//STEP 3 : send the message using Transport class

Transport.send(m);

System.out.println("Sent success................");

flag=true;

}catch (Exception e)

{

e.printStackTrace();

}

return flag;

}

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**21.style.css**

*.a* {

font-style: *oblique*;

}

**body**{

background:*#e2e2e2!important*;

}

*.bg-primary*{

background: *#009688* *!important*;

}

*.banner*{

width:*100%*;

height:*93vh*;

background: *linear-gradient(rgba(0,0,0,.2),rgba(0,0,0,.2))* *,url(../img/hos3.jpg)*;

background-size:*cover*;

color:*white*;

}

*.my-card*{

padding: *20px*;

background:*#FFFFFF*;

}

*#login* *.container* *#login-row* *#login-column* *#login-box* {

margin-top: *120px*;

max-width: *600px*;

height: *450px*;

border: *1px* *solid* *#9C9C9C*;

background-color:*white*;

}

*#login* *.container* *#login-row* *#login-column* *#login-box* *#login-form* {

padding: *20px*;

}

*#login* *.container* *#login-row* *#login-column* *#login-box* *#login-form* *#register-link* {

margin-top: *-85px*;

}

/\* SIDE BAR STYLING OF USER\_DASHBOARD \*/

@media only screen and (max-width :600px)

{

*.sidebar*

{

display:*none*;

width:*80%* *!important*;

background:*white*;

height:*100vh*;

position:*fixed*;

padding-top:*20px*;

}

*.content*

{

margin-left:*0%!important*;

}

}

*.sidebar*

{

width:*18%*;

background:*white*;

height:*100%*;

position:*fixed*;

padding-top:*45px*;

z-index:*99999999999999999999999999*;

}

*.sidebar* **a**{

display:*block*;

width:*100%*;

padding: *10px* *25px*;

text-decoration:*none*;

color:*gray*;

}

*.sidebar* **a***:hover*{

background:*#e2e2e2*;

}

*.sidebar* *.divider*

{

height: *.5px*;

width:*100px*;

background: *#e2e2e2*;

margin-top:*10px*;

}

*.sidebar* *.crossBtn*

{

position:*absolute*;

top:*0*;

right:*35px*;

font-size:*36px*;

cursor:*pointer*;

}

*.content*

{

margin-left:*20%*;

}

/\* setting profile photo \*/

*.my\_profile\_picture*

{

width:*35px*;

height:*35px*;

object-fit:*cover*; /\* To adjust all types iamges \*/

border-radius:*50%*;

border: *1px* *solid* *black*;

}

/\* SEARCH BOX \*/

*.search-container*

{

position:*relative*;

}

*.search-result*

{

position:*absolute*;

background:*white*;

width:*100%*;

padding:*10px*;

border: *1px* *solid* *gray*;

display:*none*;

}

*.active*

{

background: *#009688* *!important*;

color:*white* *!important*;

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

22. script.js

//console.log("this is script file and i had attached it here");

// write hiding configureation for side bar

**const** toggleSidebar=() **=>** {

if($('.sidebar').is(":visible"))

{

$(".sidebar").css("display","none");

$(".content").css("margin-left","0%");

}

else

{

$(".sidebar").css("display","block");

$(".content").css("margin-left","20%");

}

};

//STEP : 35

//writint the configuration for search Controller

**const** search=() **=>**{

//console.log("searching...");

**let** query=$("#search-input").val(); //attach the jquery if, not presnt

console.log(query);

if(query==''){

$(".search-result").hide();

}

else{

console.log(query);

//sending request to server

**let** url=`http://localhost:8282/search/${query}`;

fetch(url)

.then((response) **=>** {

return response.json();

})

.then((data) **=>** {

// console.log(data);

**let** text=`<div class='list-group'>`;

data.forEach((contact) **=>** {

text+=`<a href="/user/${contact.cid}/contact" class="list-group-item list-group-action">${contact.name}</a>`

});

text += `</div>`;

$(".search-result").html(text);

$(".search-result").show();

});

$(".search-result").show();

}

};

// CODING FOR HANDLEING THE PAYMENT INTEGRATION SUPPORT

//STEP : 50 first request to create the order.

**const** paymentstart = ()**=>** {

console.log("Payment method is working on clicking button");

//using jquery to fetch the amount sent

**let** amount=$("#payment\_field").val();

console.log(amount);

if(amount=='' || amount == null)

{

// alert("amount cis required !! ");

swal("Failed !", "amount is required", "error");

return;

}

//code..

// we will use the ajax to send the request to server to create the order.

//attach the new cdn of jquery

//STEP 52 :

$.ajax(

{

url : '/user/create\_order',

data:JSON.stringify({amount:amount , info:'order\_request'}),

contentType: 'application/json',

type:'POST',

dataType:'json',

success:**function**(response){

//invoked when success.

console.log(response);

//STEP 53 after receiving the order id from the server open the paymnent form.

if(response.status=='created')

{

//open payment form using the Razor Pay web interation code.

//(step 53) Need to attach the razor pay script tag in base.html before the actual js of script tag

**let** options={

key:'rzp\_test\_8TFsUYjZM9VXJ5',

amount:response.amount,

currency:'INR',

name:'PhoneBook Cloud ',

description:'Donation',

image:'https://media-exp2.licdn.com/dms/image/C4E03AQF5z\_HzMNfnfw/profile-displayphoto-shrink\_100\_100/0/1632166608135?e=1663804800&v=beta&t=KrWb83xpwSjkjIJztd79yuxnQM2Nja6Vfdbnp5xIXxQ',

order\_id:response.id,

handler:**function**(response)

{

console.log(response.razorpay\_payment\_id);

console.log(response.razorpay\_order\_id);

console.log(response.razorpay\_signature);

console.log('payment successfull');

//alert("congrats !! PAYMENT DONE ");

//STEP 56 : update the details of payment done in our database.

//defined at last of this file

updatePaymentOnServer(response.razorpay\_payment\_id,

response.razorpay\_order\_id,

"paid"

);

//This success messge is transferred to execute after saving the data in the database

// swal("Good job!", "congrats !! PAYMENT DONE", "success");

},

prefill: {

name: "",

email: "",

contact: ""

},

notes: {

address: "Shivam Java Programmar",

},

theme: {

color: "#3399cc"

},

};

// After the Showing from , Initiate the Payment.

//STEP 54

**let** rzp = new Razorpay(options);

//copid the code form web integration . for payment field.

//

rzp.on('payment.failed', **function** (response){

console.log(response.error.code);

console.log(response.error.description);

console.log(response.error.source);

console.log(response.error.step);

console.log(response.error.reason);

console.log(response.error.metadata.order\_id);

console.log(response.error.metadata.payment\_id);

//alert("Opps !! payment failed");

swal("Failed !", "Opps !! payment failed", "error");

});

//no error then success

rzp.open();

}

},

error:**function**(error){

//invoked when error.

console.log(error);

// alert("something is alot wrong !! ");

swal("Failed !", "something is alot wrong !!", "error");

},

});

};

//Function to save the status on database.

**function** updatePaymentOnServer(payment\_id,order\_id,status)

{

$.ajax({

url : "/user/update\_order", //STEP 56 : make a handler in UserController

data:JSON.stringify({

payment\_id:payment\_id,

order\_id:order\_id,

status:status,

}),

contentType: "application/json",

type:"POST",

dataType:"json",

success:**function**(response){

//take the success fun from above and put it here , after saving in the database.

swal("Good job!", "congrats !! PAYMENT DONE", "success");

},

error:**function**(error){

swal("Failed !", "Payment SuccessFull but not reflected in our database , Our team will contact u soon", "error");

},

});

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

TEMPLATES/normal FOLDER files

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**23. add\_contact\_form.html**

<!doctype html>

<html lang=*"en"* xmlns:th=*"https://www.thymeleaf.org/"*

th:replace=*"normal/base::layout(~{::section})"*>

<head>

<meta charset=*"UTF-8"* />

<title>this is User page</title>

<body>

<!-- STEPP : 22 DESIGN THE CONTACT FORM -->

<section>

<!-- <h1>Add contact form :</h1> -->

<div class=*"card mr-3 ml-3"*>

<div class=*"card-body"*>

<h1 class=*"text-center"*>Add Contact</h1>

<div class=*"container-fluid mt-1"*>

<div class=*"row"*>

<div class=*"col-md-8 offset-md-2"*>

<form

th:action=*"@{/user/process-contact}"*

action=*"#"*

th:object=*"${contact}"*

enctype=*"multipart/form-data"*

method=*"POST"*

class=*"mt-2"*

>

<!-- First Name field -->

<div class=*"input-group"*>

<div class=*"input-group-prepend"*>

<div class=*"input-group-text"*>

<i class=*"fa fa-plus"*></i>

</div>

</div>

<input required type=*"text"* id=*"firstName"* name=*"name"* placeholder=*"Enter name here"* class=*"form-control"* />

</div>

<!-- Second Name field -->

<div class=*"input-group mt-3"*>

<div class=*"input-group-prepend"*>

<div class=*"input-group-text"*>

<i class=*"fa fa-user-alt"*></i>

</div>

</div>

<input required type=*"text"* id=*"secondName"* name=*"secondName"* placeholder=*"Enter nick name"* class=*"form-control"* />

</div>

<!--Phone field -->

<div class=*"input-group mt-3"*>

<div class=*"input-group-prepend"*>

<div class=*"input-group-text"*>

<i class=*"fa fa-phone"*></i>

</div>

</div>

<input required type=*"number"* id=*"phone"* name=*"phone"* placeholder=*"Enter Phone"* class=*"form-control"* />

</div>

<!--Email Field field -->

<div class=*"input-group mt-3"*>

<div class=*"input-group-prepend"*>

<div class=*"input-group-text"*>

@

</div>

</div>

<input type=*"email"* id=*"email"* name=*"email"* placeholder=*"Enter Email Address"* class=*"form-control"* />

</div>

<!--Work Field field -->

<div class=*"input-group mt-3"*>

<div class=*"input-group-prepend"*>

<div class=*"input-group-text"*>

<i class=*"fa fa-briefcase"*></i>

</div>

</div>

<input type=*"text"* id=*"work"* name=*"work"* placeholder=*"Enter Work"* class=*"form-control"* />

</div>

<!--Contact Description field -->

<div class=*"from-group mt-3"*>

<textarea name=*"description"* id=*"mytextarea"* placeholder=*"Enter Your description"* rows=*"10"* class=*"form-control"*></textarea>

</div>

<!-- Image Field -->

<div class=*"custom-file"*>

<input type=*"file"* name=*"profileImage"* />

</div>

<div class=*"container text-center"*>

<button class=*"btn btn-outline-primary"*> Submit</button>

</div>

</form>

</div>

</div>

</div>

</div>

</div>

<!-- Script to Tiny Rich Editor -->

<script src=*"https://cdn.tiny.cloud/1/your-tinymce-apikey/tinymce/6/tinymce.min.js"* referrerpolicy=*"origin"*></script>

<script>

tinymce.init({

selector: "#mytextarea"

});

</script>

<!-- Adding the side bar tag as highlighted -->

<script>

$(document).ready(

()=> {

$(".item").removeClass("active");

$("#add-link").addClass("active");

}

);

</script>

</section>

</body>

</html>

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**24. base.html**

<!doctype html>

<html lang=*"en"* xmlns:th=*"https://www.thymeleaf.org/"* th:fragment=*"layout(content)"*>

<head>

<!-- Required meta tags -->

<meta charset=*"utf-8"*>

<meta name=*"viewport"* content=*"width=device-width, initial-scale=1, shrink-to-fit=no"*>

<!-- Bootstrap CSS -->

<link rel=*"stylesheet"* href=*"https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.min.css"* integrity=*"sha384-TX8t27EcRE3e/ihU7zmQxVncDAy5uIKz4rEkgIXeMed4M0jlfIDPvg6uqKI2xXr2"* crossorigin=*"anonymous"*>

<link rel=*"stylesheet"* th:href=*"@{/css/style.css}"* />

<!-- Link copied from the Icons (home etc)cdnjs website. -->

<link rel=*"stylesheet"* href=*"https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.1.1/css/all.min.css"* integrity=*"sha512-KfkfwYDsLkIlwQp6LFnl8zNdLGxu9YAA1QvwINks4PhcElQSvqcyVLLD9aMhXd13uQjoXtEKNosOWaZqXgel0g=="* crossorigin=*"anonymous"* referrerpolicy=*"no-referrer"* />

<title th:text=*${title}*>Base html</title>

<!-- STEP 41` MAKING the tags of side bar visible -->

<!-- Include the jquery to make the sidebar tags highligthted dynamically. -->

<!--

Remvoe this old one and add new full version of jquery (STEP 51 )

Just google the jquery cdn and attach the manified version here , min contains the functionality of the " ajax ""

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj" crossorigin="anonymous"></script>

-->

<script

src=*"https://code.jquery.com/jquery-3.6.0.min.js"*

integrity=*"sha256-/xUj+3OJU5yExlq6GSYGSHk7tPXikynS7ogEvDej/m4="*

crossorigin=*"anonymous"*></script>

</head>

<body>

<!--START NAVBAR -->

<nav class=*"fixed-top navbar navbar-expand-lg navbar-dark bg-primary"*>

<a class=*"navbar-brand"* href=*"#"*>Call Management App</a>

<button class=*"navbar-toggler"* type=*"button"* data-toggle=*"collapse"* data-target=*"#navbarSupportedContent"* aria-controls=*"navbarSupportedContent"* aria-expanded=*"false"* aria-label=*"Toggle navigation"*>

<span class=*"navbar-toggler-icon"*></span>

</button>

<div class=*"collapse navbar-collapse"* id=*"navbarSupportedContent"*>

<ul class=*"navbar-nav ml-auto"*>

<li class=*"nav-item active"*>

<a class=*"nav-link"* th:href=*"@{/}"* href=*"#"*>Home <span class=*"sr-only"*>(current)</span></a>

</li>

<li class=*"nav-item"*>

<a class=*"nav-link"* href=*"#"*>About</a>

</li>

<li class=*"nav-item"*><a class=*"nav-link"* href=*"#"*

tabindex=*"-1"* aria-disabled=*"true"*><span th:text=*"${user.name}"*></span></a></li>

<li class=*"nav-item"*><a class=*"nav-link"* th:href=*"@{/logout}"* href=*"#"*

tabindex=*"-1"* aria-disabled=*"true"*>Logout</a></li>

</ul>

</div>

</nav>

<!-- END OF NAVBAR -->

<!-- <h3>The base page will always run -> followed with other ---- This is copied from bootstrap!</h3>

-->

<!-- SIDE BAR ------------- of Normal User -->

<div class=*"sidebar mt-3"*>

<span onclick="toggleSidebar()" class=*"crossBtn"*>&times;</span>

<a id=*"home-link"* href=*"#"* th:href=*"@{/user/index}"* class=*"item "*><i class=*"fa-solid fa-house-user"*></i> Home</a>

<a id= *"contact-link"* href=*"#"* th:href=*"@{/user/show-contacts/0}"*class=*"item "*><i class=*"fas fa-id-card-alt"*> </i> View Contacts</a>

<a id= *"add-link"* href=*"#"* th:href=*"@{/user/add-contact}"* class=*"item"*><i class=*"fas fa-pencil-alt"*></i> Add Contact</a>

<a id= *"profile-link"* href=*"#"* th:href=*"@{/user/profile}"* class=*"item"*><i class=*"fas fa-user-circle"*></i> Your Profile</a>

<a id= *"settings-link"* th:href=*"@{/user/settings}"* class=*"item"* ><i class=*"fas fa-cogs"*> </i> Settings</a>

<!-- <a href="#" th:href="@{/logout}" class="item"> <i class="fas fa-sign-out-alt"> </i>Logout</a>

-->

<a href=*"#"* th:onclick=*"|putout()|"* class=*"fas fa-sign-out-alt"*>Logout</a>

<!-- <a href="#" th:onclick="|deleteContact(${c.cid} )|" class="btn btn-danger btn-sm">Delete</a>

-->

<div class=*"divider"*></div>

</div>

<!-- End of SIDE BAR -->

<!-- content -->

<div class=*"content mt-5"*>

<i onclick="toggleSidebar()" class=*"fas fa-bars m-3"*></i> <!-- Used to give 3 small lines to open and close the content -->

<!-- Show the Success Msg before the COntent : and useing session to disappear it afer view once. -->

<!-- Message Box -->

<div th:if=*"${session.message}"* class=*"container"*>

<div th:classappend=*"${'alert-'+session.message.type}"* class=*"alert text-center"* role=*"alert"*>

<span th:text=*"${session.message.content}"*></span>

</div>

<th:block th:text=*"${#session.removeAttribute('message')}"*></th:block> <!--USed to remove msg after once shown -->

</div>

<div th:replace=*"${content}"*></div>

</div>

<!-- End of Content -->

<!-- Optional JavaScript; choose one of the two! -->

<!-- Option 1: jQuery and Bootstrap Bundle (includes Popper)

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj" crossorigin="anonymous"></script>

-->

<script src=*"https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/js/bootstrap.bundle.min.js"* integrity=*"sha384-ho+j7jyWK8fNQe+A12Hb8AhRq26LrZ/JpcUGGOn+Y7RsweNrtN/tE3MoK7ZeZDyx"* crossorigin=*"anonymous"*></script>

<!--Included this for sweet alert code while deleteing -->

<script src=*"https://unpkg.com/sweetalert/dist/sweetalert.min.js"*></script>

<!--

<script>

function deleteContact(cid)

{

swal({

title: "Are you sure?",

text: "You want to delete this contact!",

icon: "warning",

buttons: true,

dangerMode: true,

})

.then((willDelete) => {

if (willDelete) {

window.location="/user/delete/"+cid;

});

} else {

swal("Contact is safe!");

}

});

}

</script>

-->

<script th:src=*"@{/js/script.js}"*></script>

<script>

**function** deleteContact(cid)

{

swal({

title: "Are you sure?",

text: "Are u sure to delete Contact!",

icon: "warning",

buttons: **true**,

dangerMode: **true**,

})

.then((willDelete) => {

**if** (willDelete) {

window.location="/user/delete/"+cid;

} **else** {

swal("Your Contact is safe!");

}

});

}

</script>

<!-- Included this for swweet alert dialogs -->

<script src=*"https://unpkg.com/sweetalert/dist/sweetalert.min.js"*></script>

<!-- STEP: 53 Applied the razor pay script , before the js file -->

<script src=*"https://checkout.razorpay.com/v1/checkout.js"*></script>

<script th:src=*"@{/js/script.js}"*></script>

<script>

**function** putout(cid)

{

swal({

title: "Are you sure?",

text: " Logout Session !",

icon: "warning",

buttons: **true**,

dangerMode: **true**,

})

.then((willDelete) => {

**if** (willDelete) {

window.location="/logout";

} **else** {

swal("Session Continues .. !");

}

});

}

</script>

</body>

</html>

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**25. contact\_detail.html**

<!doctype html>

<html lang=*"en"* xmlns:th=*"https://www.thymeleaf.org/"*

th:replace=*"normal/base::layout(~{::section})"*>

<head>

<meta charset=*"UTF-8"* />

<title>this is User page</title>

<body>

<!-- STEP 31 -->

<section>

<div class=*"card mr-3 ml-3 "* th:if=*${contact}*>

<div class=*"card-body"* > <!-- The div will only we show when use have a valid contact corresponds to the user -->

<div class=*"container text-center"*>

<img src=*""* style="height:*200px*; width:*200px*;" th:src=*"@{'/img/' +${contact.image}}"* class=*"my\_profile\_picture"* alt=*"profile\_photo "* />

<h1 class=*"mt-3"*> <span th:text=*"${contact.name}"*></span> ( <span th:text=*"${contact.secondName}"*></span> ) </h2>

<table class=*"table"*>

<thead>

<tr>

<th scope=*"col"*>#ID</th>

<th scope=*"col"*>SCM\_USER<span th:text=*"${contact.cid}"*></span></th>

</tr>

</thead>

<tbody>

<tr>

<th scope=*"row"*></span>EMail</th>

<td><span th:text=*"${contact.email}"*></td>

</tr>

<tr>

<th scope=*"row"*></span>Work</th>

<td><span th:text=*"${contact.work}"*></td>

</tr>

</tbody>

</table>

<div class=*"description mt-4"*>

<th:block th:utext=*"${contact.description}"*></th:block> <!-- th:utext is used to resolve the html provide in description section directly copied form websites. -->

</div>

<div class=*"container mt-4"*>

<!--Delete From button -->

<a href=*"#"* th:onclick=*"|deleteContact(${contact.cid} )|"* class=*"btn btn-danger btn-block"*>Delete</a>

<!-- Update Form Button -->

<form method=*"post"* th:action=*"@{'/user/update-contact/'+ ${contact.cid}}"* class=*"mt-2"*>

<button type=*"submit"* class=*"btn btn-primary btn-block"*>Update</button>

</form>

</div>

</div>

</div>

</div>

<!--To show the unauthorised contact access -->

<h1 class=*"text-center p-3 mr-4 bg-danger text-white"* th:unless=*"${contact}"*>You don't have permission to View this Contact...</h1>

<!-- STEP 40 :a , making the sidebar tags as visible, target the common class "item" -->

</section>

</body>

</html>

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**26 . profile.html**

<!doctype html>

<html lang=*"en"* xmlns:th=*"https://www.thymeleaf.org/"*

th:replace=*"normal/base::layout(~{::section})"*>

<head>

<meta charset=*"UTF-8"* />

<title>this is User page</title>

<body>

<section>

<div class=*"container"*>

<div class=*"card"*>

<div class=*"card-body"*>

<div class=*"container text-center mt-3"*>

<img src=*""* style="height:*200px*; width:*200px*;" th:src=*"@{'/img/' +${user.imageUrl}}"* class=*"my\_profile\_picture"* alt=*"profile\_photo "* />

</div>

<h1 class=*"text-center mt-3"* th:text=*"${user.name}"*></h1>

<div class=*"table-responsive"*>

<table class=*"table table-hover"* >

<thead class=*"thead-dark"*>

<tr>

<th scope=*"col"*>#USER\_ID</th>

<th scope=*"col"*>SCM\_USER<span th:text=*"${user.id}"*></span></th>

</tr>

</thead>

<tbody>

<tr>

<th scope=*"row"*></span>Role</th>

<td><span th:text=*"${user.role}"*></td>

</tr>

<tr>

<th scope=*"row"*></span>Account Active</th>

<td><span th:text=*"${user.enabled}"*></td>

</tr>

<tr>

<th scope=*"row"*></span>About</th>

<td><span th:text=*"${user.about}"*></td>

</tr>

</tbody>

</table>

</div>

</div>

</div>

</div>

<!-- Adding the side bar tag as highlighted -->

<script>

$(document).ready(

()=> {

$(".item").removeClass("active");

$("#profile-link").addClass("active");

}

);

</script>

</section>

</body>

</html>

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**27 . settings.html**

<!doctype html>

<html lang=*"en"* xmlns:th=*"https://www.thymeleaf.org/"*

th:replace=*"normal/base::layout(~{::section})"*>

<head>

<meta charset=*"UTF-8"* />

<title>this is Settings page</title>

<body>

<section>

<div class=*"container"*>

<div class=*"row"*>

<div class=*"col-md-12"*>

<div class=*"card"*>

<div class=*"card-body"*>

<h2 class=*"text-center my-3"*>Change Password</h2>

<form action=*""* method=*"POST"* th:action=*"@{/user/change-password}"*>

<div class=*"form-group"*>

<input type=*"text"* name=*"oldPassword"* class=*"form-control"* placeholder=*"Enter the Old Password "* />

</div>

<div class=*"form-group"*>

<input type=*"password"* name=*"newPassword"* class=*"form-control"* placeholder=*"Enter the New Password "* />

</div>

<div class=*"container text-center"*>

<button type=*"submit"* class=*"btn btn-success"*>Submit</button>

<button type=*"reset"* class=*"btn btn-warning"*>Reset</button>

</div>

</form>

</div>

</div>

</div>

</div>

</div>

<!-- STEP 40 :a , making the sidebar tags as visible, target the common class "item" -->

<script>

$(document).ready(

()=> {

$(".item").removeClass("active");

$("#settings-link").addClass("active");

}

);

</script>

</section>

</body>

</html>

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**28 .show\_contacts.html**

<!doctype html>

<html lang=*"en"* xmlns:th=*"https://www.thymeleaf.org/"*

th:replace=*"normal/base::layout(~{::section})"*>

<head>

<meta charset=*"UTF-8"* />

<title>this is User-Contacts page</title>

<body>

<section>

<div class=*"card mr-3 ml-3"*>

<div class=*"card-body"*>

<h1 class=*"text-center"*>Your Contacts</h1>

<!-- Search Box -->

<div class=*"search-container my-4"*>

<input onkeyup="search()" id=*"search-input"* type=*"text"* class=*"form-control"* placeholder=*"Search Your Contacts"* />

<div class=*"search-result"*>

<!-- Dynamicly mactching contact wll be shown

-->

<!-- Show the reuslts -->

</div>

</div>

<div class=*"table-responsive"*>

<table class=*"table table-hover"*>

<thead class=*"thead-dark"*>

<tr>

<th scope=*"col"*>id</th>

<th scope=*"col"*>Name</th>

<th scope=*"col"*>Email</th>

<th scope=*"col"*>Phone</th>

<th scope=*"col"*>Action</th>

</tr>

</thead>

<tbody>

<tr th:each=*"c: ${contacts}"*>

<th scope=*"row"* th:text=*"${'SCM\_USER'+c.cid}"*>1</th></td>

<td >

<img src=*""* th:src=*"@{'/img/'+${c.image}}"* class=*"my\_profile\_picture"* alt=*"profile\_photo "* >&emsp;</img>

<span th:text=*"${c.name}"*>Name</span>

</td>

<td >

<a th:href=*"@{'/user/' + ${c.cid} +'/contact/'}"*>

<span th:text=*"${c.email}"* >Email</span>

</a>

</td>

<td th:text=*"${c.phone}"*>Phonerandom</td>

<td>

<!-- STEPP : 33 <a th:href="@{'/user/delete/'+${c.cid}}" class="btn btn-danger btn-sm">Delete</a>-->

<a href=*"#"* th:onclick=*"|deleteContact(${c.cid} )|"* class=*"btn btn-danger btn-sm"*>Delete</a>

<!-- Update Form Button -->

<form method=*"post"* th:action=*"@{'/user/update-contact/'+ ${c.cid}}"* class=*"mt-2"*>

<button type=*"submit"* class=*"btn btn-primary btn-sm"*>Update</button>

</form>

</td>

</tr>

</tbody>

</table>

</div> <!-- End of div to make table responsive -->

<!-- Pagenation Filed STEPP : 27 -->

<nav aria-label=*"Page navigation example"*>

<ul class=*"pagination"*>

<li th:if=*"${currentPage!=0}"* class=*"page-item"*><a th:href=*"@{'/user/show-contacts/'+${currentPage-1}}"* class=*"page-link"* href=*"#"* >Previous</a></li>

<!-- To show the no of pages 1,2,3 on pagenation -->

<li th:classappend=*"${currentPage==i-1 ? 'active': ''}"* th:each=*"i:${#numbers.sequence(1,totalPages)}"* class=*"page-item"*> <a href=*"#"* th:href=*"@{'/user/show-contacts/'+${i-1}}"* class=*"page-link"*><span th:text=*"${i}"*></a></li>

<li th:if=*"${currentPage+1!=totalPages}"* class=*"page-item"*> <a th:href=*"@{'/user/show-contacts/'+${currentPage+1}}"* class=*"page-link"* href=*"#"* >Next</a></li>

</ul>

</nav>

</div>

</div>

<script>

$(document).ready(

()=> {

$(".item").removeClass("active");

$("#contact-link").addClass("active");

}

);

</script>

</section>

</body>

</html>

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**29. update\_form.html**

<!doctype html>

<html lang=*"en"* xmlns:th=*"https://www.thymeleaf.org/"*

th:replace=*"normal/base::layout(~{::section})"*>

<head>

<meta charset=*"UTF-8"* />

<title>this is User page</title>

<body>

<section>

<div class=*"card mr-3 ml-3"*>

<div class=*"card-body"*>

<h1 class=*"text-center"*>Update Contact</h1>

<div class=*"container-fluid mt-1"*>

<div class=*"row"*>

<div class=*"col-md-8 offset-md-2"*>

<form

th:action=*"@{/user/process-update}"*

action=*"#"*

th:object=*"${contact}"*

enctype=*"multipart/form-data"*

method=*"POST"*

class=*"mt-2"*

>

<!--Using the Hidden type as Id to make update process easy -->

<input type=*"hidden"* th:value=*"${contact.cid}"* name=*"cid"* />

<!-- First Name field -->

<div class=*"input-group"*>

<div class=*"input-group-prepend"*>

<div class=*"input-group-text"*>

<i class=*"fa fa-plus"*></i>

</div>

</div>

<input th:value=*"${contact.name}"* required type=*"text"* id=*"firstName"* name=*"name"* placeholder=*"Enter name here"* class=*"form-control"* />

</div>

<!-- Second Name field -->

<div class=*"input-group mt-3"*>

<div class=*"input-group-prepend"*>

<div class=*"input-group-text"*>

<i class=*"fa fa-user-alt"*></i>

</div>

</div>

<input th:value=*"${contact.secondName}"* required type=*"text"* id=*"secondName"* name=*"secondName"* placeholder=*"Enter nick name"* class=*"form-control"* />

</div>

<!--Phone field -->

<div class=*"input-group mt-3"*>

<div class=*"input-group-prepend"*>

<div class=*"input-group-text"*>

<i class=*"fa fa-phone"*></i>

</div>

</div>

<input th:value=*"${contact.phone}"* required type=*"number"* id=*"phone"* name=*"phone"* placeholder=*"Enter Phone"* class=*"form-control"* />

</div>

<!--Email Field field -->

<div class=*"input-group mt-3"*>

<div class=*"input-group-prepend"*>

<div class=*"input-group-text"*>

@

</div>

</div>

<input th:value=*"${contact.email}"* type=*"email"* id=*"email"* name=*"email"* placeholder=*"Enter Email Address"* class=*"form-control"* />

</div>

<!--Work Field field -->

<div class=*"input-group mt-3"*>

<div class=*"input-group-prepend"*>

<div class=*"input-group-text"*>

<i class=*"fa fa-briefcase"*></i>

</div>

</div>

<input th:value=*"${contact.work}"* type=*"text"* id=*"work"* name=*"work"* placeholder=*"Enter Work"* class=*"form-control"* />

</div>

<!--Contact Description field -->

<div class=*"from-group mt-3"*>

<textarea th:text=*"${contact.description}"* name=*"description"* id=*"mytextarea"* placeholder=*"Enter Your description"* rows=*"10"* class=*"form-control"*></textarea>

</div>

<!-- Image Field -->

<div class=*"container text-center mt-3"*>

<img style="width:*150px*; height:*150px*; " class=*"img-fluid my\_profile\_picture "* th:src=*"@{'/img/'+${contact.image}}"* alt=*"No image"* />

</div>

<p class=*"text-secondary"* >Select New Profile Pic</p>

<!-- -->

<div class=*"custom-file mt-3"*>

<input type=*"file"* name=*"profileImage"* />

</div>

<div class=*"container text-center"*>

<button class=*"btn btn-outline-primary"*> Submit</button>

</div>

</form>

</div>

</div>

</div>

</div>

</div>

<!-- Script to Tiny Rich Editor -->

<script src=*"https://cdn.tiny.cloud/1/your-tinymce-apikey/tinymce/6/tinymce.min.js"* referrerpolicy=*"origin"*></script>

<script>

tinymce.init({

selector: "#mytextarea"

});

</script>

</section>

</body>

</html>

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**30.user\_dashboard.html**

<!doctype html>

<html lang=*"en"* xmlns:th=*"https://www.thymeleaf.org/"*

th:replace=*"normal/base::layout(~{::section})"*>

<head>

<meta charset=*"UTF-8"* />

<title>this is User page</title>

<body>

<section>

<h4><i>Welcome</i> <span th:text=*"${person}"*></span></h4>

<h6 th:text=*"${date}"*></h6>

<div class=*"container"*>

<div class=*"card mt-5 md-2"*>

<div class=*"card-body mt-3"*>

<div class=*"container text-center mt-3"*>

<img src=*""* style="height:*200px*; width:*200px*;" th:src=*"@{'/img/contact.png'}"* class=*"my\_profile\_picture"* alt=*"profile\_photo "* />

</div>

<h1 class=*"text-center mt-3 text-secondary font-italic font-weight-light "*>

Start Adding your contacts on Cloud

</h1>

<a th:href=*"@{/user/add-contact}"* class=*"btn btn-primary mt-4 btn-block text-uppercase "*> Add NEW Contact</a>

<hr />

</div>

</div>

</div>

<!-- PAYMENT FORM OF DONATE US -->

<div class=*"container"*>

<div class=*"card mt-3 md-2"*>

<div class=*"card-body mt-3"*>

<h2 class=*"my-2 text-center"*> <b>Donate Us</b> </h2>

<input id=*"payment\_field"* type=*"text"* class=*"form-control my-2"* placeholder=*"Enter the amount "* />

<div class=*"container text-center mt-3"*>

<button onclick="paymentstart()" class=*" btn btn-success btn-block"* type=*"submit"* > Pay </button>

</div>

</div>

</div>

</div>

<!-- STEP 40 :a , making the sidebar tags as visible, target the common class "item" -->

<script>

$(document).ready(

()=> {

$(".item").removeClass("active");

$("#home-link").addClass("active");

}

);

</script>

</section>

</body>

</html>

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**TEMPLATE FOLDER FILES**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**31 . about.html**

<!doctype html>

<html lang=*"en"* xmlns:th=*"https://www.thymeleaf.org/"*

th:replace=*"base::layout(~{::section})"*>

<head>

<meta charset=*"UTF-8"* />

<title>this is home page</title>

<body>

<section>

<h1>\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_</h1>

<h1>this is About page</h1>

<h2>About: Smart Cnt Manager</h2>

</section>

</body>

</html>

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**32.base.html**

<!doctype html>

<html lang=*"en"* xmlns:th=*"https://www.thymeleaf.org/"* th:fragment=*"layout(content)"*>

<head>

<!-- Required meta tags -->

<meta charset=*"utf-8"*>

<meta name=*"viewport"* content=*"width=device-width, initial-scale=1, shrink-to-fit=no"*>

<!-- Bootstrap CSS -->

<link rel=*"stylesheet"* href=*"https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.min.css"* integrity=*"sha384-TX8t27EcRE3e/ihU7zmQxVncDAy5uIKz4rEkgIXeMed4M0jlfIDPvg6uqKI2xXr2"* crossorigin=*"anonymous"*>

<link rel=*"stylesheet"* th:href=*"@{/css/style.css}"* />

<title th:text=*${title}*>phonebook</title>

</head>

<body>

<!--START NAVBAR -->

<nav class=*" navbar navbar-expand-lg navbar-dark bg-primary"*>

<a class=*"navbar-brand"* href=*"#"*><b><i>PhoneBook Cloud</i></b></a>

<button class=*"navbar-toggler"* type=*"button"* data-toggle=*"collapse"* data-target=*"#navbarSupportedContent"* aria-controls=*"navbarSupportedContent"* aria-expanded=*"false"* aria-label=*"Toggle navigation"*>

<span class=*"navbar-toggler-icon"*></span>

</button>

<div class=*"collapse navbar-collapse"* id=*"navbarSupportedContent"*>

<ul class=*"navbar-nav ml-auto"*>

<li class=*"nav-item active"*>

<a class=*"nav-link"* th:href=*"@{/}"* href=*"#"*>Home <span class=*"sr-only"*>(current)</span></a>

</li>

<!--

<li class="nav-item">

<a class="nav-link" href="#">About </a>

</li> -->

<li class=*"nav-item"*><a class=*"nav-link"* th:href=*"@{/signin}"* href=*"#"*

tabindex=*"-1"* aria-disabled=*"true"*>Login</a></li>

<li class=*"nav-item"*><a class=*"nav-link"* th:href=*"@{/signup}"* href=*"#"*

tabindex=*"-1"* aria-disabled=*"true"*>Signup</a></li>

</ul>

</div>

</nav>

<!-- END OF NAVBAR -->

<!-- <h3>The base page will always run -> followed with other ---- This is copied from bootstrap!</h3>

-->

<div th:replace=*"${content}"*></div>

<!-- Optional JavaScript; choose one of the two! -->

<!-- Option 1: jQuery and Bootstrap Bundle (includes Popper) -->

<script src=*"https://code.jquery.com/jquery-3.5.1.slim.min.js"* integrity=*"sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj"* crossorigin=*"anonymous"*></script>

<script src=*"https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/js/bootstrap.bundle.min.js"* integrity=*"sha384-ho+j7jyWK8fNQe+A12Hb8AhRq26LrZ/JpcUGGOn+Y7RsweNrtN/tE3MoK7ZeZDyx"* crossorigin=*"anonymous"*></script>

<script th:src=*"@{/js/script.js}"*></script>

</body>

</html>

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**33.change\_password.html**

<!doctype html>

<html lang=*"en"* xmlns:th=*"https://www.thymeleaf.org/"*

th:replace=*"base::layout(~{::section})"*>

<head>

<meta charset=*"UTF-8"* />

<title>this is Change page</title>

<body>

<section>

<div id=*"login"*>

<div class=*"container"*>

<div id=*"login-row"* class=*"row justify-content-center align-items-center"*>

<div id=*"login-column"* class=*"col-md-6"*>

<div id=*"login-box"* style="height:*230px* ; " class=*"col-md-12"*>

<!-- Password Change Form -->

<h1 class=*"text-center m-3"*>New Password</h1>

<form action=*""* th:action =*"@{/change-password}"* method=*"post"*>

<div class=*"form-group"*>

<input required=*"required"* type=*"password"* name=*"new\_password"* class=*"form-control"* placeholder=*"Enter new password"* />

</div>

<div class=*"container text-center"*>

<button type=*"submit"* class=*"btn btn-warning"*> Confirm</button>

</div>

</form>

</div>

</div>

</div>

</div>

</div>

</section>

</body>

</html>

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**34 .forgot\_email\_form.html**

<!doctype html>

<html lang=*"en"* xmlns:th=*"https://www.thymeleaf.org/"*

th:replace=*"base::layout(~{::section})"*>

<head>

<meta charset=*"UTF-8"* />

<title>this is Forgot page</title>

<body>

<section>

<div id=*"login"*>

<div class=*"container"*>

<div id=*"login-row"* class=*"row justify-content-center align-items-center"*>

<div id=*"login-column"* class=*"col-md-6"*>

<div id=*"login-box"* style="height:*300px* ; " class=*"col-md-12"*>

<!-- Display message if errors comes in sending the OTP -->

<!-- Message Box -->

<div th:if=*"${session.message}"* class=*"container"*>

<div class=*"alert alert-danger my-2 text-center"* role=*"alert"*>

<span th:text=*"${session.message}"*></span>

</div>

<th:block th:text=*"${#session.removeAttribute('message')}"*></th:block> <!--USed to remove msg after once shown -->

</div>

<!-- Display the error message -->

<!-- Email Reset Form -->

<h1 class=*"text-center m-3"*>Reset Password</h1>

<form action=*""* th:action =*"@{/send-otp}"* method=*"post"*>

<div class=*"form-group"*>

<input required=*"required"* type=*"email"* name=*"email"* class=*"form-control"* placeholder=*"Enter Your email"* />

</div>

<div class=*"container text-center"*>

<button type=*"submit"* class=*"btn btn-success"*> Get OTP</button>

</div>

</form>

</div>

</div>

</div>

</div>

</div>

</section>

</body>

</html>

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**35 . home.html**

<!doctype html>

<html lang=*"en"* xmlns:th=*"https://www.thymeleaf.org/"* th:replace=*"base::layout(~{::section})"*>

<head>

<meta charset=*"UTF-8"* />

<title>Phone Book Cloud</title>

<body>

<section class=*"banner d-flex justify-content-center align-items-center "* >

<div class=*"text-center"*>

<h2><i>Hi! Welcome </i></h2>

<h1>PhoneBook Cloud</h1>

<br>

<p >

<h5 class=*"a"* >An efficient solution of all your requirements , Want to access your contacts and get in touch with your closed ones , just be free and assign the task to us.

</h5>

</p>

<a href=*"/signup"* class=*" mt-4 btn bg-primary btn-lg text-white"*>Lets Start</a>

</div>

</section>

</body>

</html>

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**36 . login.html**

<!doctype html>

<html lang=*"en"* xmlns:th=*"https://www.thymeleaf.org/"*

th:replace=*"base::layout(~{::section})"*>

<head>

<meta charset=*"UTF-8"* />

<title>this is home page</title>

<body>

<section>

<div id=*"login"*>

<div class=*"container"*>

<div id=*"login-row"* class=*"row justify-content-center align-items-center"*>

<div id=*"login-column"* class=*"col-md-6"*>

<div id=*"login-box"* class=*"col-md-12"*>

<form id=*"login-form"* class=*"form"* th:action=*"@{/dologin}"* method=*"post"*>

<h3 class=*"text-center text-info"*>Login</h3>

<div th:if=*"${param.error}"* class=*"alert alert-danger"*>

Invalid UserName and Password

</div>

<div th:if=*"${param.logout}"* class=*"alert alert-success"*>

You have been Successfully Logout

</div>

<div th:if=*"${param.change}"* class=*"alert alert-success"*>

<p th:text=*"${param.change}"*></p>

</div>

<div class=*"form-group"*>

<label for=*"username"* class=*"text-info"*>Username:</label><br>

<input type=*"text"* name=*"username"* id=*"username"* class=*"form-control"*>

</div>

<div class=*"form-group"*>

<label for=*"password"* class=*"text-info"*>Password:</label><br>

<input type=*"password"* name=*"password"* id=*"password"* class=*"form-control"*>

</div>

<div class=*"form-group"*>

<label for=*"remember-me"* class=*"text-info"*><span>Remember me</span> <span><input id=*"remember-me"* name=*"remember-me"* type=*"checkbox"*></span></label><br>

<div class=*"text-center"*>

<input type=*"submit"* name=*"submit"* class=*"btn btn-info btn-md "* value=*"submit"*>

</div>

</div>

<div id=*"register-link"* class=*"text-right"*>

<a href=*"/signup"* class=*"text-info"*>Register here</a>

</div>

<div class=*"container my-5 text-center"*>

<a th:href=*"@{/forgot}"* ><u>Forgot Password </u> </a>

</div>

</form>

</div>

</div>

</div>

</div>

</div>

</section>

</body>

</html>

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**37. signup.html**

<!doctype html>

<html lang=*"en"* xmlns:th=*"https://www.thymeleaf.org/"*

th:replace=*"base::layout(~{::section})"*>

<head>

<meta charset=*"UTF-8"* />

<title>this is Sign up page</title>

<body>

<section>

<div class=*"container"*>

<div class=*"row"*>

<div class=*"col-md-6 offset-md-3"*>

<div class=*"my-card mt-1 "*>

<div th:if=*"${session.message}"* th:classappend=*"${session.message.type}"* class=*"alert"*>

<p class=*"text-center"* th:text=*"${session.message.content}"*>

</p>

<th:block th:text=*"${#session.removeAttribute('message')}"*></th:block>

</div>

<div class=*"contaniner text-center"*>

<img style="width: *80px*;" th:src=*"@{/img/document.png}"* alt=*""* />

</div>

<h1 class=*"text-center"*>Register Here !!</h1>

<form novalidate th:action=*"@{/do\_register}"* method=*"post"*

th:object=*"${user}"*>

<!-- Name field -->

<div class=*"form-group"*>

<label for=*"name\_field"*>Your Name</label>

<input type=*"text"*

name=*"name"* class=*"form-control "*

th:classappend=*"${#fields.hasErrors('name') ? 'is-invalid':''}"*

id=*"name\_field"*

aria-describedby=*"emailHelp"* placeholder=*"Enter Name here"* required

th:value=*"${user.name}"* />

<div class=*"invalid-feedback"* th:each=*"e : ${#fields.errors('name')}"* th:text=*"${e}"*>

</div>

</div>

<!-- email field -->

<div class=*"form-group"*>

<label for=*"email\_field"*>Your email</label> <input

name=*"email"*

type=*"email"* class=*"form-control"* id=*"email\_field"*

aria-describedby=*"emailHelp"* placeholder=*"Enter here"* required

th:value=*"${user.email}"* />

</div>

<!-- Password field -->

<div class=*"form-group"*>

<label for=*"password\_field"*>Your password</label> <input

name=*"password"*

type=*"password"* class=*"form-control"*

id=*"password\_field"* aria-describedby=*"emailHelp"*

placeholder=*"Enter here"* required />

</div>

<!--User field -->

<div class=*"form-group"*>

<textarea name=*"about"* th:text=*"${user.about}"*

placeholder=*"Enter about yourself"* rows=*"5"*

class=*"form-control"*></textarea>

</div>

<!-- terms and condition check box -->

<div class=*"form-group form-check text-center"*>

<input name=*"agreement"* type=*"checkbox"* class=*"form-check-input"*

id=*"agreement"* /> <label for=*"agreement"*>Accept Terms

and Conditions</label>

</div>

<div class=*"container text-center"*>

<button class=*"btn bg-primary text-white "*>Submit</button>

<button class=*"btn bg-warning text-white "*>Reset</button>

</div>

<!-- Redirect to the Login page -->

<div class=*"container text-center mt-2"*>

<a href=*"/signin"* class=*"text-info"*>Login here</a>

</div>

</form>

</div>

</div>

</div>

</div>

</section>

</body>

</html>

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**38. simple.html**

<!doctype html>

<html lang=*"en"*>

<head>

<!-- Required meta tags -->

<meta charset=*"utf-8"*>

<meta name=*"viewport"* content=*"width=device-width, initial-scale=1, shrink-to-fit=no"*>

<!-- Bootstrap CSS -->

<link rel=*"stylesheet"* href=*"https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.min.css"* integrity=*"sha384-TX8t27EcRE3e/ihU7zmQxVncDAy5uIKz4rEkgIXeMed4M0jlfIDPvg6uqKI2xXr2"* crossorigin=*"anonymous"*>

<title>Hello, world!</title>

</head>

<body>

<h1>This is copied from bootstrap!</h1>

<!-- Optional JavaScript; choose one of the two! -->

<!-- Option 1: jQuery and Bootstrap Bundle (includes Popper) -->

<script src=*"https://code.jquery.com/jquery-3.5.1.slim.min.js"* integrity=*"sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj"* crossorigin=*"anonymous"*></script>

<script src=*"https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/js/bootstrap.bundle.min.js"* integrity=*"sha384-ho+j7jyWK8fNQe+A12Hb8AhRq26LrZ/JpcUGGOn+Y7RsweNrtN/tE3MoK7ZeZDyx"* crossorigin=*"anonymous"*></script>

<!-- Option 2: jQuery, Popper.js, and Bootstrap JS

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj" crossorigin="anonymous"></script>

<script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js" integrity="sha384-9/reFTGAW83EW2RDu2S0VKaIzap3H66lZH81PoYlFhbGU+6BZp6G7niu735Sk7lN" crossorigin="anonymous"></script>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/js/bootstrap.min.js" integrity="sha384-w1Q4orYjBQndcko6MimVbzY0tgp4pWB4lZ7lr30WKz0vr/aWKhXdBNmNb5D92v7s" crossorigin="anonymous"></script>

-->

</body>

</html>

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**39. verify\_otp.html**

<!doctype html>

<html lang=*"en"* xmlns:th=*"https://www.thymeleaf.org/"*

th:replace=*"base::layout(~{::section})"*>

<head>

<meta charset=*"UTF-8"* />

<title>this is home page</title>

<body>

<section>

<div id=*"login"*>

<div class=*"container"*>

<div id=*"login-row"* class=*"row justify-content-center align-items-center"*>

<div id=*"login-column"* class=*"col-md-6"*>

<div id=*"login-box"* style="height:*280px*" class=*"col-md-12"*>

<div class=*"alert alert-success mt-3 text-center"* role=*"alert"*>

<h5>Pls verify the OTP sent on mail</h5>

</div>

<!-- Error message when redirected back to this page, on entering the wrong otp -->

<!-- Display message if errors comes in sending the OTP -->

<!-- Message Box -->

<div th:if=*"${session.message}"* class=*"container"*>

<div class=*"alert alert-danger my-2 text-center"* role=*"alert"*>

<span th:text=*"${session.message}"*></span>

</div>

<th:block th:text=*"${#session.removeAttribute('message')}"*></th:block> <!--USed to remove msg after once shown -->

</div>

<!-- Display the error message -->

<!-- Submit OTP Form -->

<form action=*""* th:action =*"@{/verify-otp}"* method=*"post"*>

<div class=*"form-group"*>

<input required=*"required"* type=*"number"* name=*"otp"* class=*"form-control"* placeholder=*"Enter Your OTP"* />

</div>

<div class=*"container text-center"*>

<button type=*"submit"* class=*"btn btn-warning"*> Verify OTP</button>

</div>

</form>

</div>

</div>

</div>

</div>

</div>

</section>

</body>

</html>

**END OF CODE FILES :**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

\*\*\*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\*\*

DONE WITH THE PROJECT : CLOUD PHONEBOOK

\*\*\*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\*\*

SELF NOTES :

Spring Boot Project :

Smart Contact Manager.

Tools: Spring mvc, Springboot, Jpa-Hibernate, Thymeleaf(views),Spring security , MySQL .

Modules Used:

1.User\_Management (Register, Login, Logout)

2.Contact Management (show contacts, update contacts, add contacts , search)

Shri Ganeshye Namaha:

Make Spring-starter-Project :

name:Smart Contact Manager

dependecies:

1.spring-boot-dev-tools

2.spring boot-starter-web

3.started-thymeleaf

4.starter-data-jpa

5.my-sql-connector

6.starter-security

7.validation-api

8.hibernate-validator

Main file has @SpringBootApplication= @Configuration ,@EnableAutoConfiguration , @ComponenentScan

Setup the properties in application.properties.

server.port=8282

spring.datasource.name=smartcontact

spring.datasource.url=jdbc:mysql://localhost:3306/smartcontact

spring.datasource.username=root

spring.datasource.password=root

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL55Dialect

spring.jpa.hibernate.ddl-auto=update //used for automatic making the table

-> Make the ER diagram from draw.io website.

Video 40:

Make the entities as made in ER diagram.

- Make user class.(make is @Entity ,@Id,@GeneratedValue(strategy=GenerationType.AUTO))

- Make a contact class ,

- Set mapping 1 to many

Declare the List<contact> in User class.(gen getter,setter for it)

Declare User var in Contact class.(gen getter,setter)

User-mapping:

@OneToMany(cascade=CascadeType.ALL , fetch=FetchType.LAZY)

Contact-mapping:

@ManyToOne()

Now Make a UserRepository interface and just extends the jpaRepository(User,Integer)

Now create a new User and also save it using save method.

Now when we run project, tables are created . To avoid extra creation of table for Contacts use mappedBy="user" in @OneToMany(cascade=CascadeType.ALL , fetch=FetchType.LAZY, mappedBy="user")

No need to create seperate table for contacts, manage contact using foreign key "user" only.

Now make A TestController and check whether data is submitted to database or not.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Video : 41

Make the Controllers : and design handlers methods to setup the Thymeleaf /html view to show.

Use Inherit Thymeleaf concept : desing a base.html file and put css/html/js

and import in other files.

set the xmlns: of Thymeleaf.

In Base.html

Now we take the (content)input as framgent="layout()" and in body section use the replace="${}" to dynamically put the file here.

So what happen actually , whenever any other html file is called , it call the base.html using th replace tag and put its whole content (generally stored in section tag) inside the body of base.html ;

We can set the common header , footers in base file to show in all files.

eg include the navbar to show in all folders.

\*\* We can make out own css , js , images in static folder.

Include css.file

<link rel="stylesheet" th:href="@{/css/style.css}">

Include js file

Just go to base.html file and include the <script th:src="@{/js/script.js}">

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Video 42 :

Just design the fronted of Home page :

\_\_

Video 43 :

Design the Sign up form :

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Video 44:

We send the empty user from handler of signup , and receive it on th:object="${user}" in <form > - this is used to show the old data present in user. And we also assign the old values to fields using th:value="${user.name}"

To remove the green color tick used the from -control "is valid "

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Video 45: Server Side Validation:

apply 2 dependencies like javax validation , and hibernate-validator

Then set the server side validations using @Blank , @Email @Size @AssertTrue etc

Also go to Controller, ans use @Valid in handler method argument() and collect result in Bindingresults.

Now invalid feedback is added to show the error in name input field of signup page. BAsically giving red-border on field if error comes.

Now to show this : we sholud dynamically append the class name as "is-invalid" addition to form control if erros comes.

Keep in Mind that the binding resutl1 should be placed after the Model User

Do check the checkbox to check server validation of name field

We have make off the by defalult validation of html in <form novalidate> : this is done to test the working @Blank and @Size , server side validations.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Video 46 : BCryptPasswordEncoder

Making Login System.

dependency of security , remove version :

Apply spring-starter-security

Making the UserController , to handler requests of user and normal user

Now we are securing the localhost:8282/user/index

By defalut spring-secutiy protects all url , and give the password on console.

We need to define explicity which url needs to be protected , and take manual passord.

Make the CustomUserDetails , UserDetailsServiceImp Services classes. respectively.

Make the MyConfig class , extends the WebSecurityConfigurerAdapter

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Now designing our own login from , google login form bootstrap : open bootsnip website and copy the html and paste the inside of body into section of our login.html file , also copy the css except .body {} and paste in style.css file of static/css folder.

Keep in mind the custom form desinged by us should be , have same field name as username and password , as spring-security used it in his default form.

Go to config ,and make its as permanet login page ,

put loginPage("/signin") in configure method.

Now put th:action="@{/signin}" in <form> tag

Also show errors when , invalid given .

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Video-50 :

we can mentions the method to configure the behaviour of Login page

-- loginPage()

-- loginProcessingUrl() -to set the redirect view of username and password.

-- defalutSuccesUrl() ->after successful url.

-- failureUrl()

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Video 51:

Desinging the inner page of User/index - ie design the normal/user\_dashboard

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Video 52 :

Making side bar for normal user

Shortcut to add anchor tags

a.item\*5 + tab

ctrl+alt+down key

Use media query to hide side bar on small screens , and mentions the configuration of sidebar and content;

How to attach the icons :

google:

font awsome icon 5 cdn

open cdn js.com and copy the link tag and paste below the css link in base.html

now google and star font awsome icons and search icons of home and use them in your file

Used to give the closing symbol

<span class="crossBtn">&times;</span>

: Using the js to add the functionlaity of opening and closing the sidebar using the 3 dash lines.

Trick opened the script file in vs code.Changes will be reflected in eclipse.

Add the js fuction and display the js function span and i tag as in X marks

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Video 53:

Make the from For Contact Form

:

Designing the add-contact form form the user.

\*\*\*\*\*\*\*\*\* All the url are prefixed by the user : so that the other are relative to user

eg : user/index :

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*8

Since we need to use the user in all handlers , so making a common method to assign handler to all.

\* Fixing the navbar : apply fixed-top

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Video : 54

Beckend code of Add contact From,

To check the working print the Contact object :

To remove the max Image size upload:

#if error comes in uploading file size image the use these : -1 make infinte size access.

spring.servlet.multipart.max-file-size=-1

spring.servlet.multipart.max-request-size=-1

//to save the current contact : we need to find the logged in user : using the principal ,

then find the userby username ,then save the current contact to that logged in user.

Check the data Saved to the Contact Table of Database in MSQL WorkBench

If error comes remove the toString method : Not in mine case.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Video 55 : Saving image of the Form

Save Image to a folder and only store the name in the database, but name of images should be unique

//First finding path to the static folder , then adding it.

if(file.isEmpty())

{

System.out.println("File is empty");

}

else{

//set the file to folder ,and put in contact field of image.

contact.setImage(file.getOriginalFilename());

File saveFile = new ClassPathResource("static/img").getFile();

Path path=Paths.get(saveFile.getAbsolutePath()+File.separator+file.getOriginalFilename());

Files.copy(file.getInputStream(),path,StandardCopyOption.REPLACE\_EXISTING);

System.out.println("File uploaded successfully");

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Vido 56 - 57 :

Adding the TinyMCE / text box editor

Show success Msg after addition of User

Just include the script file of Tiny Editor inside the section tag of add-contact-from and copy its code to text box field

give the textarea id to our text box:

Copy paste this cody of Tiny script tags

<script src="https://cdn.tiny.cloud/1/your-tinymce-apikey/tinymce/6/tinymce.min.js" referrerpolicy="origin"></script>

<script>

tinymce.init({

selector: "#mytextarea"

});

</script>

If want to insert the large text in text box, need to manually change limit from MSQL workbench ,

Change the type of description field to lond text

and if html data is put into then fecth it usen th:utext

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Video : 58

Showing the Contacts Saved

We are , making the seperate Repository for the Contacts .

Implemeting pagenation NEXT and PREVIOUS FUNCTIONS

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Video 59

Implemeting pagenation NEXT and PREVIOUS FUNCTIONS

Copy from bootstrap pagenation STEP 27 :

From handler / backend code send only the requeired fields.

Show dynamic page to the user Controller in showContacts handler.

Need to change the Repository method in the Contact Repository method. And implement the Page<Contact> code.

Use classappend to show teh highlighted current page.

To see the pagenation firer the user/show-contacts/0 : page no is must.

Give the 0 contact on the side bar

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Setting profile Photo

Video 60 : Security bug of modification in url can be solved by the USING THE POST method , or by video method

Addin the image before name in show contact , and modify image in css file . aslo give defalut picture if not photo given.

But problem is that when we re run the project, images are deleted from the target folder.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Adding functionality to redirect to Profile of Contact via mail link

Desing handler to handler user/"id"/contact id to show the profile

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

VIDEO 61 : Removing bug random changing the url and see other users using hit and trial method /choose the POST METHOD Or allow this modification

Chech the current log in user, using the principal ,

So that we can't access random contacts.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

VIDEO 62 : DELETE buttom activation

Applying the delete functionality : change the button to <a> tag.

Also need to make handler , to handler the delete request.

TO delete the contact , we need to unlink it from the user , because during the mapping we have included the CascadeAll in One to MAny mapping

ALso chech in database , whethere the contact is actually deleted.

\*\*\*\*\*\*\*\*\*\* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ACTUALLY DELETEING THE CONTACTS (Here only user is delinking not the contacts are delteing)

\*\*\*\*\*

Since we have applied the cascade in @OneToMany Mapping so deleteof Repository is not funcional here

TO solve this SEE VIDEO 65 :

Add orphanRemoval =true; // this menas that ,when the contact (child ) unlikns forn parent (User) its automatically removed.

Also overide the equals method, remvoe the current Contact from the user.

TO appply the sweet alert before deleting just google it , and copy the sript tag on show\_contacts and add the code

Systax to use onclick delete dynamically

th:onclick="|deleteContact(${c.cid} )|"

Same fucntionality is added to delete button inside the main profile of Contact.

Just take care of object is contact, so whiteLAbel Error is not come.

Also need to delete the image of current object from the img folder of target folder.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Video 63:

Update Option button :

Designing new from for update , learning purposes.

Need to user POst Mapping to secure manipulations form the url :

Also need to use the form tag , if used the used post mapping , <a> tag defalut uses get mapping as for delete fucntion

Just Copy the same form for the add, and provide the update form :

Also fetch the values of data fields that are sent using the Model

To show the defalut value in text area use the th:text = ${contact.description}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

// Video 64 : Updateing the form in the beckEnd , also handler the image

TO handler the problem arises on the updating is that user is not assigned , so not updated

So get the current user, and save the updated contact in it.

//Getting the old contact details and chech if new one is present or not.

// VIDEo 65 : Deleteing actually

Similaary code to delete old photo , using the path and old id .

Look in dataBase ,and target folder for the changes.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

VIdeo 66

Desing the Profile Home :

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Video : 68

Desing the Search Controllr

Adding the concepts of java script :

Make new Controller SearchController

Refer video to Understand the code for the Script file. and onclick method.

SPRNIG -SECURITY CRASH COURSE VIDEO :

Crash Course By Learn Code With Durgesh.

1.Refer the diagram show in Image How security is taken care.

Firstly there is a Filter Interface which create the Object of Authentication , then this object is passed to AuthenticationManager Interface to check its validity, but its inteface so need the other helps to check authentication, it uses the Provider Manager to check all the Authentication\_Provider implementation ,and find whether obj is supported or not.If good then authentication is validated and its property is set to true, then this is returned to Provider\_Manager .... and finally returned to Filter , which then sets the Securitycontext of object.

Now to We are setting our own security .

Time 1:00 --------------

Create the MySecurityConfig class and extends the WebSecurityConfigurerAdapter. and over ride its configure(HttpSecurity) methods.

http

.authorizeRequests()

.anyRequest()

.authenticated()

.and()

.httpBasic(); //pop-up login mechanism will be created , no logout features in this.

//username will be "user" and password generated on console.

To run it using POSTMAN : set the Authorization tab -> choose the Type: Basic Auth , and put user and password.

--------------------------------------------------

override the another configure (AuthenticationManagerBuilder auth)

auth.inMemoryAuthentication().withUser("shivam").password(this.passwordEncoder().encode("shivam")).roles("ADMIN");

auth.inMemoryAuthentication().withUser("john").password(this.passwordEncoder().encode("john")).roles("NORMAL");

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_-

// now since we are using the password encoder http .formbasic() , we need to give Bean of PasswordEncoder

@Bean

public PasswordEncoder passwordEncoder()

{

//return NoPasswordEncoder.getInstance(); //simple plain text as password

}

// If not work , just restart the project and open it in incognito mode.

---------------------------------------------------

AntMatchers - are used to allow the access directly skipping the spring security.

Make Modification in configure (httpSecurity)

http .authorizeRequests()

.antMatchers(HttpMethod.GET, "/home","/public/\*\* ").permitAll()

// we can also define here which Role user can access the which url

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ROLE -- high level overview (NORMAL user)

Authority -- read,write update permissions

Now if we want to allow only the users/ url access to ADMIN

and /public/ url access to NORMAL then we can use the ant Matchers like this.

.antMatchers("/public/\*\*").hasRole("NORMAL");

.antMatchers("/users/").hasRole("ADMIN");

// That means we can use the POSTMAN to check API , with users of different roles.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

If we dont want to use the antMatcher , we can directly set the method with accessfier

@PreAuthorize("hasRole('ADMIN')") before particular method

, also need to enable the @EnableGlobalMethodSecurity(prePostEnabled=true) in SpringSecurity class.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CSRF token (Cross site request forgery) :

its type of attack, spring saves us from this and does not allow to add new user/update them ,

so to skip this functionality we need to make csrf().disable() in the SpringSecurity class.

We need to disable it , if using the in non\_browser client eg . POSTMAN .

In place of disabling the csrf we can also pass the Header of key, value pair (copy from cookies section.) get the csrf token need to use csrf.TokenRepository( .withHttpRequestFalse())

and we can go to Cookies seciton of PostMan and can copy the values and put them in Header tab with key ="X-XSRF-TOKEN" value="...."

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

FORM based Authentication.

(refer the Diagram / screenshot )

/logout available.

/username,password are sent,

Now in SpringSecurity //.httpBasic() , use

.formLogin()

.loginPage("/signin") //self made signup , also use antMatcher to make the sign as public , .permitAll()

.loginProcessingUrl("/dologin")

.defaultSuccessUrl("/users/")

Now make the signin page in templates folder , and also inject the spring-boot-stater-thymeleaf dependency in the pom.xml

Desing your login page.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Connecting it with DATABASE (spring\_security)

just replace the inMemoryAuthentication in Configure Method()

Need to setup the configuration in application.properties file

spring.datasource.username=root

spring.datasource.password=root

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

spring.datasource.url=jdbc:mysql://localhost:3306/spring\_security

spring.jpa.hibernate.ddl-auto=update

spring.jpa.properties.dialect=org.hibernate.dialect.MYSQL8Dialect

spring.jpa.show-sql=true

\*\* Attach the spring-boot-starter-jpa (remove the version) and my-sql-connector depedencies.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

To make table of UserName in database

@Entity

class User

{

@Id

String username;

}

Now work with general Structure :

1.Make the UserRepository Interface extends jpaRepository in Repo package

2. Now in main SpringBootSecurityLearnApplication implements the CommandLineRunner, it has the fun run, which run as soon as working starts.

3. Now we will create the users in the run method ()

using the setter function

-Inject the field ROLE in User , set construcctors , setter, getters etc

\*\* Imp : while setting role , put ROLE\_USER in setter injection , other times "ROLE\_" is attacjed internally by spring.

4.Autowire the UserRepository and BCryptPassword encoder to save the values

Make the another user like this process only.

The Bean type should be of BCryptPassword encoder

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Now next Thing : Make the new class

CustomUserDetail and it implemtents the UserDetails interface , which Spring security uses to maintain the configuration of The Model.

Now override all the methods in CustomUserDetail :

We need User obj to getPassword()....etc

- make constructor of user. and set the password, username using this obj, and set all other properties as true;

-Also in GrantedAuthority Method just : make HashSet

@Override

public Collection<? extends GrantedAuthority> getAuthorities() {

HashSet<SimpleGrantedAuthority> set= new HashSet<>();

set.add(new SimpleGrantedAuthority(this.user.getRole()));

return set;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Now make new class @Service CustomUserDetailsService it implements UserDetailsService ,

and override the loadUserByUserName()

declare the obj of UserRepository

.

. public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {

//User user=this.userRepository.getById(username);

User user=this.userRepository.findByUsername(username);

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Now at last we need only to use the @Autowired CustomUserDetailService obj in main class.

and just modify the configure()

{

auth.userDetailsService(customUserDetailService).passwordEncoder(passwordEncoder());

}

Refer video of last 20 min

Now just re run your project with the spring-boot-app and try to access the localhost:8080/users/

it will redirect to the signin page, now when admin signin then successfully page is show, other case Forbidden Error.

if we try to access localhost:8080/public/login/ ->

it will redirect to signin page and when normal user credintials are given , its successfullu open the login page.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_-------

Error comed in video , because while setting the role, we need to pass the setRole("ROLE\_USER");

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_--

JWT AUTHENTICATION NOTES :

//Install the plugin in the intellij idea version .

To create the spring boot project :

@Spring Tool Suite :

But we are following the sts for the developement purpose.

:::::::::::: JWT-AUTHENTICATION CONCEPTS ::::::::::::::;;

JSON WEB TOKEN ( JWT ) :

- no concepts of cookeies, session , etc

-its is an encrypted platform to securely send the information beteen the two parties.

\* Json web token is used to authenticate users.

---------------------------------

WORKING

-----------------

\* THE basic functionality if JWT is to protect the all url , unless client access then with the Header token.

When the client first time access the url, the jwt verifies the user , and a token is assigned to it, the client(browser) save the token in its localStorage , and next time when it makes request to url , it send the Header of token along with the request.

The Jwt verifies the token, and if it matches with allowed user , then grant access to it.

The format to the token is

"Bearer 454sdfsdfs5f45sd9979794sdf"

To send it vai POSTMANT , need to be passed in HEader Tab with key as Authorization, and val as above format.

\_------------------------------

STEPS used in JWT working :

check the jwt is well formed or not.

check signature.

validate the standard claims

check the client permission.

-----

Authentication : to check the validatity , whoom they claim to be.

Authorization : giving the permission to access a specific resource.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Working of JWT

1 time when the client request the server , a token is generated. The server response with the JWT token

The client save the token in the localstorager.

When next time the client the request , its also send the token with it . So server will check the token authentication .

(Just like ID card, first time id created , next time ownward , allowed to access the resource by validating the id card. )

The encode information contains the header(algo used , type ) , payload (data) , signature .

// To implement the jwt , create the project in sts ( i prefered )

Create the jwt authentiacation project.

dependencies -----

import devtools dependency (auto reload )

import spring web .

import spring security : user , password gen : on console.

Video 72 : creating the jwt project.

TO access the API present in the website. client need to generate the token.

The is of format key and value

Authorization :

"Bearer 45645465xdcfsdf1231857asdff48"

Lets we are accessing use the postman as the client .

Go to Header Tab Section and add new key : Authorization and values as : Bearer 4546854sd54654

Now by default all the url are hidden, need to make a security configuration class , to allow public access to some url and protect others .

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

STEP : 1 Make security Config class

Annotate with @Configuration , @EnableWebSecurity

extends the WebSecurityConfigurer adapter.

Override its two configure methods with different arguments . (Polymorphism)

void configure(HttpSecurity ) --> to allow url to public and which url to be authenticated.

void configure( AuthenticationManagerBuilder auth ) --->with authentication system to use, UserDetailsService , in memory , etc : ie validation on which method.

-----

Need to implement the UserDetailsService interface in CustomUserDetails Service.

Also need to override its load by username method.

Basically when configure methods works, it loads the user with given user name;

\* we have configured the Http method and the Authentication Manager configure methods.

Now we need to create the token on the first request.

so include the jwt dependecy first.

<!-- jjwt dendency STEP :

( if using + JAVA8+ version then also add dependency of)

javax.xml.bind -->

// Now generate the first time token .

Now download the helper class of jwt . from net / github repository.

And put the downloaded class in helper package as JwtUtil class.

STEP : 6

( This class helps in checking the authentication of the token using its variousd methods )

STEP : 7

Make the Jwt Controller to generate the token on first requerst of the client.

STEP: 8

To generate the token we need to accept the user sent through POSTMAN in json format

So make a class with variable as username and password

For authentication purpose we need the 3 object , which need to be autowired.

- Authentication Manager

- CustomUserDetailsService

- JwtUtil

Now inside the try catch block , use Authenticate manager , pass obj of UserPasswordAuthenticationToken ( pass username , password )

Now after successful validation of user identity , token will be generated.

to use the generate token method of Util class ,we need UserDetails Service obj.

So get its obj from the CustomUserDetailsServer.loaduserbyusername method.

then generate token.

STEP : 9

Now we need to send this token back to client . So using the Response class to send it as json object.

(This is done , so that the cleint can use this token for future purposes) In future the client need to send their requests along with the Header of this Token.

Structure of token is

Bearer sdf454654sdf4s6d4f6s

If client want to access the protected url /api , he need to use Header tab and pass Authorisation and values as token structure .

//STEP 10 :

To test the code is working or not :

OPEn the POSTMAN :

POST - localhost:8080/token

In Body tab under json as type : pass the object details

{

"username" : "Durgesh",

"password" :"Durgesh123"

}

So we will observe the Token generated "dsfgfdg454a65454sdfsd544654654213sdf"

Now copy this token , and go to Header Tab and add new {key, value}

Authorization : Bearer dsfgfdg454a65454sdfsd544654654213sdf

This token will help the client to access the protected API .

// WRITING THE CODE FOR Validating the token send from the client side.

//STEP : 11

Now we will Add filter to check the token of client , who want to access the secured api.

Make a seperate class of JwtAuthenticationFilter and extend the OncePerRequest

override itsw mwewNow we will Add filter to check the token of client , who want to access the secured apwNow we will Add filter to check the token of client , who want to access the secured api.

// Now the token is generated : upon entering the correct username , and password as the provided, in json format.

//So Doing coding to validate the token. // FILTERING

//overide the method in class and

@Override

protected void doFilterInternal(HttpServletRequest request, HttpServletResponse response, FilterChain filterChain)

throws ServletException, IOException {

// and get the requird header from it. and perform all the validation in it.

//STEP 12:

//Now we to use the filter method in Http configure method in SecurityConfig class , we need to define the usernamepasswordauthenticationtoken.

Now we can perfom the checks of accessing the protected api using the token generated .

This way THE AWT concepts helps to keeps api safe and trankit information in the coded form (token) .

And alowing access to the validated users only.

STEP : 1 -- To generate the token on POSTMAN

POST localhost:8080/token

Body Tab : raw -> JSON

{

"username":"Durgesh",

"password":"Durgesh123"

}

STEP 2: to access the api with token v

-copy the token generated above step.

-Put it in HeaderTab new key Authorization with format of value as

key : Authorization

value: Bearer 5645454sadfas5d4fa54sdf65a4

Make request

POST localhost:8080/secondtest

//Hence Proved Success .

//Now adding the proper unauthorised response.

Need to create the class, which extend AuthenticationEntry points.

Just override the method

and use it in the Configure method :

.exceptionHandling().authenticationEntryPoint(entryPoint);

;

///////////////////////////////////////////////////////////////////////////////////////////////////////////

VIDEO 74 : Now using the JWT authentication :

WITH FRONTEND ANGULAR (login app )

--> Go to angular folder under the E drive, springboot , learning of springcore , angular

--> As angular is already installed , so create new project .

--> ng new loginapp --> to create the new project.

(choose routing , CSS )

--> go inside the project folder and to run type

ng serve

Now to enable the angular materail in our project :

ng add @angular/material

Say yes to all option , choose theme etc

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Now can google angular matrial website and can use the buttons. toogleside bar etc.

Now what ever u want to use add its import and refer the email.gui project for more reference , and watch durgesh video.

-- If errors comes in writing the componets inside the router [] , in module.ts file .

Then import the required component at the top.

Now we can add the <a> tags in the button, and also user routerLink to jump to related url.

To shift the Login to right side we have added a class in css and flex 1 1 auto;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Now use Bootstrap grid system to centre items.

and open github website , see demo and NPM tag and then copy the cdn code from here.

and include the link in index.html file

Now include the code in homecomponent.html

To use the phots save them in asset folder. open it in file explorer and pasete them.

After desingning the form :

Need to do coding in the typescript file of the form.ts

To get the data in the field we user property and event binding together .using ngModel in the required input field.

[] - property binding

() - event binding

[(ngModel)]="credentials.username" used for two way event binding.

Before binding the field , give the name="username" to the fields .also add requireed to the field (so cant be submitted empty)

STEP : 12 :

//generate the token for the client

//START THE SPRING boot project of JWT authentication

//STEP : 12 : with respect to our JWt Project : we need to fire http:localhost:8080/token

POST METHOD

localhost:8080/token

We need to send the json data

{

"username": "sdf",

password:"123454"

}

//STEP 13 :

So in angular we need to create the service ,

create the services folder in src/app

and then genareate the service

ng g service loginservice

Now do coding in the ts file of login service.

TO call the server we are using the http client by importing it.

Provide the var of HttpClient in constructor .

: WE design the generateToken() method , to generate the request to create token.

Now from our login page , we call this login , method to generate the token.

STEP : 14 Go to Springboot project and add the @CrossOrigin in the mail Jwt Controller

To run this project from the angular.

//STEP :" 15 " Check the working with Form : Since we had already connected out application with the database, so we are using the MQSQL client to check the username ,and password,

//Upon submission , a token is generated on console.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Now after saving the token , in the localstorage , we can redirct to other page.

//But we need to keep user on dashboard page till it has the valid token , So use the auth gaurd

in services folder , genearte the guard

ng g guard auth

It has the method to allow visiblity of pages

First declare the obj LoginService in constuctor of auth guard.

Now apply this auth guard , in app.module.js

as canActivate:[AuthGuard]

now in guard.js file , include the obj of Router in cnstr

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

TO test the application :

\* Run spring boot application :

\* start the angulare project -loginapp by command of vs code terminal .

ng serve

Now u can login it , and admin panel will be accessed , if user is valid in database.

Click the logout button to remove out from the app.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Now we want to check whether the valid person is able to check the protected url .

So creating the another service.

ng g service user

After creating the accessing elements in Login service.

when we try to access the getUsers url it will be not allowed , so need to do some modification in spring boot code

In Controller class, containing the this mnethod we need to put @Crossorigin(origin \*)

Also need to Make seperate functionmality to send the header with the token to allow access.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Now we will make a interceptor : which every time add the header authorization in it.

Make a new file :::: auth.interceptor.ts file in services folder.

Add coding of interceptor to add the header to each request.

Now app.module.ts ---

in providers:[] , add the privade:HTTP\_INTERCEPTOR,

////////////////////////////////

VIDEO 75 : CONNECTING THE DATABASE with JWT authentication project.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

((Done coversion from fake service to database service beforer video 74 : Making the Angular Login page.))

We will made model classes :

@Entity

User -- all field username, pass, role etc

Also make a CustomUserDetails class, which interacts will User and return the various method of User.

- also declare the User object in this class,

and make Constructor of the class.

and in other methods return the user.pass etc respective fields.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Now come to CustomUserDetailsSErvice Class and comment out the old code for fake User .

Create the Dao repository to get user from the database.

-provide the custom finder method to getUser

Now come back to CustomUserDetailsService

-autowire the userRepo

and fetch the obj from the database and return the CustomUserDetails obj.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

After this re run the springboot app and see the MYSQL workbench the table will be created .

Now we can add data from the registratioon form (like done in Phonebook project) .

or for leaning purpose,

In main class of this project , @SpringBootApplication

we create the User , for demo purpose.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Now to test this we can use the POST MAN

Look the MYSQL client for the purpose getting the username and password.

Now make request to generate Token :

Copy that token to generate the protected Url

19/01/2021

06/01/2022

Stmt : 04/02/2022

REVISION JWT :

Structure :

1 com.jwt -base class , used to fakey inject user to database using the command line runner

com.jwt.config

-MySecurityConfig (configure(http) , configure(auth))

-JwtAuthenticationFilter (dofilter internal)

-JwtAuthenticationEntryPoint (error sending)

com.jwt.controller

-HomeController (welcome , getuser)

-JwtController ( generateToken , loaduserByUsername,authenticate )

com.jwt.helper

-JwtUtil (downloaded from net)

com.jwt.model

-CustomUserDetails(to pass userdefind user to check auth)

-JwtRequest (username,pass)

-JwtResponse (token send as obj)

-User (store entity in DB)

com.jwt.repo

-UserRepository(finduser from DB)

com.jwt.service

-CustomUserDetailsService(service to fetch and compare user)

Deploy z fake list book

@ReqestMapping("/api/v1")

open cmd in our project directory.

chaged java version to 1.8

(my-feature)$ git checkout master

Switched to branch 'master'

(master)$

------------------------------------------------------

DURGESH GIT IN 1 VIDEO ::::::::::::::::::::

NOTES Gits :

$ git --version

$ git config --global user.name "Shivam"

$ git config --global user.email "shivamraiias@gmail.com"

Create a project folder

To open vs code from direclty cmd in current project.

code vs .

$ git init

$ git status

$ git add index.html

$ git commit -m "first commit"

\*\*\*\* git add -A : to add all files

$ git checkout about.html : to remove / undo the changes befor the previous commit.

//To see the changes made to modified file (not added to git, nor commited )

use

git diff

--to see changes after adding to git use :

git diff --staged

--to see changes after commmit done

git diff HEAd

----- author who is doing changes

git show

--------to show previous commits

git log

-------previous two changes

git log -p -2

---------------------------------------------

To delete file permenetly

git rm -f service.html

-- only remove it staging area(make it untack)

git rm --cached service.html

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

BRANCHING

to see all branches present.

$ git branch --list

-to crate a new branch

git branch login

-to move to another brach

git checkout login

- to delete a subbranch , without its merged

git branch -D login2

--to merge sub branch into main

Go to parent branch

$ git merge login

--to delete sub brach after merging the data

$ git branch -d login

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

New method to create branch and swtich it

git checkout -b register

Search from net command to dirclty add and commit

----------------------------------------------

Learning to Work on Github from the git

- Go to the github account , create a new repository.

- Copy the link and open the GitBash from the folder , of the project.

-git init

-Now add the files to remote github using

remote add origin https://github.com/Shivamrai714/learninggit.git

-- to check current active url

git remove -v

--Come to the main branch master and rename it to main (acc to the repository).

branch -M "main"

To load all the code to github

---git push -u origin "main"

Now click on the repository , you can see the files uploaded there.

// Now lets try to add new brach

git branch register

// Change branch

git checkout register

// Do coding changes to the file fomr the vs code.

//add the files

git add -A

//commit the changes

git commit -m "updating data"

//now again push the whole brach "register" to github repo.

$ git push -u origin register

NOW WE CAN SWITCH THis branches on github

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Pull the request from GitHub WEbsite

From GitHub repository , copy the link of Repo

Create a new folder to extract the files.

Go to that project folder.

git clone https://github.com/Shivamrai714/learninggit.git

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

AFTER LEARNING THE GIT COMMAND , I WAS SUCCESSFULL IN DEPLOYIG MY APPLICATION TO THE HEREUKU

- TO CORRECTLY DEPLOY THE PROJECT

1. delete the old folder of the git from the project directory.

2. Change java version of Heruku to 1.8

3. Create a App on the Heruku and follow the instruction for the deploy.

4.Go to project explorer from the currrent sts project., then inside the project,

5.Now open the cmd

//FOllOW STEP from deploy Tab :

a) heroku login -i (login from cmd , provide id shivamraiias@gmail.com,passShivamrai714@)

b) $ heroku git:clone -a springbootdeploytesting

c)$ cd springbootdeploytesting

d)$ git add .

e)$ git commit -am "make it better"

f).$ git push heroku master

To accewss your project go to Setting Tab bottom link , to run your project.

Click link

Now append the api/v1/books to see the results.

Use postman with complete link to add the products etc on the server.

(ALL WORKED FINE ............)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

NOW learning to upload the springboot with the database on heruku .....

Open the project of fakelist of Mobiles ;

Need to attach posgres dep in the pom.xml

Need to update the application.properties file

Open Pg (Postgres) software

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Deploying the springboot mobile with the database on postgres

Create new Project on the Heruku .

Add the resource : Heruku POSTGRES

Open the created link.

Now open the setting , and change the url, database etc according to the heruku settings.

copy the host and paste them in the application.properties of the file.

--Heruku CLI is used to connect the Heruku postgrs to out localdatabase of postgres.

Now after the full process ,

open the curent folder of springboot from CMD , deploy the project on server using the previus commands .

Open the link of project form the settting of project.

Use POSTMAN to add the mobiles first then do further operation.,

If also want to save the entries in the localPostGre Db use the CLI command like this from the CMD of the current project.

But before this set the path of psql in envn var.

Check the cmd and check : psql

if it ask for pass, then path is set otherwise set the path of the bin (of POSTgres) in envm var.

otherwise it will show not recogised as internal or extenal command.

(In my case I need to set the path)

After then start new cmd .

And copy paste the CLI code of Heruku Database,

VERY IMP THING ............. I changed the port of local postgres , to 3308 , but when u are connecting the local posgres to heruku postgres the port no should be same 5432

Connecting the local posgres to heruku posgres

AFTER THis

COpy this CLI from online heruku -> resouces -> setting -> HERUKU CLI

If cmd responds with Password ,on typing the psql in cmd, then run this copied HERUKU CLI command.

(otherwise first need to set up the path.)

-------------------------------------

Now u can se the current client of online server from local cmd machine.

/dt will show the userdetails here.

Also to see the data of the online posgres from here just

select \* from mobilee

you can try this , by updating the data from the POSTMAN , and then seeing updates here

MEANS OUR HERUKU posgres is now connnected with local posgress.

Commands to see the data on the cmd.

\dt --to see all the tables

\d mobbile -- to see description of tabl