



Design, Develop, Implement & Document Community Portal Website

Module Project

Module: Web Development Foundations

Course: NICF Advanced Certificate in Software & Applications (Development & Deployment)

Start Date : 12/09/2022

End Date : 23/09/2022

Submission Date : 02/11/2022

Learner Name : Chathushi Jayarathna

Enrollment ID :

Presentation Date :

Document History

Version Number	Effective Date of release	Summary of Included Changes	Author
1	4 th March 2016	First Edition	Satya CVS
2	08 th Sep 2017	Changed for Module 5	Shrinivas K R
3	20 th Jul 2018	Changed for Module 5 RQF	Shrinivas K R

Contents

S. No.	Description
01	Development Tools
02	Business Process
03	Application Block Diagram
04	Applications
05	Models, Controllers & Classes Developed
06	UI Design
07	Includes
08	Software Development Methodology
09	Project Plan
10	Application Screen Shots
11	Project Milestones & Tasks
12	Milestone Feedback & Action Taken
13	Modifications Made On Feedback
14	Project Results
15	Proposed Improvements

1. Development Tools

LITHAN

❑ Development Tools Screen captures



Eclipse IDE

```
File Edit Source Refactor Run Window Help  
Project Explorer X Servers  
ABC-Portal (com.manysoft.abcportal) Main Webapp index.jsp - Eclipse DS  
src  
  main  
    Java Library (jivedev-1.7)  
    Maven Dependencies  
    ManySoft Resources  
    art  
      assets  
        img  
        js  
        style  
      fonts  
      icons  
      images  
      model  
      resources  
      static  
        assets  
        css  
        img  
        js  
        views  
      webapp  
        config  
        feedback.jsp  
        forget-password.jsp  
        forgot-password-confirmation.jsp  
        home.jsp  
        login.jsp  
        profile.jsp  
        register.jsp  
        search-user.jsp  
        thank-you.jsp  
        user-profile.jsp  
        view-profile.jsp  
        welcome.jsp  
        web.xml  
        index.jsp  
      web.xml  
    target  
      promax  
      project  
      Mybatis  
      Servers
```

```
index.jsp [MuController] [user-pool...] [web.xml] [ABCPortal...] [spring-servlet...] [feedback.jsp] [index.jsp] [home.jsp] [?]  
1 <%@ page language="Java" contentType="text/html; charset=ISO-8859-1" %>  
2 <%@ page encoding="ISO-8859-1" %>  
3 <%@page errorPage="errorPage.html" %>  
4 <html>  
5   <script src="https://kit.fontawesome.com/844836b72b.js" crossorigin="anonymous"></script>  
6   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0-beta/dist/css/bootstrap.min.css" href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0-beta/dist/css/bootstrap.min.css" />  
7  
8   <link href="assets/css/style.css" rel="stylesheet" />  
9   <link href="assets/css/home.css" rel="stylesheet" />  
10  <head>  
11    <meta class="noUser_noUser_expanded" lg="text-bg-light sticky-top" />  
12    <div class="container">  
13      <div class="row noUserBrand">  
14        <div class="col-12 noUserBrand fs-1" style="text-align: center;">ABC Tasks <i>(X)</i>
```



phpMyAdmin

The screenshot shows the phpMyAdmin configuration interface. At the top, there's a navigation bar with links for Databases, SQL, Status, User accounts, Export, Import, Settings, Replication, Variables, Charsets, Engines, and Plugins. The main content area is divided into several sections:

- General settings**: Includes a dropdown for "Server connection collation" set to "utf8mb4_unicode_ci".
- Database server**: Lists server details: "Server: 127.0.0.1 via TCP/IP", "Server type: MariaDB", "Server connection: SSL is not being used", "Server version: 10.4.24-MariaDB - mariadb.org binary distribution", "Protocol version: 10", "User: root@localhost", and "Server charset: UTF-8 Unicode (utf8mb4)".
- Appearance settings**: Shows language set to "English" and theme set to "pmahomme".
- Web server**: Lists PHP and MySQL information: "Apache/2.4.53 (Win64) OpenSSL/1.1.1n PHP/8.1.6", "Database client version: libmysql - mysqld 8.1.6", "PHP extension: mysqli curl mbstring", and "PHP version: 8.1.6".

At the bottom, there's a footer with "ohnMyAdmin" and links for Bookmarks, Options, History, and Clear. A status bar at the very bottom says "Press Ctrl+Enter to execute query".

1. Development Tools

LITHAN



Microsoft Word

File Tools View chathuhi.docx - Word

2. System

a. List down all the modules to be developed

- Private Module - Administrator - Manage user Data and Bulk Emailing.
- Public Module - Software Programmers

- Registration Page
- Login Page
- Forget Password Page
- User Profile Page
- Update Profile Page
- Search User Page

b. List down all the pages in each of the modules and briefly describe the purpose

- Registration Page - Using this Page visitors can register to the community portal by filling out the registration form.
- Login Page - Registered users can log in to the community portal by entering their username and password

c. Note down which are Create, Edit & View pages.

Insert - Create Page for users to register and data are inserted into database

Select - View Page for the users can view the others profile page

Update - Update Page for the user to edit and update their profile page

<you may answer this way>

a. Module	b. Pages	C. Brief description	c. Operations
Registration page	Register	Getting register data from new users	View
Login Page	Login	Provide the series where can users can login into their account	Insert
Search User Page	Search-user	To search for other users in the community portal	Search
User Profile	User-profile	To see the details of user profile	View
Update profile page	Update	update the details	Update
Forget password page	Forget-password	Reset the user password	Insert

d. Create the Sitemap of Public & Private Site



Microsoft PowerPoint

1. Development Tools

LITHAN



XAAMP Control Panel v3.3.0

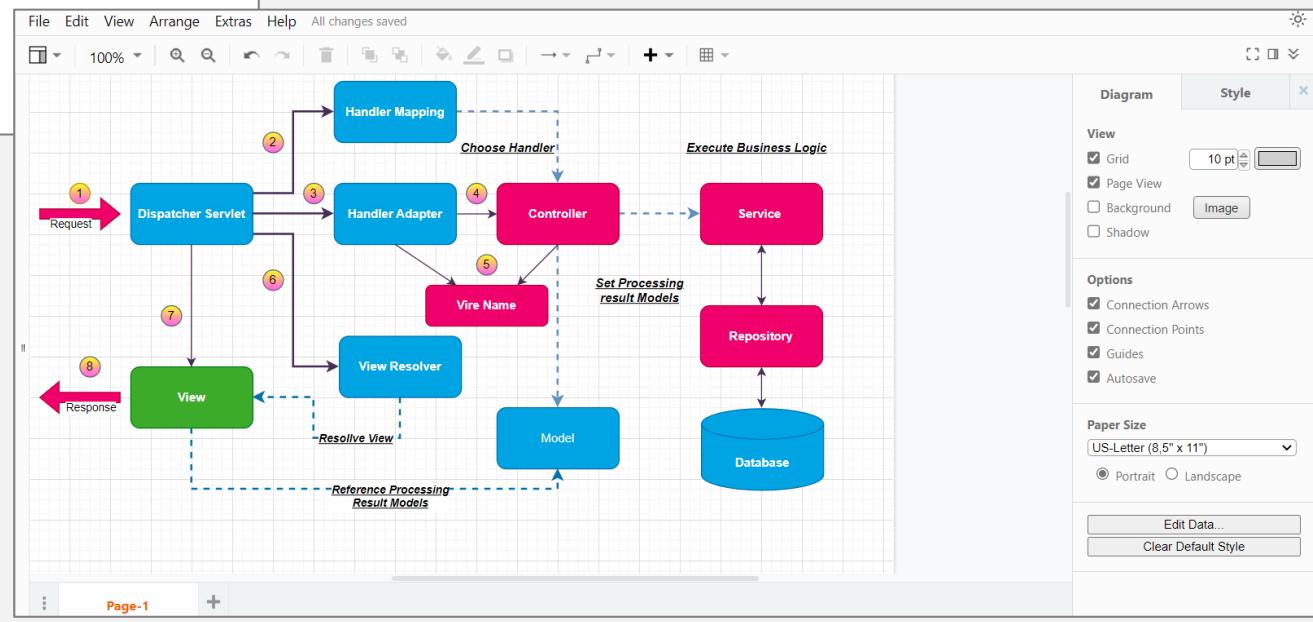
XAMPP Control Panel v3.3.0 [Compiled: Apr 6th 2021]

Modules	Service	Module	PID(s)	Port(s)	Actions
	Apache		17620 14872	80, 443	Stop Admin Config Logs Shell
	MySQL		10128	3306	Stop Admin Config Logs Explorer
	FileZilla				Start Admin Config Logs Services
	Mercury				Start Admin Config Logs Help
	Tomcat				Start Admin Config Logs Quit

```
4:20:51 PM [main] Initializing Control Panel
4:20:51 PM [main] Windows Version: Enterprise 64-bit
4:20:51 PM [main] XAMPP Version: 8.1.6
4:20:51 PM [main] Control Panel Version: 3.3.0 [ Compiled: Apr 6th 2021 ]
4:20:51 PM [main] You are not running with administrator rights! This will work for most application stuff but whenever you do something with services there will be a security dialogue or things will break! So think about running this application with administrator rights!
4:20:51 PM [main] XAMPP Installation Directory: "c:\xampp\"
4:20:51 PM [main] Checking for prerequisites
4:20:51 PM [main] All prerequisites found
4:20:51 PM [main] Initializing Modules
4:20:51 PM [main] Starting Check-Timer
4:20:51 PM [main] Control Panel Ready
4:20:54 PM [Apache] Attempting to start Apache app...
4:20:54 PM [Apache] Status change detected: running
4:20:55 PM [mysql] Attempting to start MySQL app...
4:20:56 PM [mysql] Status change detected: running
```



Diagram.Net

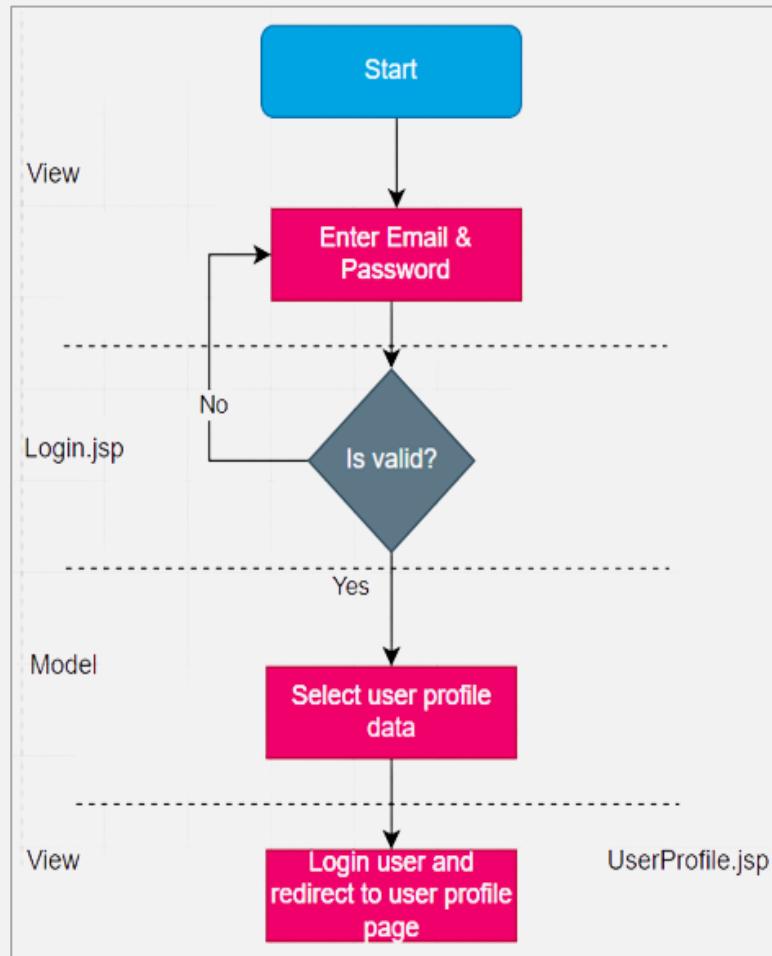


2. Business Process

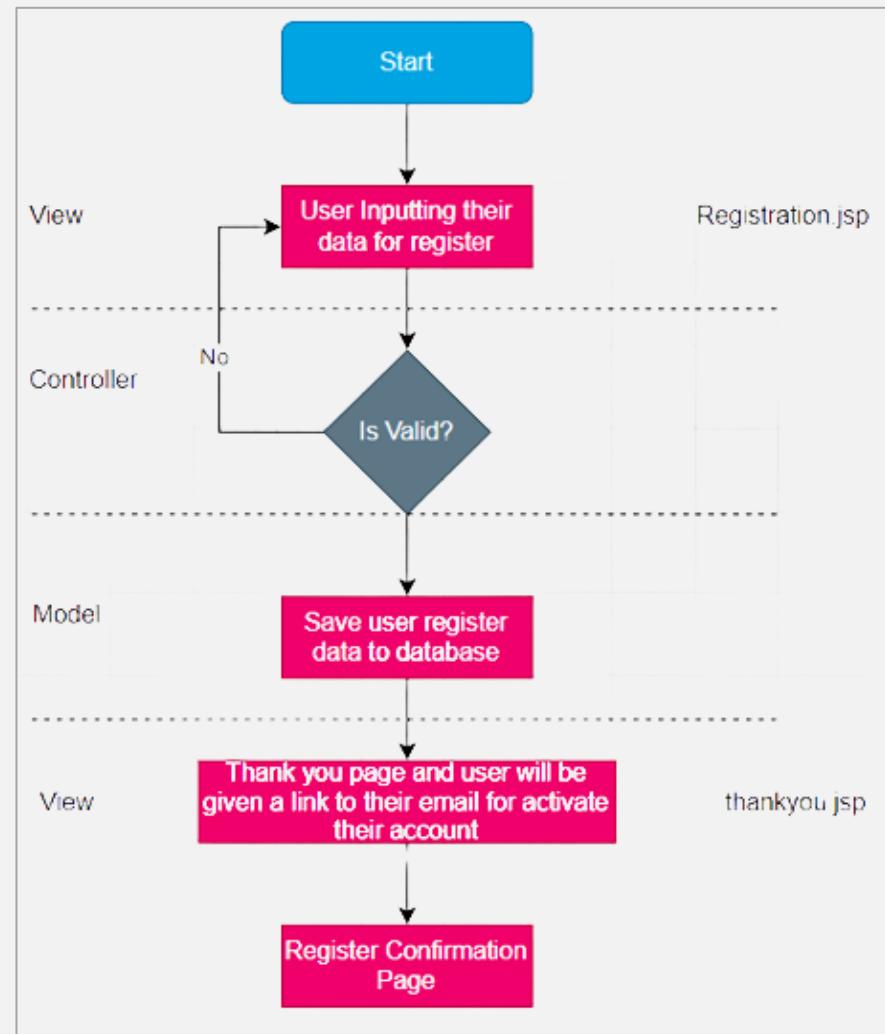
LITHAN

Block Diagram of Various Business Process

Login



Registration

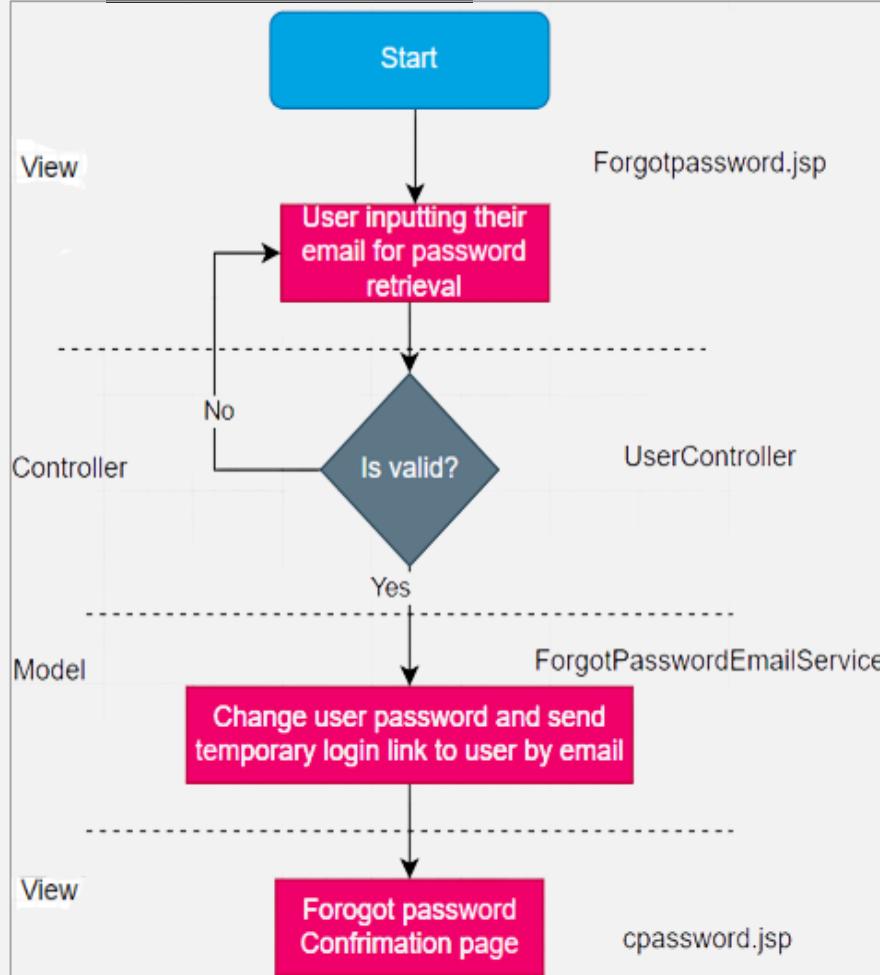


2. Business Process

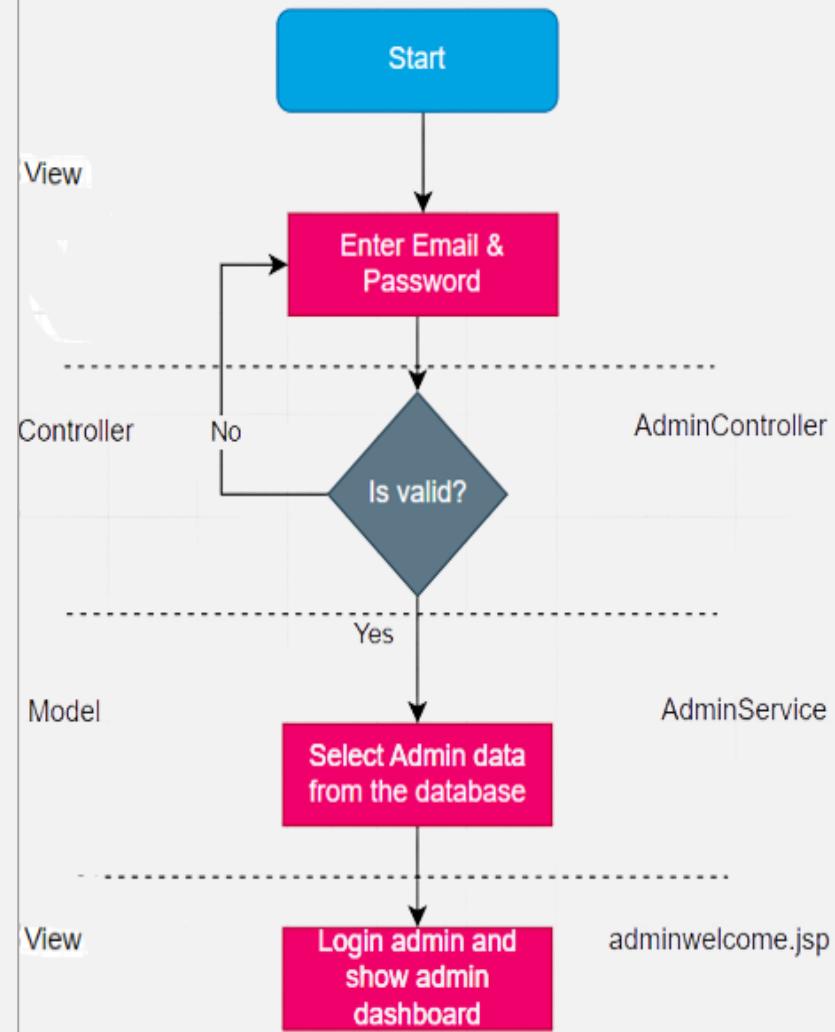
LITHAN

Block Diagram of Various Business Process

Forgot Password



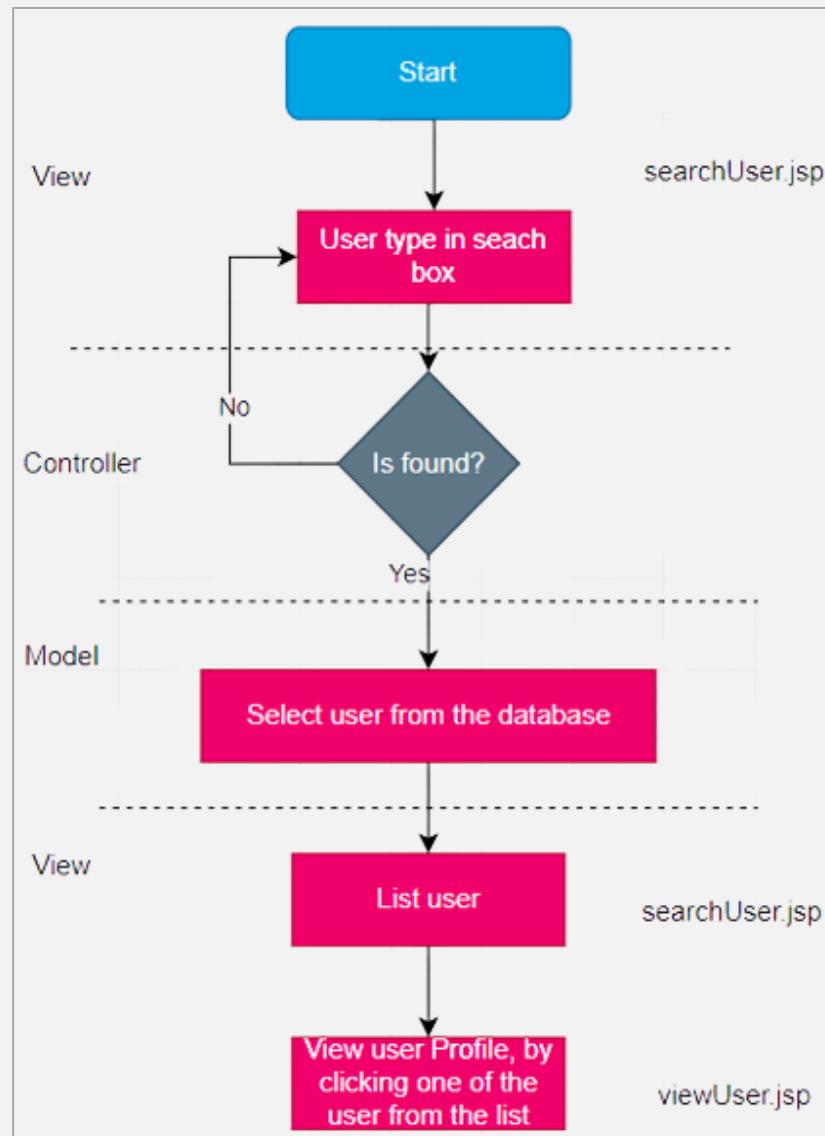
Admin User



2. Business Process

LITHAN

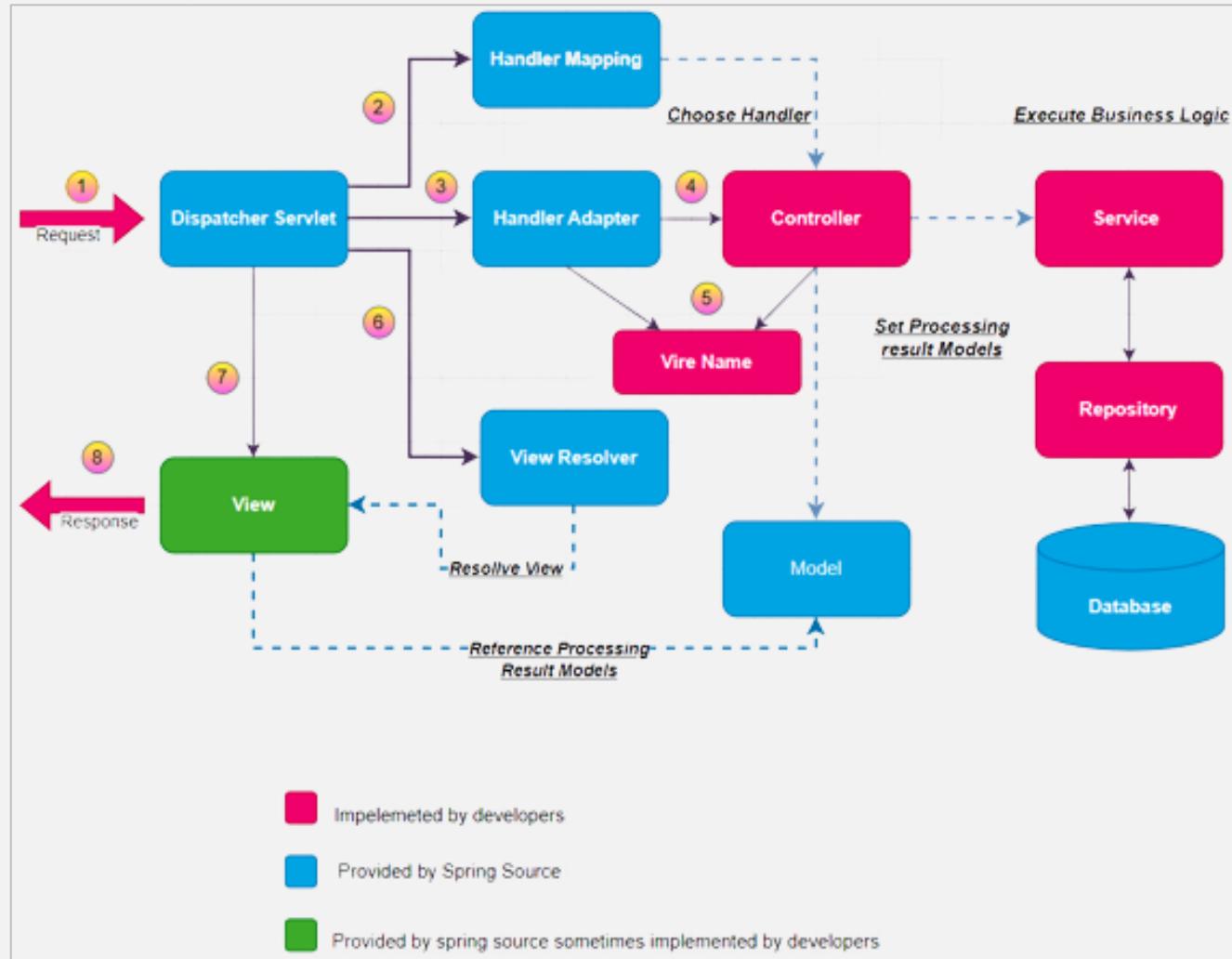
Search User



3. Application Block Diagram

LITHAN

Block Diagram of the Over all Application



3. Application Block Diagram

LITHAN

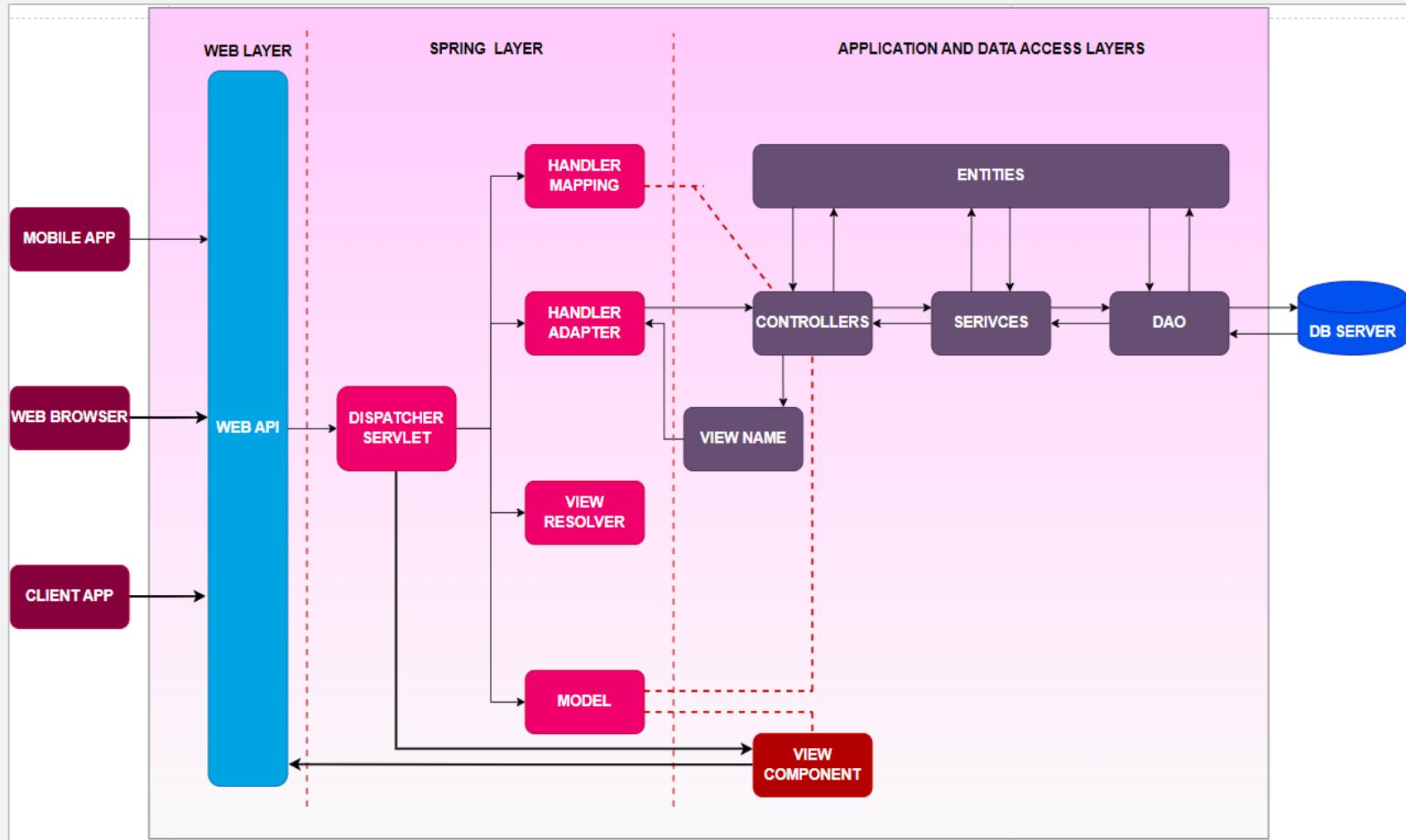
Architecture of the Application

All the jsp pages of this ABC community portal is handled by the controller. The controller requests a response from the service and after receiving the response from the service it calls the dao layer which is the data access layer. Then Dao layer takes data from Database. After this whole process application will work properly. After successfully importing the ABC JOB portal on the System the running server will Go to the dispatcher servlet for view resolver. The dispatcher servlet will locate server to view resolver Where Web-Inf called the JSP pages and css pages and also the Annotation class called all the annotated classes and behaving/presenting the multipart view of JSP pages according to the classes.

3. Application Block Diagram

LITHAN

□ Architecture of the system



3. Application

LITHAN

Architecture of the system

1. Private Module - Administrator - Manage user Data and Bulk Emailing.

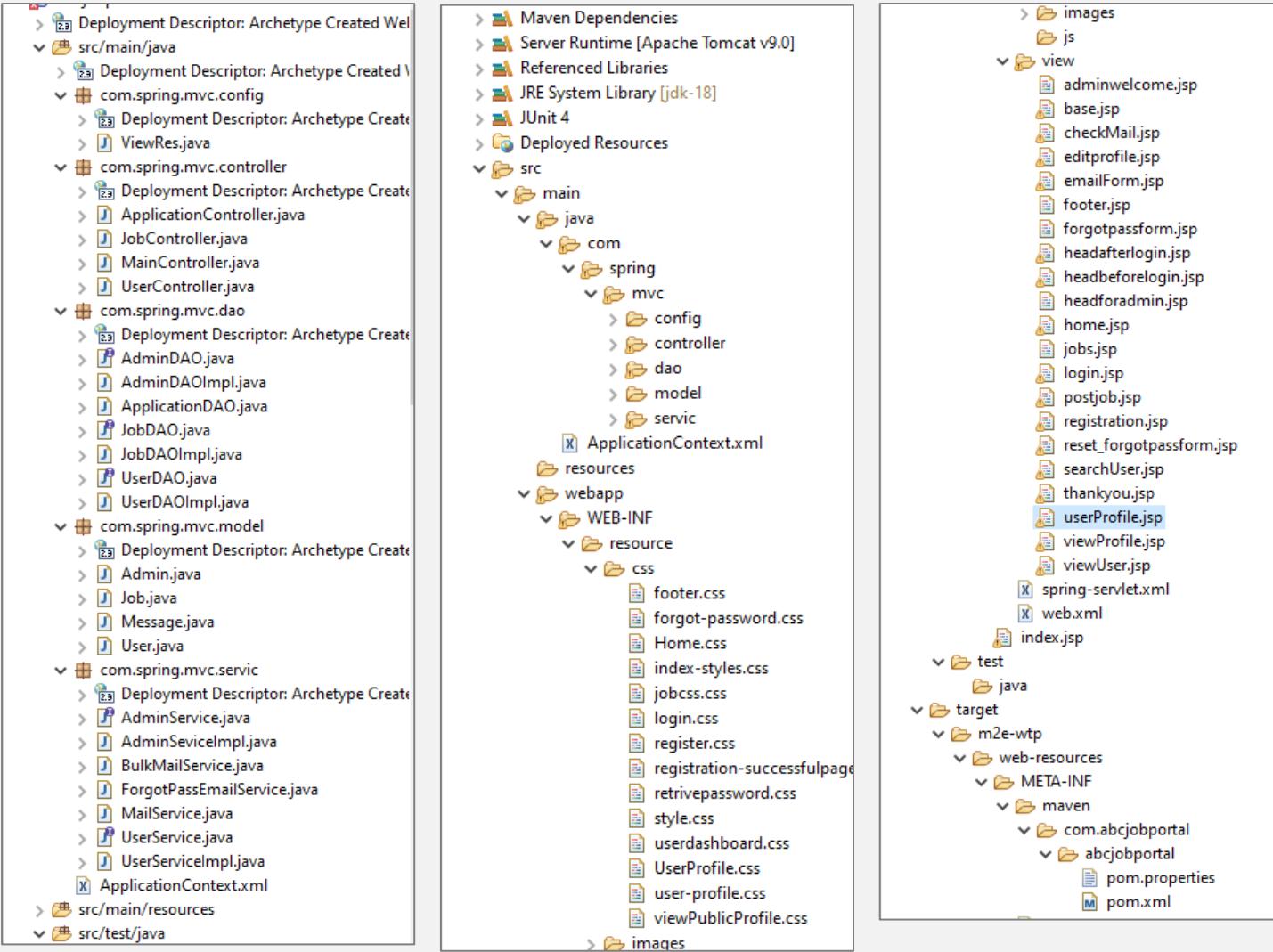
2. Public Module - Software Programmers

- Registration Page
- Login Page
- Forgot Password Page
- Forgot Password Reset
- User Profile Page
- User Home Page
- Edit User Profile
- Search User Page

5. Models, Controllers, Classes Developed LITHAN

□ Overview of Models, Controllers & Any Classes Developed

Project Structure



5. Models, Controllers, Classes Developed **LITHAN**

□ Overview of Models, Controllers & Any Classes Developed

Project Structure

1. com.spring.mvc.config

- **ViewRes (To connect resources files "images, css,js")**

```
1 package com.spring.mvc.config;
2
3 import org.springframework.context.annotation.Bean;
4 import org.springframework.context.annotation.ComponentScan;
5 import org.springframework.context.annotation.Configuration;
6 import org.springframework.web.servlet.config.annotation.EnableWebMvc;
7 import org.springframework.web.servlet.config.annotation.ResourceHandlerRegistry;
8 import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;
9 import org.springframework.web.servlet.view.InternalResourceViewResolver;
10
11 @EnableWebMvc
12 @Configuration
13 @ComponentScan("abcjobportal")
14 public class ViewRes implements WebMvcConfigurer {
15
16     @Bean(name = "viewResolver")
17     public InternalResourceViewResolver getViewResolver() {
18         InternalResourceViewResolver viewResolver = new InternalResourceViewResolver();
19         viewResolver.setPrefix("/WEB-INF/views/");
20         viewResolver.setSuffix(".jsp");
21         return viewResolver;
22     }
23 }
24
25 @Override
26 public void addResourceHandlers(ResourceHandlerRegistry registry)
27 {
28     registry.addResourceHandler("/resource/**")
29         .addResourceLocations("/resource/css")
30         .addResourceLocations("/resource/images")
31         .addResourceLocations("/resource/js")
32     ;
33 }
34 }
35 }
```

Used for displaying the information to the user in a specific format.

5. Models, Controllers, Classes Developed LITHAN

2. com.spring.mvc.controller

✓ ApplicationController.java

```
1 package com.spring.mvc.controller;
2
3 import java.util.List;
4
5 import org.springframework.beans.factory.annotation.Autowired;
6 import org.springframework.context.ApplicationContext;
7 import org.springframework.context.support.ClassPathXmlApplicationContext;
8 import org.springframework.stereotype.Controller;
9 import org.springframework.ui.Model;
10 import org.springframework.web.bind.annotation.ModelAttribute;
11 import org.springframework.web.bind.annotation.PathVariable;
12 import org.springframework.web.bind.annotation.RequestMapping;
13 import org.springframework.web.bind.annotation.RequestMethod;
14 import org.springframework.web.bind.annotation.RequestParam;
15
16 import com.spring.mvc.dao.ApplicationDAO;
17
18 import com.spring.mvc.model.User;
19
20 @Controller
21 public class ApplicationController {
22
23     private ApplicationContext context;
24     @Autowired
25     private ApplicationDAO applicationDAO;
26
27     @RequestMapping(value = "/resetPass", method = RequestMethod.POST)
28     public String edit(@ModelAttribute("bean") User user, Model model) {
29         context = new ClassPathXmlApplicationContext("ApplicationContext.xml");
30         ApplicationDAO obj = context.getBean("dao", ApplicationDAO.class);
31         List list = obj.resetPassword(user);
32         model.addAttribute("data", list);
33         model.addAttribute("reset", " Reset Password Successful");
34         return "login";
35     }
36
37     @RequestMapping("/edit/{userId}")
38     public String updateUserForm(@PathVariable("userId") Long uId, Model model) {
39         User user = this.applicationDAO.getUser(uId);
40         model.addAttribute("user", user);
41         return "editprofile";
42     }
43
44     // for update profile
45     @RequestMapping(value = "edit/update", method = RequestMethod.POST)
46     public String updateUser(@RequestParam("id") Long id, @RequestParam("name") String name,
47                             @RequestParam("email") String email, @RequestParam("contact") String contact,
48                             @RequestParam("city") String city) {
49         User user = new User();
50         user.setId(id);
51         user.setName(name);
52         user.setEmail(email);
53         user.setCity(city);
54         user.setContact(contact);
55
56         applicationDAO.updateprofile(user);
57         return "editprofile";// will come edit profile page
58
59 }
```

Used to mark a class as a web request handler.

```
62     @RequestMapping("/find")
63     public String search(Model model) {
64         context = new ClassPathXmlApplicationContext("ApplicationContext.xml");
65         User user = context.getBean("argon", User.class);
66         model.addAttribute("bean", user);
67         return "search";
68     }
69
70     @RequestMapping(value = "/searchview", method = RequestMethod.GET)
71     public String searchView(@ModelAttribute("bean") User user, Model model) {
72         return "search";
73     }
74
75     @RequestMapping(value = "/searchUser", method = RequestMethod.POST)
76     public String searchUsrs(@ModelAttribute("bean") User user, Model model) {
77         context = new ClassPathXmlApplicationContext("ApplicationContext.xml");
78         ApplicationDAO obj = context.getBean("dao", ApplicationDAO.class);
79
80         List list = obj.search(user);
81         if (!list.isEmpty()) {
82             model.addAttribute("data", list);
83         } else {
84
85             model.addAttribute("msg", "User Not found");
86         }
87         return "searchUser";
88     }
89
90     @RequestMapping("searchUser")
91     public String searchUser(Model model, @ModelAttribute("bean") User user) {
92         context = new ClassPathXmlApplicationContext("ApplicationContext.xml");
93
94         ApplicationDAO obj = context.getBean("dao", ApplicationDAO.class);
95         List<User> list = obj.display();
96
97         model.addAttribute("data", list);
98
99         return "searchUser";
100    }
101    @RequestMapping("viewUser")
102    public String viewUser(Model model, @ModelAttribute("bean") User user) {
103        context = new ClassPathXmlApplicationContext("ApplicationContext.xml");
104
105        ApplicationDAO obj = context.getBean("dao", ApplicationDAO.class);
106        List<User> list = obj.display();
107
108        model.addAttribute("data", list);
109
110        return "viewUser";
111    }
112
113    }
114
115    @RequestMapping("/viewProfile/{userId}")
116    public String userProfilesearch(@PathVariable("userId") Long userId, Model model) {
117
118        User user = this.applicationDAO.getUser(userId);
119        model.addAttribute("user", user);
120        return "viewProfile";
121    }
122 }
```

```
124     @RequestMapping("admindash")
125     public String admindisplay(Model model, @ModelAttribute("bean") User user) {
126         context = new ClassPathXmlApplicationContext("ApplicationContext.xml");
127
128         ApplicationDAO obj = context.getBean("dao", ApplicationDAO.class);
129         List<User> list = obj.display();
130
131         model.addAttribute("data", list);
132
133         return "admindash";
134     }
135
136     @RequestMapping("/findadmin")
137     public String adminsearch(Model model) {
138         context = new ClassPathXmlApplicationContext("ApplicationContext.xml");
139         User user = context.getBean("argon", User.class);
140         model.addAttribute("bean", user);
141
142         return "admindash";
143     }
144
145     @RequestMapping(value = "/adminsearchview", method = RequestMethod.GET)
146     public String adminsearchView(@ModelAttribute("bean") User user, Model model) {
147         return "admindash";
148     }
149     @RequestMapping(value = "/adminsearchUsrs", method = RequestMethod.POST)
150     public String adminsearchUsrs(@ModelAttribute("bean") User user, Model model) {
151         context = new ClassPathXmlApplicationContext("ApplicationContext.xml");
152         ApplicationDAO obj = context.getBean("dao", ApplicationDAO.class);
153
154         List list = obj.search(user);
155         if (!list.isEmpty()) {
156             model.addAttribute("data", list);
157         } else {
158
159             model.addAttribute("msg", "User Not found");
160         }
161         return "admindash";
162     }
163
164     @RequestMapping(value = "/editor", method = RequestMethod.POST)
165     public String adminedit(@ModelAttribute("bean") User user, Model model) {
166         context = new ClassPathXmlApplicationContext("ApplicationContext.xml");
167         ApplicationDAO obj = context.getBean("dao", ApplicationDAO.class);
168         List list = obj.update(user);
169
170         model.addAttribute("data", list);
171         model.addAttribute("msg", " Record update succesfully");
172         return "admindash";
173     }
174     @RequestMapping(value = "/delete", method = RequestMethod.GET)
175     public String admindetele(@ModelAttribute("bean") User user, Model model) {
176         context = new ClassPathXmlApplicationContext("ApplicationContext.xml");
177         ApplicationDAO obj = context.getBean("dao", ApplicationDAO.class);
178         List list = obj.delete(user);
179
180         model.addAttribute("data", list);
181         model.addAttribute("msg", " Record deleted succesfully");
182         return "viewUser";
183 }
```

5. Models, Controllers, Classes Developed LITHAN

MainController.java

```
1 package com.spring.mvc.controller;
2
3 import org.springframework.stereotype.Controller;
4 import org.springframework.web.bind.annotation.RequestMapping;
5
6 @Controller
7 public class MainController {
8
9     @RequestMapping("/registration")
10    public String registration() {
11        return "registration";
12    }
13
14    @RequestMapping("/thankyou")
15    public String thankyou() {
16        return "thankyou";
17    }
18
19
20    @RequestMapping("/login")
21    public String login() {
22        return "login";
23    }
24
25    @RequestMapping("/jobs")
26    public String jobs() {
27        return "jobs";
28    }
29
30
31    @RequestMapping("/adminwelcome")
32    public String adminwelcome() {
33        return "adminwelcome";
34    }
35
36    @RequestMapping("/postjob")
37    public String postjob() {
38        return "postjob";
39    }
40
41    @RequestMapping("/forgotpassform")
42    public String forgotpassform() {
43        return "forgotpassform";
44}
```

The Controller in MVC architecture **handles any incoming URL request.**
The Controller is a class, derived from the base class System

```
43        return "forgotpassform";
44    }
45}
46 @RequestMapping("/forgot")
47 public String forgot() {
48    return "forgot";
49}
50
51 @RequestMapping("/home")
52 public String home() {
53    return "home";
54}
55
56 @RequestMapping("/userProfile")
57 public String userProfile() {
58    return "userProfile";
59}
60
61 @RequestMapping("/editprofile")
62 public String editprofile() {
63    return "editprofile";
64}
65
66 @RequestMapping("/jobDetails")
67 public String jobDetails() {
68    return "jobDetails";
69}
70
71 @RequestMapping("/viewProfile")
72 public String viewProfile() {
73    return "viewProfile";
74}
75
76
77 @RequestMapping("/emailForm")
78 public String emailForm() {
79    return "emailForm";
80}
81
82 @RequestMapping("error")
83 public String error() {
84    return "error";
85}
86
87 @RequestMapping("success")
88 public String success() {
89    return "success";
90}
91
92}
```

5. Models, Controllers, Classes Developed LITHAN

UserController.java

```
1 package com.spring.mvc.controller;
2
3 import java.util.Map;
4
5 import javax.servlet.http.HttpServlet;
6 import javax.servlet.http.HttpSession;
7
8 import org.springframework.beans.factory.annotation.Autowired;
9 import org.springframework.stereotype.Controller;
10 import org.springframework.ui.Model;
11 import org.springframework.web.bind.annotation.GetMapping;
12 import org.springframework.web.bind.annotation.ModelAttribute;
13 import org.springframework.web.bind.annotation.PathVariable;
14 import org.springframework.web.bind.annotation.PostMapping;
15 import org.springframework.web.bind.annotation.RequestMapping;
16 import org.springframework.web.bind.annotation.RequestMethod;
17 import org.springframework.web.bind.annotation.RequestParam;
18 import org.springframework.web.servlet.ModelAndView;
19
20 import com.spring.mvc.model.Admin;
21 import com.spring.mvc.model.User;
22 import com.spring.mvc.servic.AdminService;
23 import com.spring.mvc.servic.ForgotPassEmailService;
24 import com.spring.mvc.servic.MailService;
25 import com.spring.mvc.servic.UserService;
26
27 @Controller
28 public class UserController {
29
30     @Autowired//To call the userservice
31     private UserService userService;
32     @Autowired//To call the mail servic
33
34     private MailService mailService;
35     @Autowired
36     private AdminService adminService;
37     @Autowired
38     private ForgotPassEmailService forgotPassEmailService;
39
40
41     //To get value from registration form
42     @GetMapping(value="/registration")
43     public ModelAndView registration(Model model) {
44         User user=new User();
45         model.addAttribute("user",user);
46         System.out.println(user);
47         ModelAndView modelAndView=new ModelAndView("registration");
48
49         return modelAndView;
50     }
51 }
```

```
53     @PostMapping("/registration")
54     public String register(@ModelAttribute("user") User user, Model model,
55                           HttpServletRequest request,
56                           String emailToRecipient, String userName, String EmailID) {
57         userName = request.getParameter("email");
58         user = request.getParameter("name");
59
60         userService.registerUser(user);
61         mailService.sendMail(emailToRecipient, userName); // sending mail to the user
62         that his registration is completed
63         request.getSession().setAttribute("name", userName);
64         request.getSession().setAttribute("email", emailToRecipient);
65
66         model.addAttribute("success", "Registration is successful");
67     }
68     @RequestMapping(value = "/login", method = RequestMethod.POST)
69     public String login(@ModelAttribute("user") User user, Model model, HttpSession
70                         session,
71                         @ModelAttribute("admin") Admin admin) {
72         User user2 = userService.loginUser(user); // For user
73         Admin user3 = adminService.adminLogin(admin); // For admin
74         System.out.println("user2" + user2);
75         System.out.println("user3" + user3);
76
77         if (user2 != null) { // If user not null it will come welcome page for him
78             System.out.println("hello");
79             model.addAttribute("user", user2);
80             session.setAttribute("user", user2);
81             return "home";
82         }
83
84         if (user3 != null) { // If admin not null it will come adminwelcome page for him
85             System.out.println("hello");
86             model.addAttribute("admin", user3);
87             session.setAttribute("admin", user3);
88             return "adminwelcome";
89
90         //If user could not get it will show error
91         if (user2 == null) {
92             System.out.println("on");
93             model.addAttribute("error", "Invalid Credential");
94         }
95
96         //If user information can not get it will show error
97         if (user3 == null) {
98             System.out.println("on");
99             model.addAttribute("error", "Invalid Credential");
100        }
101    }
102
103    return "login";
104}
105 }
```

```
108     //logout
109     public String loginDisplay(Model m, HttpSession session) {
110         User user=new User();
111         if(session.getAttribute("user") !=null){
112             session.invalidate();
113             System.out.println("here");
114             m.addAttribute("success", "You have logout successfully");
115         }
116         m.addAttribute("user", user);
117         return "login";
118     }
119
120
121
122
123
124     //For checking email is it in database or not and send mail to the definite email
125     //start
126     @RequestMapping(value = "/emailConfirm", method = RequestMethod.POST)
127     public String checkMail(@ModelAttribute("user") User user, Model model,
128                           HttpSession session,
129                           @RequestParam(value = "email") String email, HttpServletRequest request) {
130         User user4 = userService.checkMail(user);
131         System.out.println(user4);
132         if (user4 != null) {
133             System.out.println("hello");
134             model.addAttribute("user", user4);
135             session.setAttribute("user", user4);
136             forgotPassEmailService.resetMail(email); // sending mail to the user to reset
137             his/her password
138             request.getSession().setAttribute("email", email);
139
140             //return "checkMail";
141             return "checkMail";
142         }
143         if (user4 == null) {
144             System.out.println("on");
145             model.addAttribute("error", "Invalid Credential");
146         }
147
148         return "forgotpassform";
149
150
151
152
153
154
155     @RequestMapping(value="/resetPassword/{email}")
156     public String resetPassword(@PathVariable String email, Map<String, String> model)
157     {
158         //check if the email id is valid and registered with us.
159         model.put("emailid", email);
160         return "reset_forgotpassform";
161
162
163
164 }
```

5. Models, Controllers, Classes Developed LITHAN

3. com.spring.mvc.dao

✓ AdminDao.java

```
1 package com.spring.mvc.dao;
2
3 import com.spring.mvc.model.Admin;
4
5 public interface AdminDAO {
6
7     public Admin adminlogin(Admin admin);
8 }
```

DAO means data access object. It support in Spring is aimed at making it easy to work with data access technologies like JDBS, Hibernate, JPA or JDO in a consistent way

✓ AdminDaoImpl.java

```
1 package com.spring.mvc.dao;
2
3 import javax.persistence.NoResultException;
4
5 @Repository
6 @Transactional
7 public class AdminDAOImpl implements AdminDAO{
8     @Autowired
9     private SessionFactory factory;
10
11     public Admin adminlogin(Admin admin) {
12
13         Session session=factory.getCurrentSession();
14         try {
15             Query<Admin> query=session.createQuery("FROM admin_table where email=:email and password=:password",Admin);
16             query.setParameter("email",admin.getEmail());
17             query.setParameter("password",admin.getPassword());
18             admin=(Admin) query.getSingleResult();
19             // query.setMaxResults(1).uniqueResult();
20
21             return admin;
22         }catch(NoResultException e) {
23             return null;
24         }
25
26     }
27
28 }
```

5. Models, Controllers, Classes Developed LITHAN

✓ ApplicationDao.java

```
1 package com.spring.mvc.dao;
2
3 import java.sql.PreparedStatement;
4 import java.sql.ResultSet;
5 import java.sql.SQLException;
6 import java.util.ArrayList;
7 import java.util.List;
8
9 import javax.transaction.Transactional;
10
11 import org.springframework.beans.factory.annotation.Autowired;
12 import org.springframework.dao.DataAccessException;
13 import org.springframework.jdbc.core.JdbcTemplate;
14 import org.springframework.jdbc.core.PreparedStatementCallback;
15 import org.springframework.jdbc.core.ResultSetExtractor;
16 import org.springframework.orm.hibernate5.HibernateTemplate;
17 import org.springframework.stereotype.Repository;
18
19 import com.spring.mvc.model.Job;
20 import com.spring.mvc.model.User;
21
22 @Repository
23 public class ApplicationDAO {
24     @Autowired
25     private JdbcTemplate jdbcTemplate;
26
27     @Autowired
28     private HibernateTemplate hibernateTemplate;
29
30     //getting the data from jdbc template
31     public JdbcTemplate getJdbcTemplate() {
32         return jdbcTemplate;
33     }
34
35     //setting the data from jdbc
36     public void setJdbcTemplate(JdbcTemplate jdbcTemplate) {
37         this.jdbcTemplate = jdbcTemplate;
38     }
39
40     public List search(User user) {
41         final List list = new ArrayList();
42         String sql = "select * from user_table where name = '" + user.getName() + "' or
43             email = '" + user.getEmail()
44             + "' or contact = '" + user.getContact() + "' or password = '" + user.getPassword() +
45             "' or id = '" + user.getId() + "'";
46         jdbcTemplate.query(sql, new ResultSetExtractor<User>() {
47
48             public User extractData(ResultSet rs) throws SQLException, DataAccessException
49             {
50                 while (rs.next()) {
51                     User user = new User();
52                     user.setId(rs.getLong(1));
53                     user.setName(rs.getString(2));
54                     user.setEmail(rs.getString(3));
55                     user.setContact(rs.getString(4));
56                     user.setCity(rs.getString(5));
57                     user.setPassword(rs.getString(6));
58
59                     list.add(user);
60                 }
61             }
62         });
63
64         public List display() {
65             final List list = new ArrayList();
66             String sql = "select * from user_table";
67             jdbcTemplate.query(sql, new ResultSetExtractor<User>() {
68                 public User extractData(ResultSet rs) throws SQLException, DataAccessException
69                 {
70                     while (rs.next()) {
71                         User user = new User();
72                         user.setId(rs.getLong(1));
73                         user.setName(rs.getString(2));
74                         user.setEmail(rs.getString(3));
75                         user.setContact(rs.getString(4));
76                         user.setCity(rs.getString(5));
77                         user.setPassword(rs.getString(6));
78
79                         list.add(user);
80                     }
81                 }
82             });
83             return list;
84         }
85
86         public List resetPassword(final User user) {
87             String sql = "update user_table set password=? where email=?";
88             jdbcTemplate.execute(sql, new PreparedStatementCallback<Boolean>() {
89
90                 public Boolean doInPreparedStatement(PreparedStatement ps) throws SQLException,
91                     DataAccessException {
92                     ps.setString(1, user.getPassword());
93                     ps.setString(2, user.getEmail());
94
95                     return ps.execute();
96                 }
97             });
98
99             List list = display();
100            return list;
101        }
102
103        public List updateprofile(final User user) {
104            String sql = "update user_table set name=?,email=?,contact=? where id=?";
105            jdbcTemplate.execute(sql, new PreparedStatementCallback<Boolean>() {
106
107                public Boolean doInPreparedStatement(PreparedStatement ps) throws SQLException,
108                    DataAccessException {
109                    ps.setLong(1, 1);
110                    ps.setString(2, user.getName());
111                    ps.setString(3, user.getContact());
112                    ps.setString(4, user.getEmail());
113
114                    return ps.execute();
115                }
116            });
117        }
118    }
119    List list = display();
120    return list;
121}
122
123    public List<User> findUserByID(Long id) {
124        final List list = new ArrayList();
125        String sql = "select * from user_table where id=?";
126        jdbcTemplate.query(sql, new ResultSetExtractor<User>() {
127
128            public User extractData(ResultSet rs) throws SQLException, DataAccessException
129            {
130                while (rs.next()) {
131                    User user = new User();
132                    user.setId(rs.getLong(1));
133                    user.setName(rs.getString(2));
134                    user.setEmail(rs.getString(3));
135                    user.setContact(rs.getString(4));
136                    user.setCity(rs.getString(5));
137                    user.setPassword(rs.getString(6));
138
139                    list.add(user);
140                }
141            }
142        });
143    }
144    return list;
145}
146
147    @Transactional
148    public User getUser(Long userId) {
149        return this.hibernateTemplate.get(User.class, userId);
150    }
151
152    @Transactional
153    public List<Job> getjobDetails() {
154        List<Job> jobs = this.hibernateTemplate.loadAll(Job.class);
155
156        return jobs;
157    }
158}
159
160
161    public List update(final User user) {
162        String sql = "update user_table set name=?,email=?,contact=?,city=? where id=?";
163        jdbcTemplate.execute(sql, new PreparedStatementCallback<Boolean>() {
164
165            public Boolean doInPreparedStatement(PreparedStatement ps) throws SQLException,
166                DataAccessException {
167                ps.setString(4, user.getCity());
168
169                ps.setString(3, user.getContact());
170                ps.setString(2, user.getEmail());
171                ps.setString(1, user.getName());
172
173                ps.setLong(5, user.getId());
174            }
175
176        });
177
178        // TODO Auto-generated method stub
179        List list = display();
180        return list;
181    }
182
183    public List delete(User user) {
184        String str = "delete from user_table where id=" + user.getId() + " ";
185        jdbcTemplate.update(str);
186        List list = display();
187        return list;
188    }
189}
190}
```

5. Models, Controllers, Classes Developed LITHAN

✓ UserDao.java

```
1 package com.spring.mvc.dao;  
2  
3 import com.spring.mvc.model.User;  
4  
5 public interface UserDAO {  
6     public void registerUser(User user);  
7     public User loginUser(User user);  
8     public User checkMail(User user);  
9 }  
10
```

✓ UserDaoImplementation.java

```
1 package com.spring.mvc.dao;  
2  
3  
4 import javax.persistence.NoResultException;  
5  
6 import org.hibernate.Session;  
7 import org.hibernate.SessionFactory;  
8 import org.hibernate.query.Query;  
9 import org.springframework.beans.factory.annotation.Autowired;  
10 import org.springframework.stereotype.Repository;  
11 import org.springframework.transaction.annotation.Transactional;  
12 import com.spring.mvc.model.User;  
13 @Repository  
14 @Transactional  
15 public class UserDAOImpl implements UserDAO {  
16     //To call the session factory  
17     @Autowired  
18     private SessionFactory factory;  
19  
20     public void registerUser(User user) {  
21         Session session=factory.getCurrentSession();  
22         session.save(user);  
23     }  
24 }
```

5. Models, Controllers, Classes Developed LITHAN

4.com.spring.mvc.model ✓ Admin.java

Model means bean. JavaBeans are classes that encapsulate many objects into a single object(the bean). It is a java class that should follow conventions and also model can be an object or collection of objects which basically contains the data of the application

```
1 package com.spring.mvc.model;
2
3 import javax.persistence.GeneratedValue;
4 import javax.persistence.GenerationType;
5 import javax.persistence.Id;
6 import javax.validation.constraints.NotNull;
7
8 @javax.persistence.Entity(name = "admin_table")
9 @javax.persistence.Table(name = "admin_table")
10 public class Admin {
11     @Id
12     @GeneratedValue(strategy = GenerationType.IDENTITY)
13     private Long id;
14     @NotNull
15     private String name;
16     @NotNull
17     private String email;
18     @NotNull
19     private String password;
20     public Long getId() {
21         return id;
22     }
23     public void setId(Long id) {
24         this.id = id;
25     }
26     public String getName() {
27         return name;
28     }
29     public void setName(String name) {
30         this.name = name;
31     }
32     public String getEmail() {
33         return email;
34     }
35     public void setEmail(String email) {
36         this.email = email;
37     }
38     public String getPassword() {
39         return password;
40     }
41     public void setPassword(String password) {
42         this.password = password;
43     }
44     public Admin(Long id, String name, String email, String password) {
45         super();
46         this.id = id;
47         this.name = name;
48         this.email = email;
49         this.password = password;
50     }
51     @Override
52     public String toString() {
53         return "Admin [id=" + id + ", name=" + name + ", email=" + email + ", password="
54         + password + "]";
55     }
56     public Admin() {
57         super();
58         // TODO Auto-generated constructor stub
59     }
60 }
```

5. Models, Controllers, Classes Developed LITHAN

✓ User.java

```
41}
42 public void setCity(String city) {
43     this.city = city;
44 }
45 public String getPassword() {
46     return password;
47 }
48 public void setPassword(String password) {
49     this.password = password;
50 }
51 public String getContact() {
52     return contact;
53 }
54 public void setContact(String contact) {
55     this.contact = contact;
56 }
57 public String getEmail() {
58     return email;
59 }
60 public void setEmail(String email) {
61     this.email = email;
62 }
63 public User(Long id, String name, String city, String password, String contact,
64             String email) {
65     super();
66     this.id = id;
67     this.name = name;
68     this.city = city;
69     this.password = password;
70     this.contact = contact;
71     this.email = email;
72 }
73 @Override
74 public String toString() {
75     return "User [id=" + id + ", name=" + name + ", city=" + city + ", password=" +
76             password + ", contact=" + contact
77             + ", email=" + email + "]";
78 }
79 public User() {
80     super();
81     // TODO Auto-generated constructor stub
82 }
83 }
```

```
1 package com.spring.mvc.model;
2
3 import javax.persistence.Column;
4 import javax.persistence.GeneratedValue;
5 import javax.persistence.GenerationType;
6 import javax.persistence.Id;
7
8 @javax.persistence.Entity(name = "User_table")
9 @javax.persistence.Table(name = "User_table")
10 public class User {
11
12     @Id
13     @GeneratedValue(strategy = GenerationType.IDENTITY)
14
15     @Column(name="id")
16     private Long id;
17     @Column(name="name")
18     private String name;
19     @Column(name="city")
20     private String city;
21     @Column(name="password")
22     private String password;
23     @Column(name="contact")
24     private String contact;
25     @Column(name="email")
26     private String email;
27     public Long getId() {
28         return id;
29     }
30     public void setId(Long id) {
31         this.id = id;
32     }
33     public String getName() {
34         return name;
35     }
36     public void setName(String name) {
37         this.name = name;
38     }
39     public String getCity() {
40         return city;
41     }
42 }
```

5. Models, Controllers, Classes Developed LITHAN

5. com.spring.mvc.service

- ✓ AdminService.java

```
1 package com.spring.mvc.servic;
2
3 import com.spring.mvc.model.Admin;
4
5 public interface AdminService {
6     public Admin adminlogin(Admin admin);
7 }
8
```

A service class is used by a client to interact with some functionality in the application. Usually, it is public and has some business meaning.

- ✓ AdminServiceImpl.java

```
1 package com.spring.mvc.servic;
2
3+ import javax.transaction.Transactional;...
4
5 @Service
6 @Transactional
7 public class AdminSeviceImpl implements AdminService {
8
9     @Autowired
10    private AdminDAO adminDAO;
11
12    public Admin adminlogin(Admin admin) {
13        // TODO Auto-generated method stub
14        return adminDAO.adminlogin(admin);
15    }
16
17 }
```

5. Models, Controllers, Classes Developed LITHAN

✓ BulkMailService.java

```
11 import org.springframework.stereotype.Service;
12
13
14 @Service
15 public class BulkMailService {
16
17
18@   @Autowired
19   private JavaMailSender mailSender;
20
21@   public void sendMails(String receiver , String subject, String body) {
22
23     MimeMessage message = this.mailSender.createMimeMessage();
24     MimeMessageHelper mimeHelper;
25
26     try {
27       mimeHelper = new MimeMessageHelper(message, true);
28       mimeHelper.setTo(receiver);
29       mimeHelper.setFrom("abcjob36@gmail.com");
30       mimeHelper.setSubject(subject);
31       mimeHelper.setText(body, true);
32       mailSender.send(message);
33
34     } catch (MessagingException e) {
35       System.out.println("Error Sending message" + e.getMessage());
36     }
37   }
38 }
```

✓ ForgotPasswordMail.java

```
1 package com.spring.mvc.servic;
2
3@import javax.mail.MessagingException;□
10
11 @Service
12 public class ForgotPassEmailService {
13
14@   @Autowired
15
16   private JavaMailSender mailSender;
17
18@   public void resetMail(String emailID) {
19     MimeMessage message = this.mailSender.createMimeMessage();
20     MimeMessageHelper messageHelper;
21     try {
22       messageHelper = new MimeMessageHelper(message, true);
23       messageHelper.setTo(emailID);
24       messageHelper.setFrom("abcjobs36@gmail.com");
25       messageHelper.setSubject("Forgot password reset");
26       messageHelper.setText(
27         "<html><body style='font-size: 1rem;'><h2>Good day! This is ABC JOBS</h2><h4>Use this link to
28         + <button style='''>
29         + " border-radius: 1.5rem; text-decoration:none;background:white;padding:15px; color:
30         + <a href='http://localhost:8082/portalAbc/reset_forgotpassform/" + emailID
31         + "/'> Reset your Password</a></button> </body></html>",
32         true);
33       mailSender.send(message);
34     } catch (MessagingException e) {
35       System.out.println("Error sending mail" + e.getMessage());
36     }
37   }
38 }
```

5. Models, Controllers, Classes Developed LITHAN

✓ MailService.java

```
1 package com.spring.mvc.servic;
2
3 import java.util.Date;
4
5 @Service
6 public class MailService {
7
8     @Autowired
9     private JavaMailSender mailSender;
10
11     public void sendMail(String emailId, String userName) {
12
13         MimeMessage message = this.mailSender.createMimeMessage();
14         MimeMessageHelper mimeHelper;
15
16         try {
17             mimeHelper = new MimeMessageHelper(message, true);
18             mimeHelper.setTo(emailId);
19             mimeHelper.setFrom("abcjob3@gmail.com");
20             mimeHelper.setSubject("Registration Confirmation");
21             mimeHelper.setText("<html><body><h1>ABC Ptd Ltd</h1>" + "\n" + "<h3>Hi " + userName + "</h3>" + "\n" +
22                         + " <h3 style='color: rgb(114, 114, 202);>You have registered successfully</h3>\r\n"
23                         + "" + "\n" + new Date() + "</body></html>", true);
24             mailSender.send(message);
25
26         } catch (MessagingException e) {
27             System.out.println("Error Sending message" + e.getMessage());
28         }
29     }
30
31 }
32
33 }
```

✓ UserService.java

```
1 package com.spring.mvc.servic;
2
3 import com.spring.mvc.model.User;
4
5
6 public interface UserService {
7     public void registerUser(User user);
8     public User loginUser(User user);
9     public User checkMail(User user);
10 }
11
```

5. Models, Controllers, Classes Developed **LITHAN**

✓ **UserServiceImplementation.java**

```
1 package com.spring.mvc.servic;
2
3④ import javax.transaction.Transactional;□
4 @Service
5 @Transactional
6 @Repository
7 public class UserServiceImpl implements UserService {
8     @Autowired
9     private UserDAO userDAO;
10
11    public void registerUser(User user) {
12        userDAO.registerUser(user);
13    }
14
15    public User loginUser(User user) {
16        return userDAO.loginUser(user);
17    }
18
19    public User checkMail(User user) {
20        // TODO Auto-generated method stub
21        return userDAO.checkMail(user);
22    }
23
24 }
```

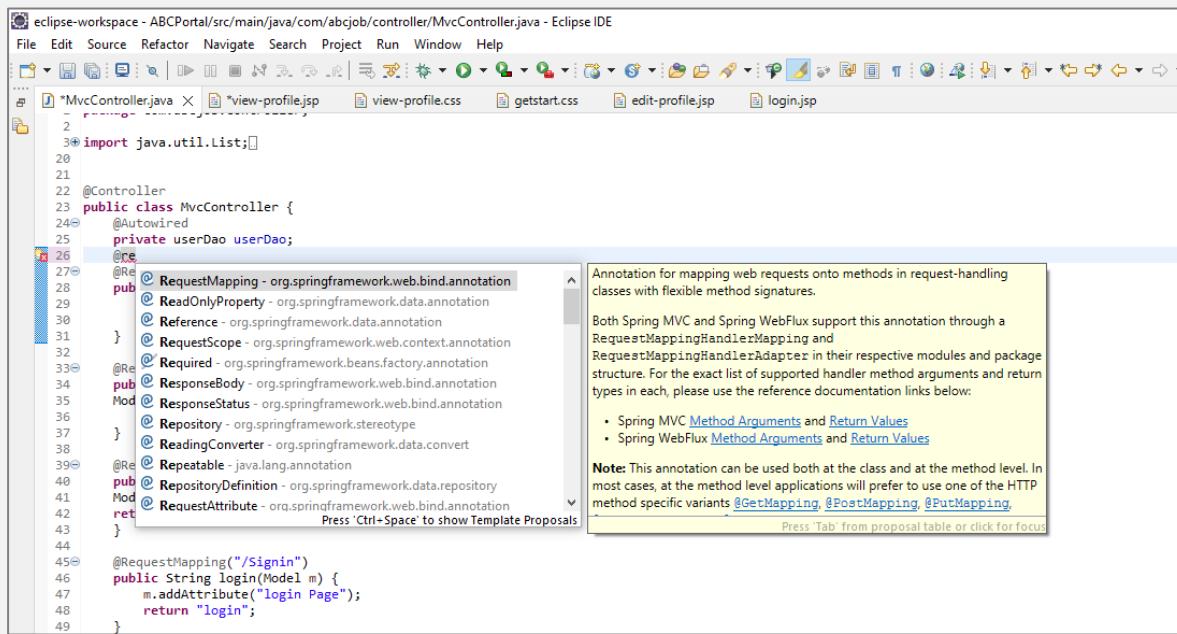
6. Tool Screens

LITHAN

❑ Tool Screens features

Smart Code Completion

We get code of completion as quickly as we start typing in eclipse, we don't always need to apply a keyboard shortcut to deliver the suggestions. suggestions are also context sensitive so that they must make feel for the current place we're typing



6. Tool Screens

LITHAN

Quick Fixes

Each time Ide detects an error, the eclipse highlights the error the use of a wavy red line underneath the offending code and a marker in the left editor margin. moving the mouse over the underlined code or the marker will indicate the error. The marker may be selected with the left mouse to set off the quick restoration pop-up, indicating moves that may be undertaken to repair the error. rather, pressing Ctrl+1 will activate brief restore from the keyboard.

The screenshot shows a Java code editor in Eclipse. The code is as follows:

```
1 package abcproject.services;
2
3 import org.springframework.stereotype.Repository;
4 import org.springframework.stereotype.Service;
5
6 import abcproject.model.User;
7
8 @Repository
9 @Service
10 public class UserService {
11     public User findByEmailAndPassword(String email, String password){
12
13
14         return userRepository.findByEmailAndPassword(email, password);
15     }
16
17
18
19 }
20
```

A red squiggle underline is under the word `userRepository` in the line `return userRepository.findByEmailAndPassword(email, password);`. A red circle marker is at the start of line 14. A yellow tooltip box titled "userRepository cannot be resolved" appears, containing the message "9 quick fixes available:" followed by a list of options:

- ① Create local variable 'userRepository'
- ② Create field 'userRepository'
- ③ Create parameter 'userRepository'
- ④ Create class 'userRepository'
- ⑤ Create constant 'userRepository'
- ⑥ Change to 'DefaultLoaderRepository' (javax.management)
- ⑦ Change to 'DefaultLoaderRepository' (javax.management.loading)
- ⑧ Change to 'NamedQueryResolution' (org.hibernate.query.criteria)

Getter and Setter

Getters and setters are used to shield your data, in particular when developing classes. For each example variable, a getter method returns its value while a setter method sets or updates its value. Given this, getters and setters are also called accessors and mutators, respectively

```
j
public String getFristname() {
    return fristname;
}
public void setFristname(String fristname) {
    this.fristname = fristname;
}
public String getLastname() {
    return lastname;
}
public void setLastname(String lastname) {
    this.lastname = lastname;
}
public String getEmail() {
    return email;
}
public void setEmail(String email) {
    this.email = email;
}
```

7. Classes - Entities

LITHAN

Entities

S. No.	Class Name	Description
1	User	Entity for connecting to tb_users table, used for saving user account
2	UserProfile	Entity for connecting to tb_user_profile table, used for saving user profile
3	Admin	Entity for connecting to tb_admin table, used for fetch admin table
4	BulkEmail	Entity for sending bulk email from admin side

8. Classes Developed

LITHAN

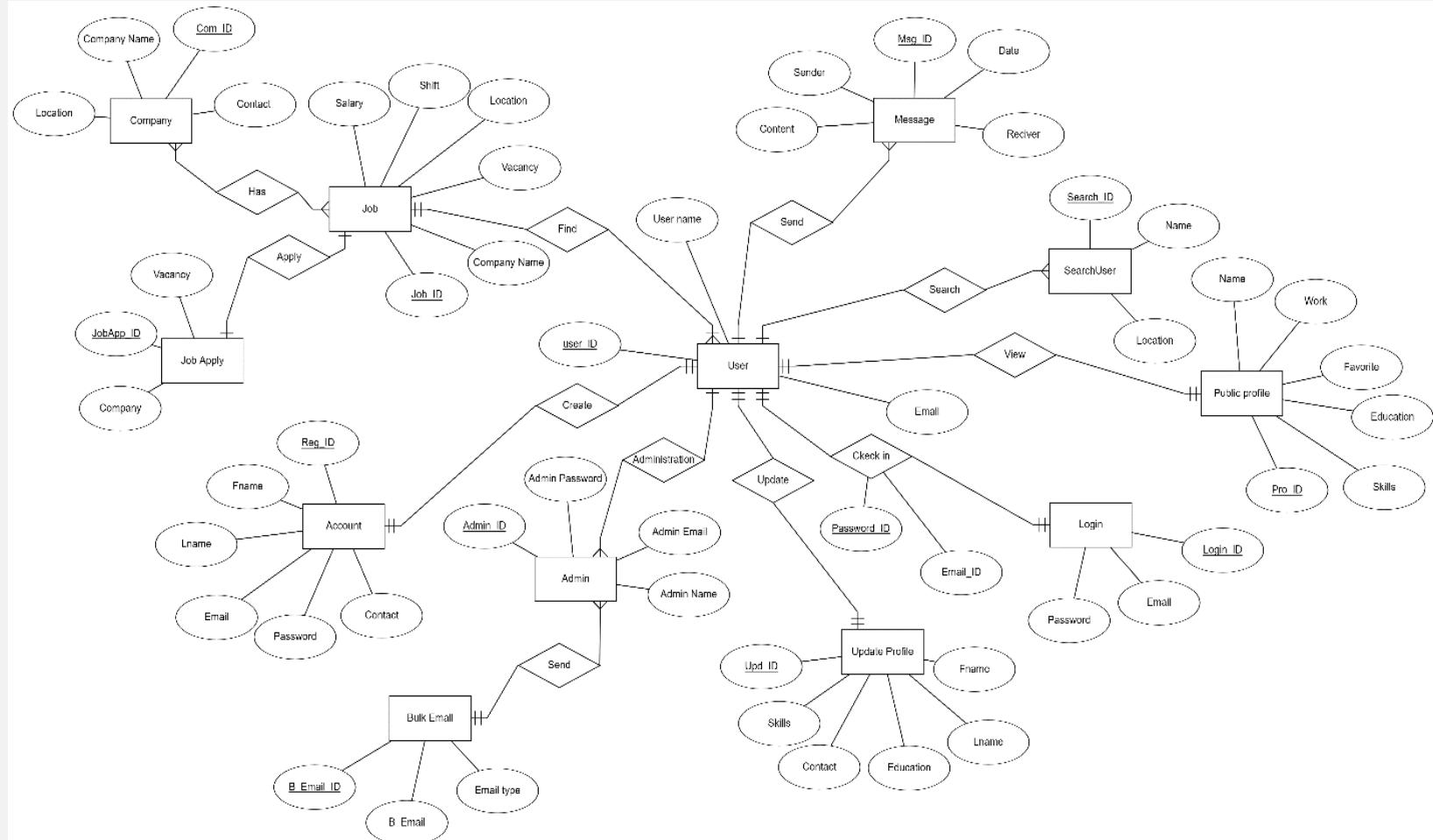
❑ Business Classes

S. No.	Class Name	Description
1	LoginService	Service class to handle business logic for user login
2	RegisterService	Service class to handle business logic for user registering to the website
3	AdminService	Service class to handle business logic for administration system of the website
4	ForgotPasswordService	Service class to handle business logic for user password retrieval functionality
6	UpdateProfileService	Service class to handle business logic for user who want edit their profile page
7	EmailSenderService	Service class to handle business logic for sending email
8	SearchService	Service class to handle business logic for search user and view user profile page

9. Database Design

LITHAN

□ Paste the ER Diagram Developed in Module 4



Registration

Thank You



Thank you for Your Registration

We have sent email to name@gmail.com to confirmation.
Please confirm your email to activate your account

[Continue](#)

Create Account

Get started with your free account

[Login via Twitter](#)

[Login via facebook](#)

OR

Full name

b4_wanniarachige@gmail.c

Phone number

Select Gender

....

Repeat password

[Create An Account](#)

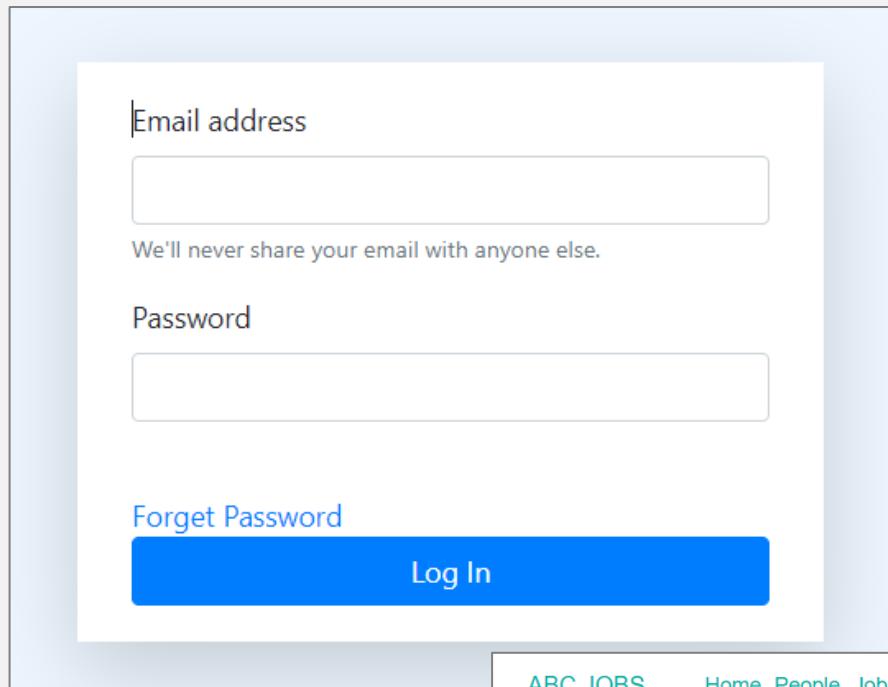
Have an account? [Log In](#)

© ABC JOBS. All right reserved

10. UI Design

LITHAN

Login



The login form is contained within a light gray rectangular box. It features two input fields: 'Email address' and 'Password', each with a placeholder text below it. Below the password field is a small note stating 'We'll never share your email with anyone else.' At the bottom of the form are two buttons: 'Forget Password' in blue and 'Log In' in white.

Email address

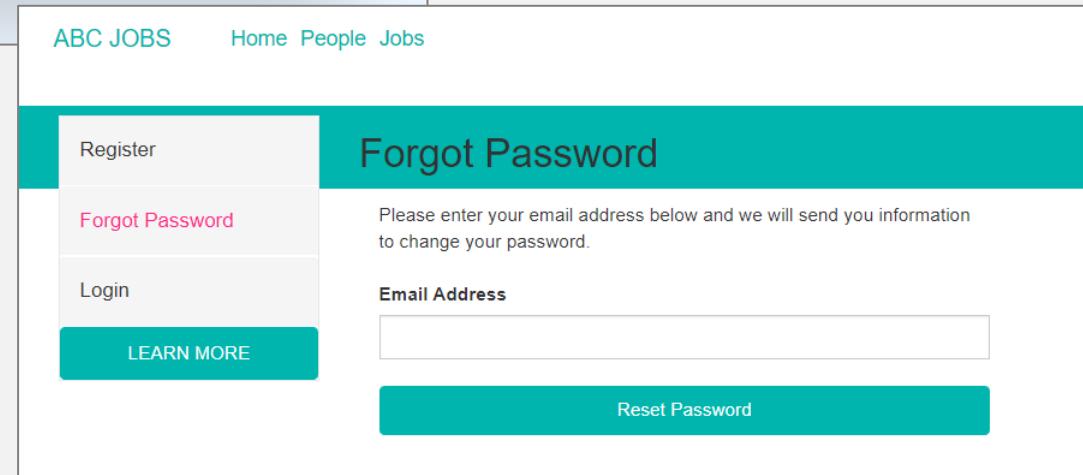
We'll never share your email with anyone else.

Password

Forget Password

Log In

Forgot Password



The forgot password form is part of a larger interface. At the top, there's a navigation bar with 'ABC JOBS' and links to 'Home', 'People', and 'Jobs'. A vertical sidebar on the left contains links for 'Register', 'Forgot Password' (which is highlighted in pink), and 'Login'. A large teal button labeled 'LEARN MORE' is located at the bottom of this sidebar. The main content area has a teal header with the text 'Forgot Password'. Below the header, a message asks users to enter their email address to receive password change information. An 'Email Address' input field and a 'Reset Password' button are present at the bottom.

ABC JOBS Home People Jobs

Register

Forgot Password

Login

LEARN MORE

Forgot Password

Please enter your email address below and we will send you information to change your password.

Email Address

Reset Password

10. UI Design

LITHAN

Search User

Search User

Name	Position
Tiger Nixon	System Architect
Garrett Winters	Accountant
Ashton Cox	Junior Technical Author
Cedric Kelly	Senior Javascript Developer
Airi Satou	Accountant
Brielle Williamson	Integration Specialist

Place	Age	Connect
London	25	Connect
Sri Lanka	46	Connect
Tokyo	45	Connect
Edinburgh	30	Connect
Tokyo	28	Connect
New York	22	Connect

© ABC JOBS. All right reserved

Forgot Password Confirmation



Forgot Password Confirmation

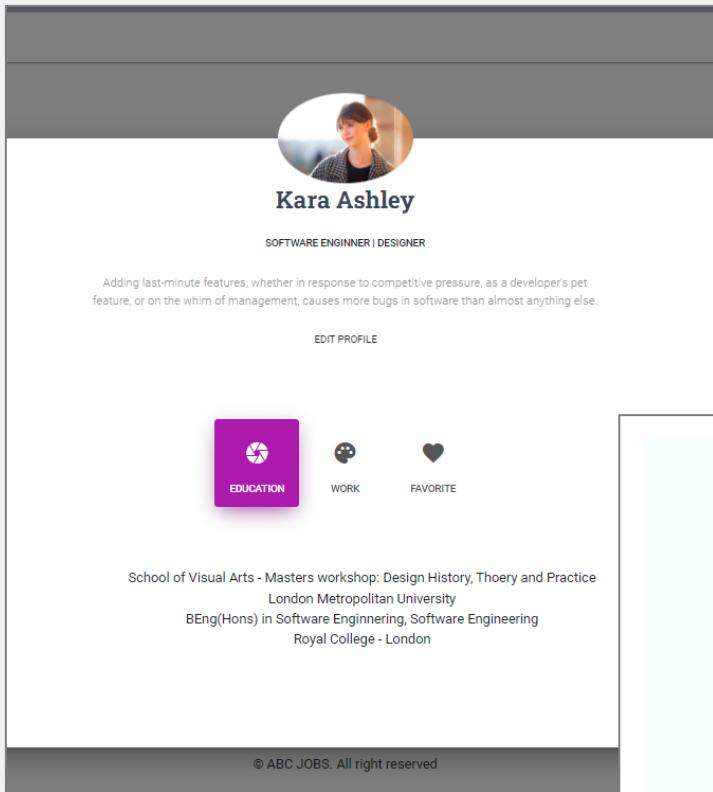
As per your request, you will receive an email with a temporary password. Use this password to login. Once you logged in you will be able to change your password.

[Continue](#)

10. UI Design

LITHAN

User Profile



A user profile card for Kara Ashley. It features a circular profile picture of a woman with short brown hair. Below the picture is the name "Kara Ashley" and the title "SOFTWARE ENGINEER | DESIGNER". A quote at the bottom reads: "Adding last-minute features, whether in response to competitive pressure, as a developer's pet feature, or on the whim of management, causes more bugs in software than almost anything else." At the bottom of the card are three icons: a purple square labeled "EDUCATION" with a graduation cap icon, a grey circle labeled "WORK" with a person icon, and a grey heart labeled "FAVORITE" with a heart icon.

Kara Ashley
SOFTWARE ENGINEER | DESIGNER

Adding last-minute features, whether in response to competitive pressure, as a developer's pet feature, or on the whim of management, causes more bugs in software than almost anything else.

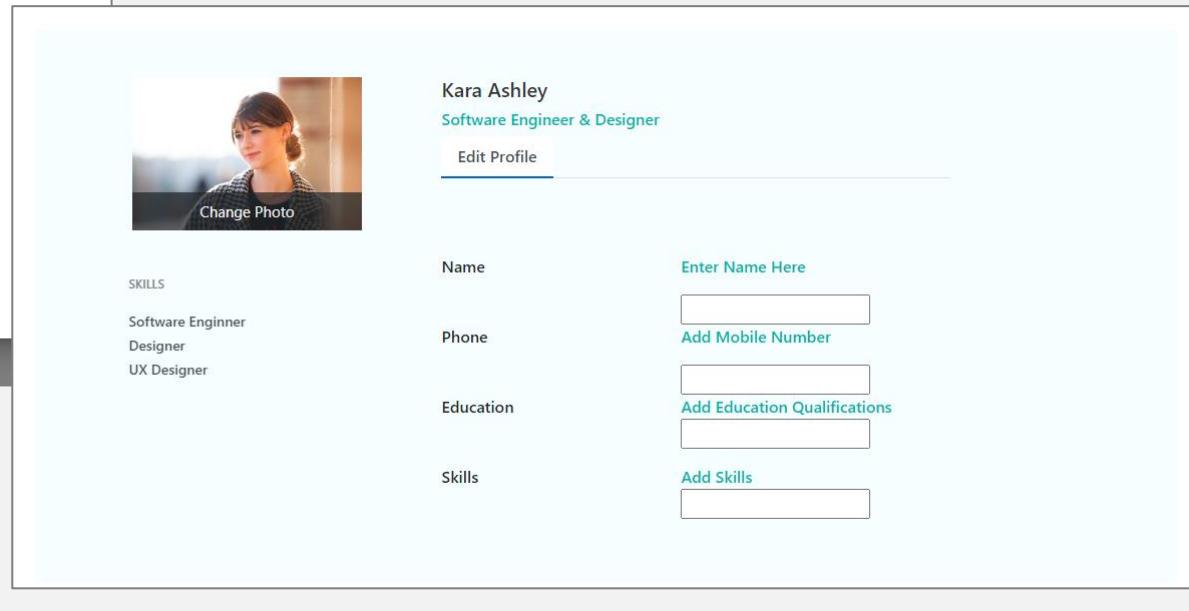
EDIT PROFILE

EDUCATION WORK FAVORITE

School of Visual Arts - Masters workshop: Design History, Theory and Practice
London Metropolitan University
BEng(Hons) in Software Engineering, Software Engineering
Royal College - London

© ABC JOBS. All right reserved

Edit Profile



The edit profile form for Kara Ashley. It shows her current profile picture and the option to "Change Photo". Below the photo, her name is displayed as "Kara Ashley" and her title as "Software Engineer & Designer". There is a link to "Edit Profile". The form is divided into several sections: "SKILLS" (listing "Software Engineer", "Designer", and "UX Designer"), "Name" (with an input field "Enter Name Here" and a "Save" button), "Phone" (with an input field "Add Mobile Number" and a "Save" button), "Education" (with an input field "Add Education Qualifications" and a "Save" button), and "Skills" (with an input field "Add Skills" and a "Save" button).

Kara Ashley
Software Engineer & Designer

Edit Profile

SKILLS

Software Engineer
Designer
UX Designer

Name

Enter Name Here

Phone

Add Mobile Number

Education

Add Education Qualifications

Skills

Add Skills

11. Spring Frameworks

LITHAN

□ Java spring includes

S. No.	Include	Purpose of Inclue
01	Spring-Core	Spring allows you to create apps out of "plain old java objects" and non-invasively deploy corporate services to POJOs
02	Spring-context	Responsible for instantiating, configuring, and assembling beans by reading configuration metadata from XML, Java annotations, and/or Java code in the configuration files
03	Spring-beans	Beans make up the applications structure. It's a just object that IoC containers instantiates
04	Spring-jdbc	This helps to connect the database and run queries
05	Spring-webmvc	Model view controller architecture and ready components for developing flexible applications provided

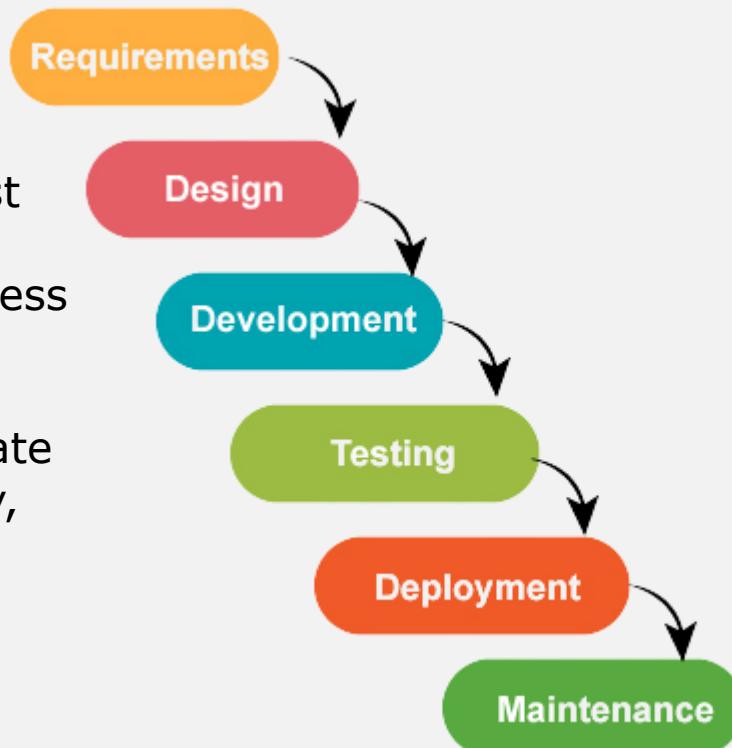
12. Software Development Methodology

LITHAN

- Software Development Methodology being used

Waterfall Model

The waterfall method turned into the first SDLC version to be used broadly in Software Engineering to ensure the success of the assignment. In "The Waterfall" technique, the entire system of software program development is split into separate stages. in this Waterfall model, generally, the outcome of one segment acts as the input for the subsequent section sequentially



13. Project Plan

LITHAN

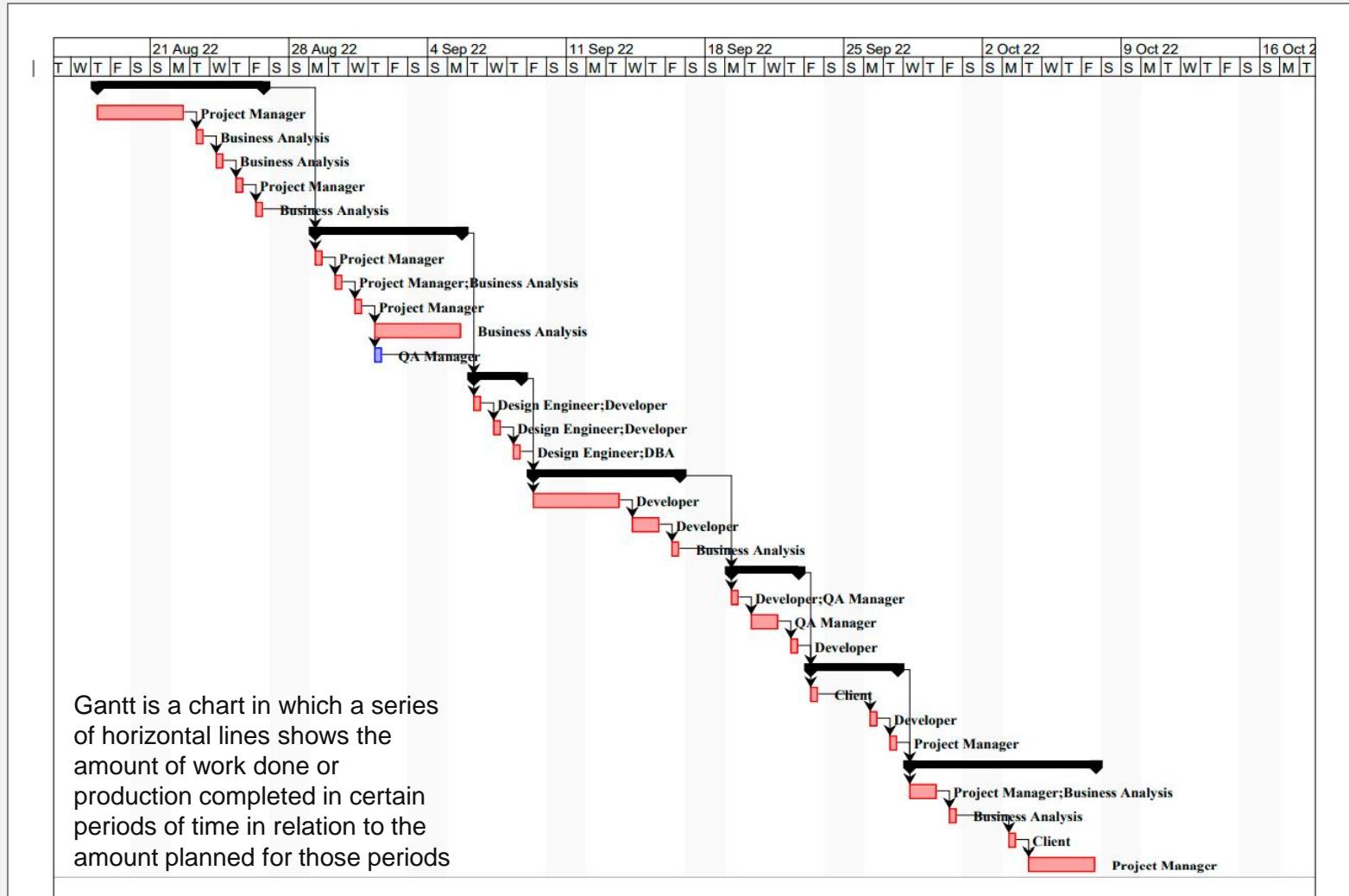
Project Plan

		Name	Duration	Start	Finish	Predecessors	Resource Names	M
1		Requirement Gathering	7 days	8/18/22 8:00 AM	8/26/22 5:00 PM			
2		Kickoff Meetings	3 days	8/18/22 8:00 AM	8/22/22 5:00 PM		Project Manager	
3		Identify Stakeholders	1 day	8/23/22 8:00 AM	8/23/22 5:00 PM	2	Business Analysis	
4		Gather and Documentation	1 day	8/24/22 8:00 AM	8/24/22 5:00 PM	3	Business Analysis	
5		Set the Objective	1 day	8/25/22 8:00 AM	8/25/22 5:00 PM	4	Project Manager	
6		Prepare SRS Document	1 day	8/26/22 8:00 AM	8/26/22 5:00 PM	5	Business Analysis	
7		Planning	6 days	8/29/22 8:00 AM	9/5/22 5:00 PM	1		
8		Document the Scope	1 day	8/29/22 8:00 AM	8/29/22 5:00 PM	6	Project Manager	
9		Cost Estimation	1 day	8/30/22 8:00 AM	8/30/22 5:00 PM	8	Project Manager;Business Analysis	
10		Allocating Resources	1 day	8/31/22 8:00 AM	8/31/22 5:00 PM	9	Project Manager	
11		Risk Planning	3 days	9/1/22 8:00 AM	9/5/22 5:00 PM	10	Business Analysis	
12		Test Planning	1 day	9/1/22 8:00 AM	9/1/22 5:00 PM	10	QA Manager	
13		Design	3 days	9/6/22 8:00 AM	9/8/22 5:00 PM	7		
14		Prepare Technical Design	1 day	9/6/22 8:00 AM	9/6/22 5:00 PM	12	Design Engineer;Developer	
15		Prepare Prototype	1 day	9/7/22 8:00 AM	9/7/22 5:00 PM	14	Design Engineer;Developer	
16		Design Database	1 day	9/8/22 8:00 AM	9/8/22 5:00 PM	15	Design Engineer;DBA	
17		Implementation	6 days	9/9/22 8:00 AM	9/16/22 5:00 PM	13		
18		Coding the Source Code	3 days	9/9/22 8:00 AM	9/13/22 5:00 PM	16	Developer	
19		Implement the database	2 days	9/14/22 8:00 AM	9/15/22 5:00 PM	18	Developer	
20		Prepare Business Logics	1 day	9/16/22 8:00 AM	9/16/22 5:00 PM	19	Business Analysis	
21		Testing	4 days	9/19/22 8:00 AM	9/22/22 5:00 PM	17		
22		Conduct Unit Testing	1 day	9/19/22 8:00 AM	9/19/22 5:00 PM	20	Developer;QA Manager	
23		Conduct Integration Test	2 days	9/20/22 8:00 AM	9/21/22 5:00 PM	22	QA Manager	
24		Fix the errors	1 day	9/22/22 8:00 AM	9/22/22 5:00 PM	23	Developer	
25		Deployment	3 days	9/23/22 8:00 AM	9/27/22 5:00 PM	21		
26		UAT Testing	1 day	9/23/22 8:00 AM	9/23/22 5:00 PM	24	Client	
27		Fix errors	1 day	9/26/22 8:00 AM	9/26/22 5:00 PM	26	Developer	
28		Deploy the website	1 day	9/27/22 8:00 AM	9/27/22 5:00 PM	27	Project Manager	
29		Maintenance	8 days	9/28/22 8:00 AM	10/7/22 5:00 PM	25		
30		Release the user guide Document	2 days	9/28/22 8:00 AM	9/29/22 5:00 PM	28	Project Manager;Business Analysis	
31		Provide post deployment	1 day	9/30/22 8:00 AM	9/30/22 5:00 PM	30	Business Analysis	
32		Service level agreement	1 day	10/3/22 8:00 AM	10/3/22 5:00 PM	31	Client	
33		Complete Project	4 days	10/4/22 8:00 AM	10/7/22 5:00 PM	32	Project Manager	

13. Project Plan

LITHAN

□ Gantt Chart

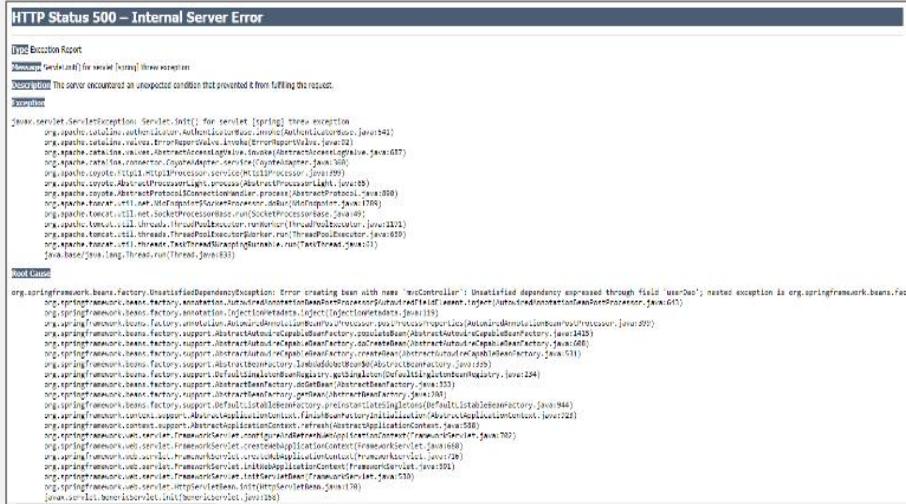


14. Application Screen Shots

LITHAN

- Screenshots demonstrates the error

Error 1



The screenshot shows an "HTTP Status 500 – Internal Server Error" page. The title bar says "HTTP Status 500 – Internal Server Error". Below it is a "Detailed Error Report" section with a "Description" and "Root Cause" link. The "Root Cause" section contains a very long stack trace starting with:

```
org.springframework.beans.factory.UnsatisfiedDependencyException: Error creating bean with name 'mcController': Unsatisfied dependency expressed through field 'userService'; nested exception is org.springframework.beans.factory.BeanCreationException: Error creating bean with name 'UserServiceImpl': Unsatisfied dependency expressed through field 'sessionFactory'; nested exception is org.hibernate.SessionException: Error creating session factory.
```

... (The stack trace continues for several pages of text)

Solution

- Look at HTTP Status
- Check out the Root Cause
- Look at the last sentence
- Unable to create requested service

[**org.hibernate.engine.jdbc.env.spi.JdbcEnvironment**]

- Likely caused by JDBC or repository interfaces
- Check persistence.xml
- Fixed
jdbc:mysql://localhost:5476 to
jdbc:mysql://localhost:3306

14. Application Screen Shots

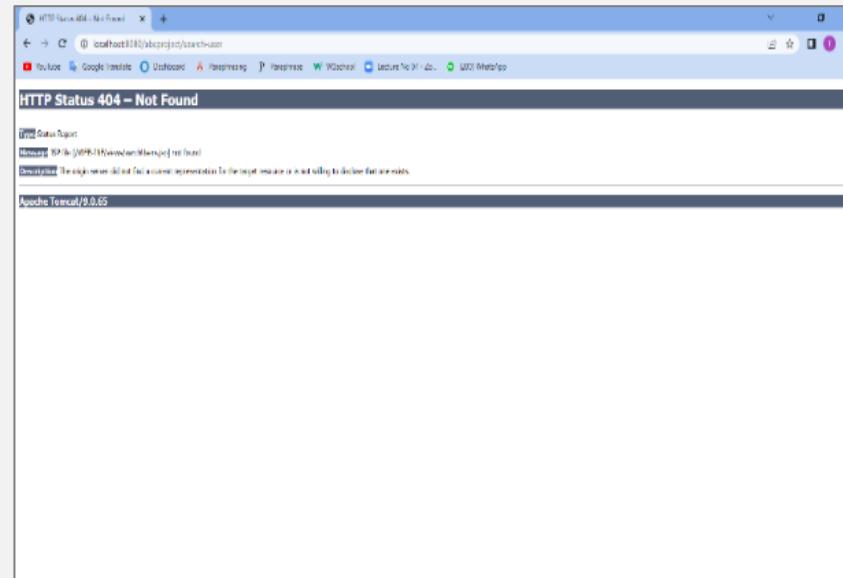
LITHAN

- Screenshots demonstrates the error

Error 2

Solution

- Look at HTTP Status
- Check out the Root Cause
- Look at the message
- JSP file [/WEB-INF/views/searchUsers.jsp] not found
- Check the MainController.java class
- Add return page to search method

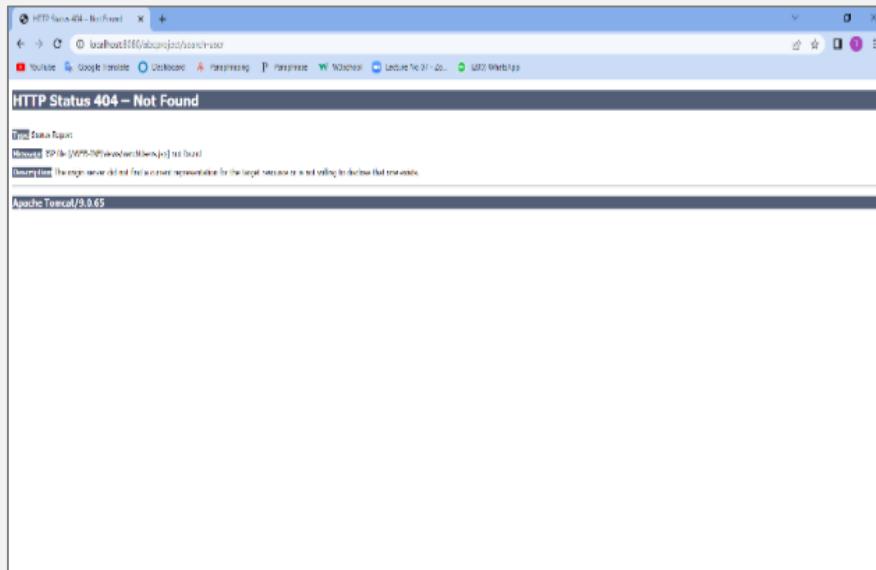


14. Application Screen Shots

LITHAN

- Screenshots demonstrates the error

Error



Solution

- Look at HTTP Status
- Check out the Root Cause
- Look at the description
- The origin server did not find a current representation for the target resource or is not willing to disclose that one exists.
- Check the MainController.java class
- Add request mapping for home.jsp

15. Project Milestones & Tasks

LITHAN

Project Task ID	Project Task Description	Project Milestone ID
1	Setup the Development Environment based on the Technical Development Environment	1
2	Power point Detailing the Business Process in the Proposed Application	1
3	Project Report detailing the Technical Design	1
4	Power Point Detailing the Project Plan	1
5	Develop the UI HTML standard pages (Based on Module 3 work)	2
6	Create common Struts Includes, Models, Views & Controllers	2
7	Develop Classes	2
8	Develop Struts Application	2

16. Project Milestones & Tasks

LITHAN

Project Task ID	Project Task Description	Project Milestone ID
9	Develop Test Cases	3
10	Develop Project Documentation	3
11	Develop User Manual	3

17. Milestone Feedback & Action taken

LITHAN

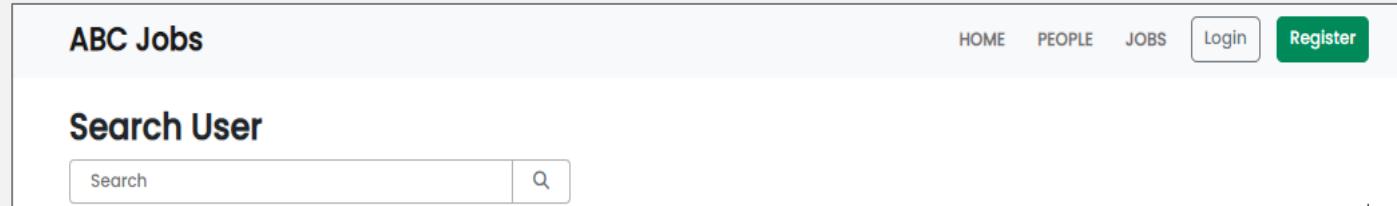
Project Milestone ID	Milestone Feedback received from Tutor / Learning Facilitator	Action Taken (Yes / No)
1	Add dependencies	Yes
	Download spring frameworks and add	Yes
2	Connect registration with data base	Yes
	Add user dashboard	Yes
3	If u can add bulk email system	yes

18. Modifications Made based On Feedback

LITHAN

I received feedback to add suggestions to the search user page. So I added account suggestions to the search user page

Before



18. Modifications Made based On Feedback

LITHAN

After

ABC Jobs

HOME PEOPLE JOBS MY PROFILE Logout Register

Search People Here

Search

	Chathushi	<input type="button" value="View Profile"/>
	chathudili77@gmail.com	
	0742269976	
	Rithu	<input type="button" value="View Profile"/>
	rithu@gmail.com	
	0553423456	
	Nethu	<input type="button" value="View Profile"/>
	nethu@gmail.com	
	0712345678	
	Chathu	<input type="button" value="View Profile"/>
	chathushi0707@gmail.com	
	0760528233	

19. Project Results

LITHAN

Index Page

ABC Jobs

Login Register



Welcome to your Professional Community

The Easiest Way to Get Yours New Job Find Jobs, Receive Instant Job Alerts, Apply Online. Employers Post jobs for free!

Let the right people know you're open to work

With the Open To Work feature, you privately tell recruiters or publicly share with the ABC Community that you are looking for new job opportunities.

[View More](#)



Services

ABC JOBS

Marketing and salespeople frequently express their love/hate relationship with LinkedIn. Do you think the site as a form of social stalking for businesspeople? There are various powerful ABC features that are very helpful in your regular problems. If you simply use the site to search up a prospect's credentials or to check a connection request email, you're missing out on important possibilities to further your career, increase your sales leads, and, ultimately, expand your business.

[Get a Quote](#)

Find the right job

Once your profile is built out, you can integrate a few strategic features that ABC offers specifically to job seekers to boost your visibility and help with targeting the right opportunities

Post your Job for million of people

With ABC Jobs, you'll get easy-to-use tools to help you find the right hire quickly. You'll be able to reach and engage with a community of millions of job seekers who visit ABC every week.

Connect with people

Building your ABC network is a great way to stay in touch with alumni, colleagues, and recruiters, as well as connect with new, professional opportunities

ABOUT US

ABC is an Sri Lanka business and employment-oriented online service that operates via websites and mobile apps. Launched on July 7, 2022, the platform is primarily used for professional networking and career development, and allows job seekers to post their CVs and employers to post jobs.

OUR SERVICES

[Web Design](#)
[Web Development](#)
[Marketing](#)
[Product Management](#)
[Offer Jobs](#)

USEFUL LINKS

[Home](#)
[About Us](#)
[Services](#)
[Term of Service](#)
[Privacy Policy](#)

CONTACT

23/B
Virtusa Road
Colombo 8
abcjobs36@example.com
+ 94 762369976
+ 94 760528233

[Facebook](#) [Twitter](#) [Instagram](#) [LinkedIn](#)

© 2021 Copyright: All Rights Reserved by ABC JOBS

19. Project Results

LITHAN

Registration

ABC Jobs

[Login](#) [Register](#)

Sign up

Name

Email

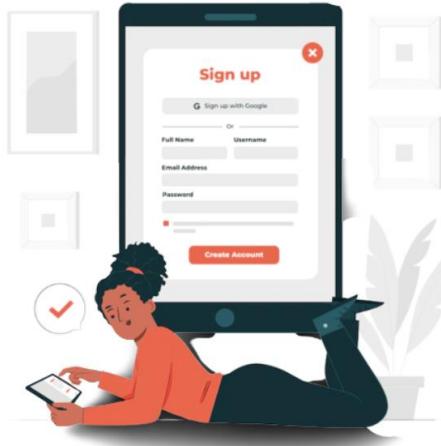
Phone

City

Password

Repeat Password

[Register](#)



[!\[\]\(9e495a407410ff9660147aa2d767824e_img.jpg\) ABOUT US](#)

ABC is an Sri Lanka business and employment-oriented online service that operates via websites and mobile apps. Launched on July 7, 2022, the platform is primarily used for professional networking and career development, and allows job seekers to post their CVs and employers to post jobs.

[OUR SERVICES](#)

Web Design
Web Development
Marketing
Product Management
Offer Jobs

[USEFUL LINKS](#)

Home
About Us
Services
Term of Service
Privacy Policy

[CONTACT](#)

23/B
Virtusa Road
Colombo 8
abcjobs36@example.com
+ 94 762269976
+ 94 760528233

[!\[\]\(ee6aac46c359c56a431a119462132dea_img.jpg\)](#) [!\[\]\(e98d528b84701acd70f2b3b93d5632da_img.jpg\)](#) [!\[\]\(d668ffb47898e8080f1923525414e180_img.jpg\)](#) [!\[\]\(57d5956976e1466de8b8c9ab18afc9a5_img.jpg\)](#)

© 2021 Copyright: All Rights Reserved by ABC JOBS

19. Project Results

LITHAN

Thank you

ABC Jobs

Login Register

Thank You For Registration

Your account has been created successfully
A verification email has been sent to your registered email address.
Please click on the verification link included in the email to activate your account

CONTINUE

ABOUT US ABC is an Sri Lanka business and employment-oriented online service that operates via websites and mobile apps. Launched on July 7, 2022, the platform is primarily used for professional networking and career development, and allows job seekers to post their CVs and employers to post jobs.	OUR SERVICES Web Design Web Development Marketing Product Management Offer Jobs	USEFUL LINKS Home About Us Services Term of Service Privacy Policy	CONTACT 23/B Virtusa Road Colombo 8 abcjobs36@example.com + 94 762269976 + 94 760528233
--	---	--	--

[f](#) [t](#) [g](#) [in](#)

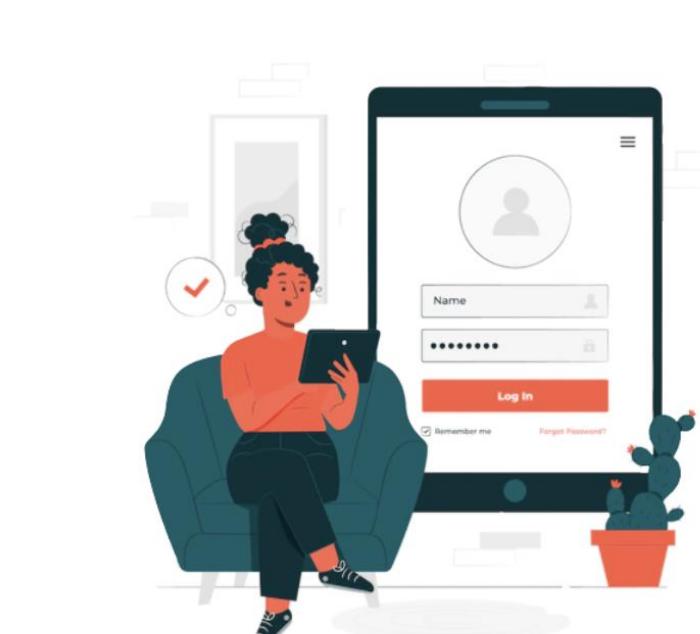
© 2021 Copyright: All Rights Reserved by ABC JOBS

19. Project Results

LITHAN

Login

ABC Jobs



Log in here

Email address

Password

Don't have an account? [Register](#)

[Forgot Password?](#)

Sign in

19. Project Results

LITHAN

Forgot Password

ABC Jobs

HOME PEOPLE JOBS Login Register

Forgot your password???????



Enter the email address associated with your account.
We will email you a link to reset your password

Reset Password

 **ABOUT US**

ABC is an Sri Lanka business and employment-oriented online service that operates via websites and mobile apps. Launched on July 7, 2022, the platform is primarily used for professional networking and career development, and allows job seekers to post their CVs and employers to post jobs.

 **OUR SERVICES**

- Web Design
- Web Development
- Marketing
- Product Management
- Offer Jobs

 **USEFUL LINKS**

- Home
- About Us
- Services
- Term of Service
- Privacy Policy

 **CONTACT**

- 23/B
Virtusa Road
Colombo 8
- abcjobs36@example.com
- + 94 762269976
- + 94 760528233

[f](#) [t](#) [g](#) [in](#)

© 2021 Copyright: All Rights Reserved by ABC JOBS

Forgot Password Confirmation

Forgot Password Confirmation



As per your request,
you will receive an email with a temporary password.
Use this password to login.
Once logged in, you will be able to change your password.

[Back To Home](#)

19. Project Results

LITHAN

User Dashboard

ABC Jobs

★ Manage My Work

#	All	Revenue
Connections	500+	
Events	250+	
Pages	1765	

My Skills

Default Primary Success Info Warning
Danger Success Ui Design Primary

Jobs

Senior Software Engineer (Python)
Virtusa
Sri Lanka (Remote)
Full Time
[More](#)

Full Stack Developer
Soul Studio
Maldives
[More](#)

Resort Housekeeper
Sun Communities & Sun Doors
45/C St.Francis Road, Colombo 7
[More](#)

Activity Social Media

Leo Ashley Posted - 3 hours ago



[Like](#) [Share](#) [Comment](#)

Leo Ashley Posted - 3 hours ago

Finished Project Management, Spring MVC, HTML Foundations

Jessica Wong - 3 hours ago

Congratulations

Kevin Wix - 3 hours ago

Congrats..Keep Going

Jessica Wong Posted - 5 hours ago



[Like](#) [Share](#) [Comment](#)

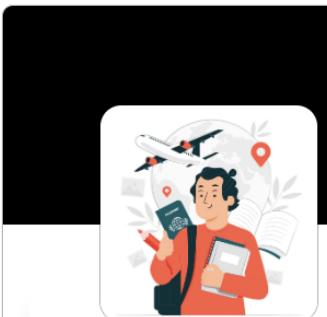
19. Project Results

LITHAN

User Profile Page

ABC Jobs

HOME PEOPLE JOBS MY PROFILE Logout Register



About

A Lead UX & UI designer based in Canada
I **design and develop** services for customers of all sizes, specializing in creating stylish, modern websites, web services and online stores. My passion is to design digital user experiences through the bold interface and meaningful interactions.

Chathu

Edit profile

User Id	6
Name	Chathu
Email	chathushi0707@gmail.com
Phone	0760528233
City	Bandarawela
Profession	Lead UX & UI Designer

19. Project Results

LITHAN

Search User Page

ABC Jobs

HOME PEOPLE JOBS MY PROFILE Logout Register

Search People Here

Search

	Chathushi	<input type="button" value="View Profile"/>
	chathudii77@gmail.com	
	0742269976	
	Rithu	<input type="button" value="View Profile"/>
	rithu@gmail.com	
	0553423456	
	Nethu	<input type="button" value="View Profile"/>
	nethu@gmail.com	
	0712345678	
	Chathu	<input type="button" value="View Profile"/>
	chathushi0707@gmail.com	
	0760528233	

19. Project Results

LITHAN

Admin Profile Page

ABC Jobs

ADMIN HOME PEOPLE JOBS Log-out

Hello Admin

Projects

- Server Migration: 37%
- Sales Tracking: 37%
- Customer Database: 37%
- Payout Details: 37%
- Account Setup: 37%

Modify

[Search Users](#)

View users

[view users](#)

Email

[Compose email](#)

Post Jobs

[Post Jobs](#)

EARNINGS (MONTHLY) \$40,000

EARNINGS (ANNUAL) \$215,000

TASKS 50%

PENDING REQUESTS 18

ABOUT US

ABC is an Sri Lanka business and employment-oriented online service that operates via websites and mobile apps. Launched on July 7, 2022, the platform is primarily used for professional networking and career development, and allows job seekers to post their CVs and employers to post jobs.

OUR SERVICES

- Web Design
- Web Development
- Marketing
- Product Management
- Offer Jobs

USEFUL LINKS

- Home
- About Us
- Services
- Term of Service
- Privacy Policy

CONTACT

- 23/B
Virtusa Road
Colombo 8
- abcjobs36@example.com
- + 94 762269976
- + 94 760528233

© 2021 Copyright: All Rights Reserved by ABC JOBS

19. Project Results

LITHAN

Bulk Email

ABC Jobs

ADMIN HOME PEOPLE JOBS Log-out

Admin Email

To:

Subject:

Compose Email

Attachment: Choose File No file chosen

Send Email

19. Project Results

LITHAN

View User (Admin)

ABC Jobs						
User id	User city	User email	User contact	User name	User password	Action
1		chathudil77@gmail.c	0742269976	Chathushi	12345	 
3		rithu@gmail.com	0553423456	Rithu	56789	 
5	Colombo	nethu@gmail.com	0712345678	Nethu	09876	 
6	Bandarawela	chathushi0707@gma	0760528233	Chathu	777777	 
12	Tokyo	aarvi@gmail.com	0764567893	Aarvi	56789	 
13	Kandy	silent@gmail.com	0745687546	Silent	1234567	 

20. Proposed Improvements

List of Improvements

- Creating more better code and not repeating that can be reused
- Consistent with the naming convention
- Implement other functionalities