ID: 23174

NAME: UMUGABA Martin Emmanuel

WEBTECH FINAL PROJECT

RAB nkunganire application System

[PART 1]

1. Describe key features of web application functional + non-functional requirements.

2. Web-UI with menu to access those key features (min 3 functional, and 2 non-functional).

Problems

When a farmer is going to ask for fertilizer from the government, there are many obstacles that cause the service to be delayed and he finds that the person in charge of the service does not see it. For example, when he went to get fertilizer and found that the fertilizer he wanted was not there, he lost production because he did not know that he did not get any information that there will be no fertilizer at that time.

What Web application solution do I propose to this problem?

- Easier access to information online: Visitors & registered former can browse fertilizer from agriculture in the area online without having to search for them on countless websites or offline.

- Registered former can apply online to fertilizer opportunities: Former can create accounts and be able to submit their application using forms which they submit to us and to those agricultural managers.

Key features of my web application including functional minimum (3) and non-functional requirements at least 2:

Functional Requirements

1. The system shall provide signup system for new former to register new accounts.

2. The system must provide login system for returning umurenge with accounts.

3. The system shall provide a login system for the admin where they shall add, delete, display internships to the database.

4. The system shall provide the admin the capacity to view all types of fertilizers applications.

5. The system shall allow registered former to display and choose a fertilizer and apply on the fertilizer of their choosing.

6. The system shall provide a browse option to display available fertilizer to website visitors.

Non-functional requirements

1. Security: former must have valid credentials to be authenticated.

2. Responsive UI: The web application is responsive to smaller screens and medium screens and can be used on different devices.

3. Portability – Web application shall be compatible with all modern browser who can load html and JavaScript and css.

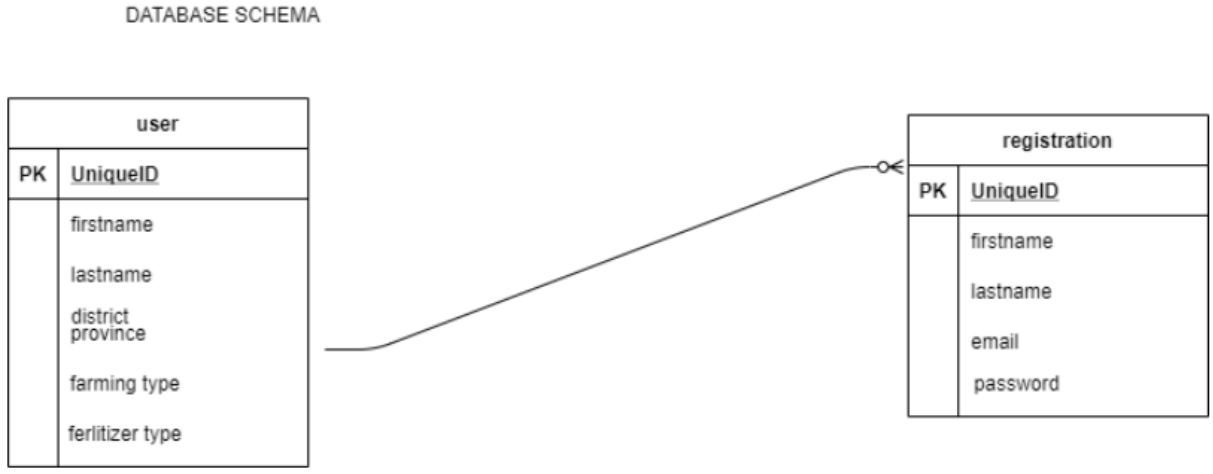
Source code:

```
package com.mycompany.mywebapp.com.mycompany.mywebapp.user;

import jakarta.persistence.*;

@Entity
@Table(name = "farmerrequest")
public class User {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
   private Integer id;
    @Column(nullable = false,unique = true,length = 45)
   private String District;
    @Column(length = 45,nullable = false)
   private String Province;
    @Column(length = 45,nullable = false )
  private String Firstname;
    @Column(length = 45,nullable = false )
  private  String Lastname;
    @Column(length = 45,nullable = false  )
  private String Fertilizer_type;
    @Column(length = 45,nullable = false )
  private String Farming type;

    public void setPassword(String rugambwa1234) {
    }

    public void setEmail(String mail) {
    }

    public void setFirstname(String ishimwe) {
    }

    public void setLastname(String patrick) {
    }
}
```

```java
package com.mycompany.mywebapp.com.mycompany.mywebapp.user;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import java.util.List;

@Controller
public class UserController {

@Autowired private Userservices service;
@GetMapping( "/users")

public String showUserList(Model model){

    List<User> listUsers = service.listAll();
    model.addAttribute("listUsers",listUsers);
    return "users";
}
@GetMapping("/users/new")
public String shownewFrom(Model model){

    model.addAttribute("user", new User());
    model.addAttribute("pageTitle" , "Add New user");
    return "user_form";
}
@PostMapping("/users/save")

    public String saveuser(User user , RedirectAttributes redirectAttributes
){
    service.save(user);
    RedirectAttributes message =
redirectAttributes.addFlashAttribute("message", "the user has been saved
successfully");
    return "redirect:/users";

}
@GetMapping("/users/edit/{id}")
public String showEditForm(@PathVariable("id") Integer id, Model model ,
RedirectAttributes redirectAttributes){
    try {
        User user =service.get(id);
        model.addAttribute("user",user);
        model.addAttribute("pageTitle" , "Edit user(ID: " + id +")");
                return "user_form";
```

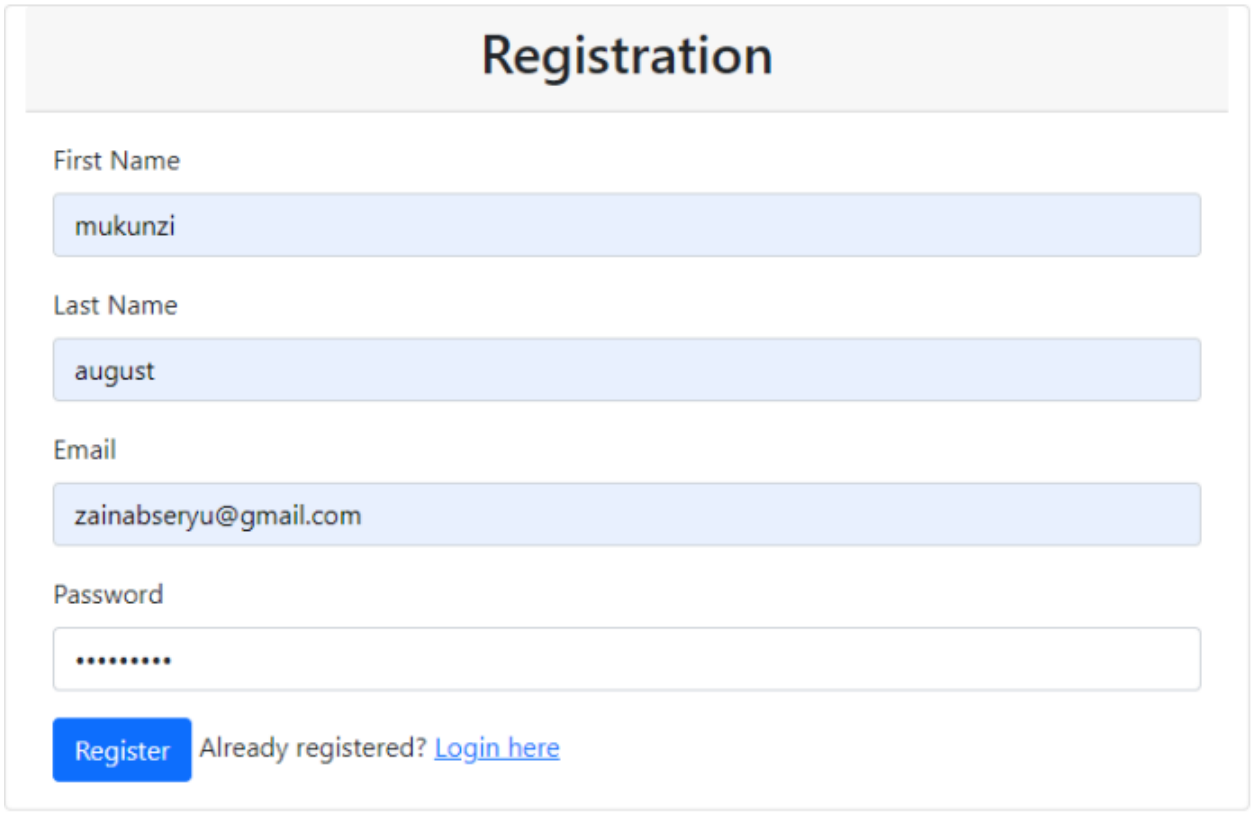
```
    } catch (UserNotFoundException e) {
        RedirectAttributes message =
redirectAttributes.addFlashAttribute("message", e.getMessage());
        return "redirect:/users";
    }

}



    @GetMapping("/users/delete/{id}")
    public String deleteUser(@PathVariable("id") Integer id  ,
RedirectAttributes redirectAttributes){
        try {
           service.delete(id);

            redirectAttributes.addFlashAttribute("message" , "the user id"
+  id  +   "has been deleted");


        } catch (UserNotFoundException e) {
            RedirectAttributes message =
redirectAttributes.addFlashAttribute("message", e.getMessage());

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

ERD entity relationship diagram



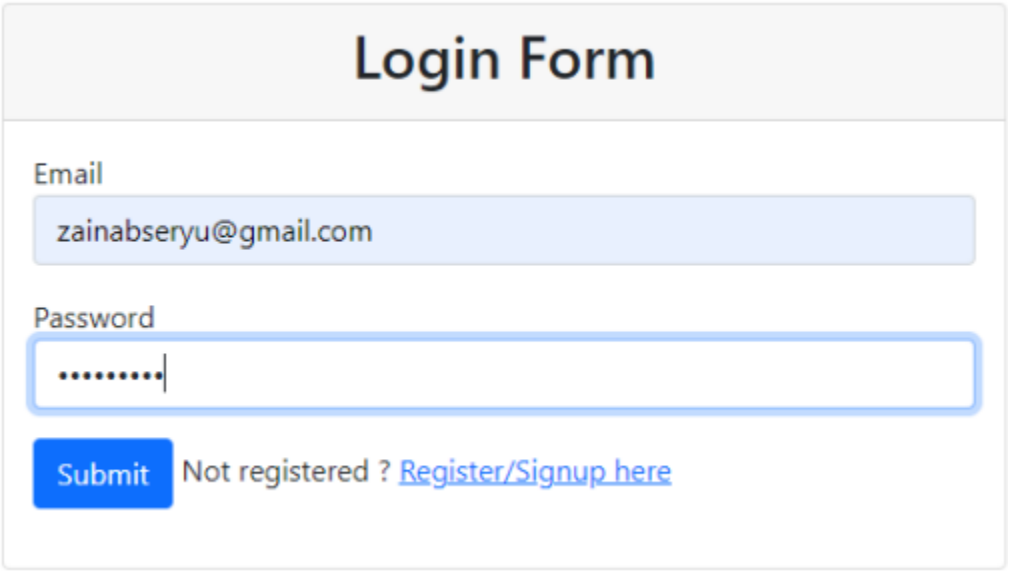
Screenshots/ workflow

1. The system shall provide signup system for new former to register new accounts.

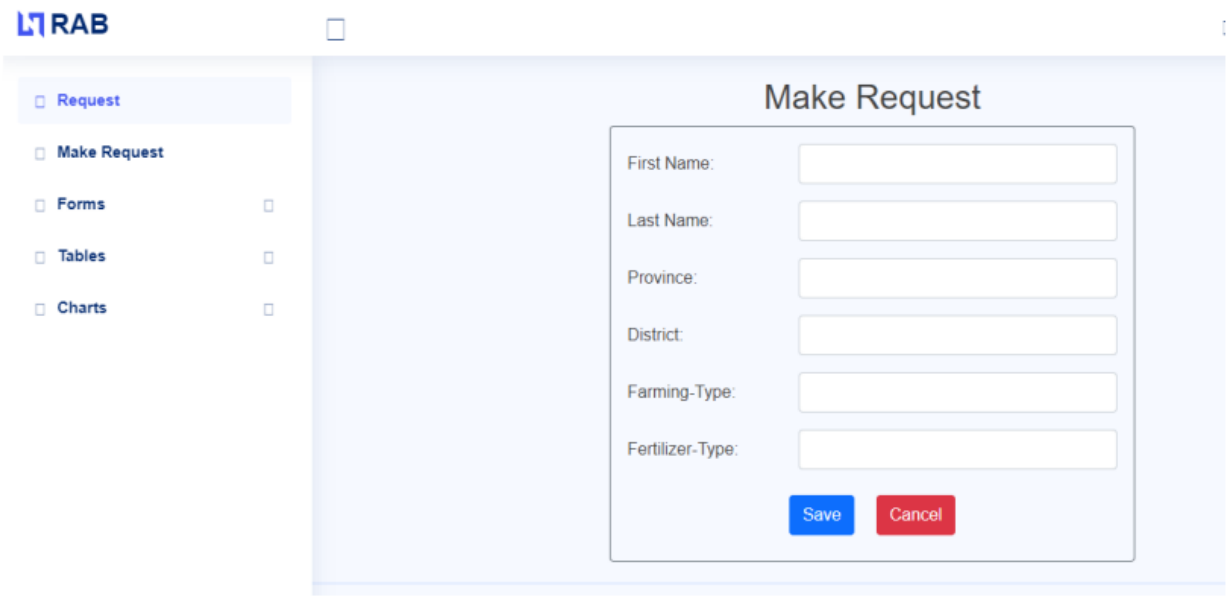


2. The system must provide login system for former with accounts.

Login form



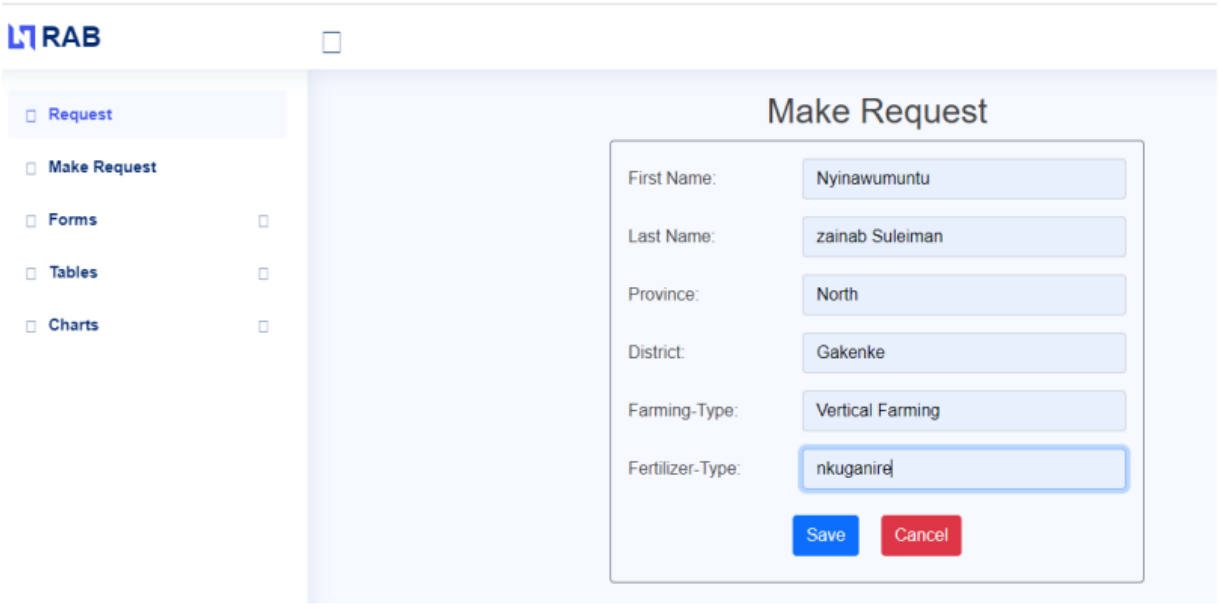
3. The system shall provide a login system for request



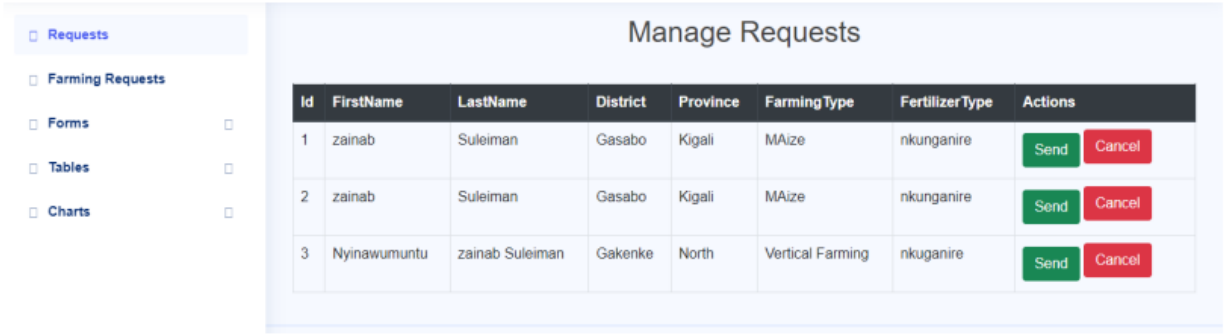
Farmers should make request

a. Insert

In this example I clicked on insert new internships in Machine Learning Category



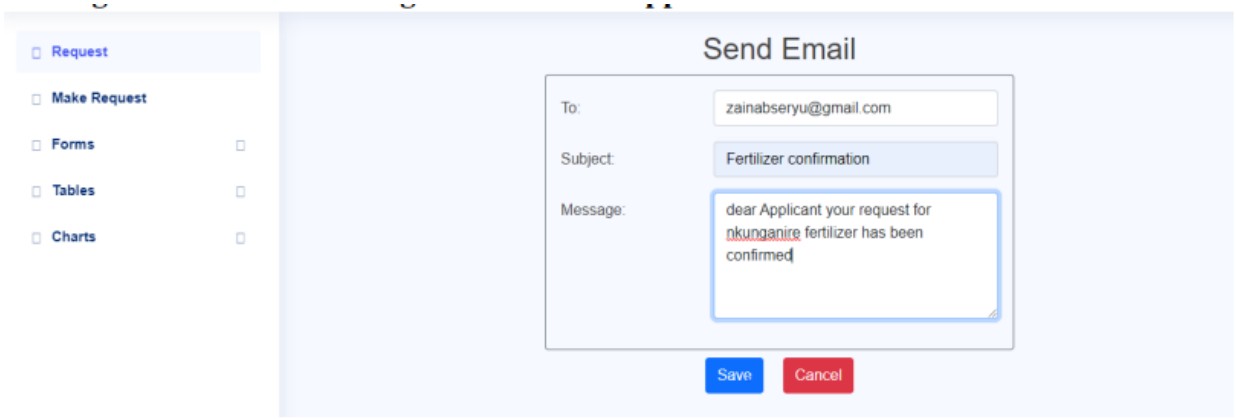
After inserting I clicked Display to see if it was inserted



4. The system shall provide the manger the capacity to view all request applications submissions from farmers

Manager you click on View request

Manager able to send massage to the former application want fertilizer

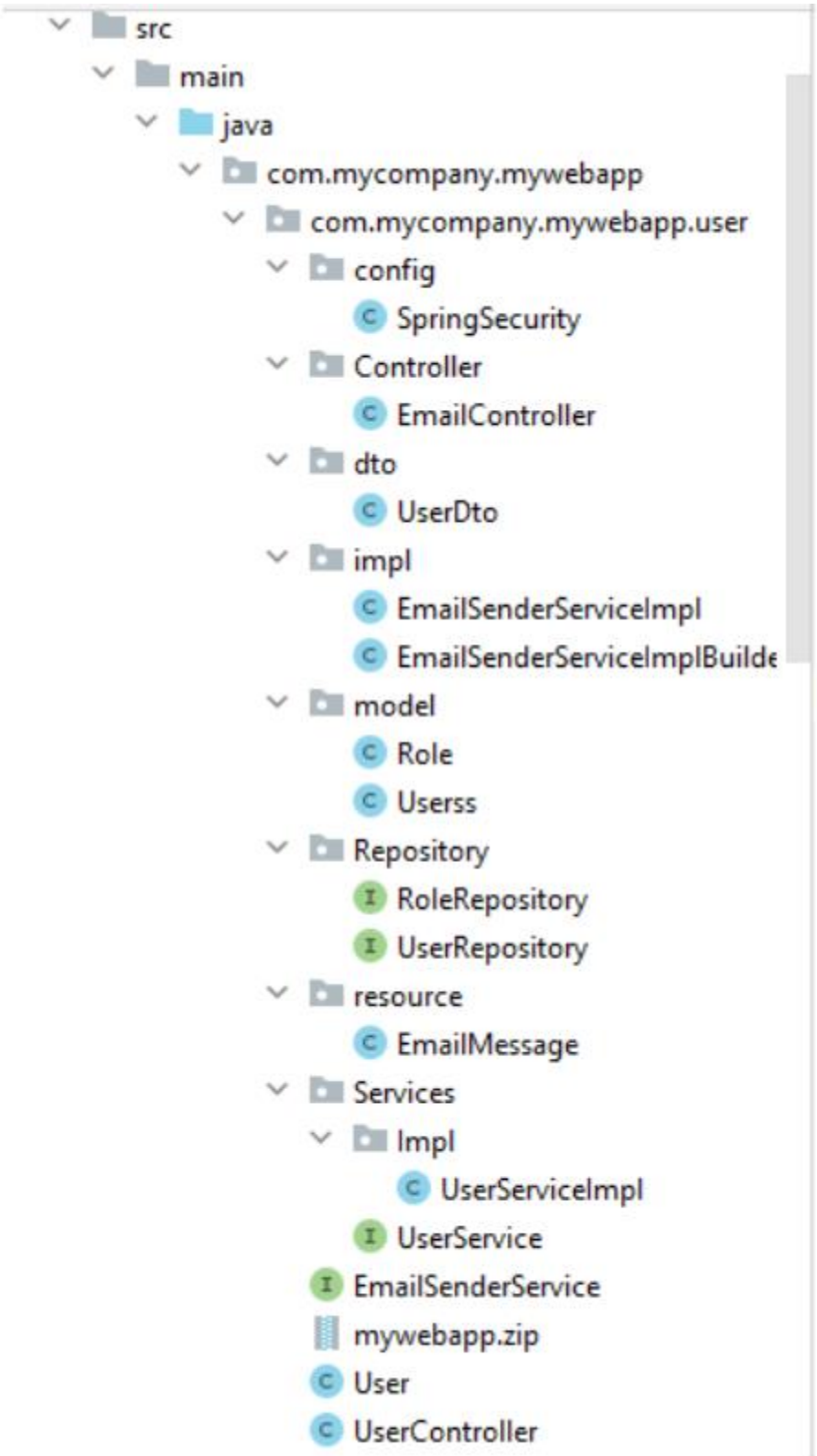


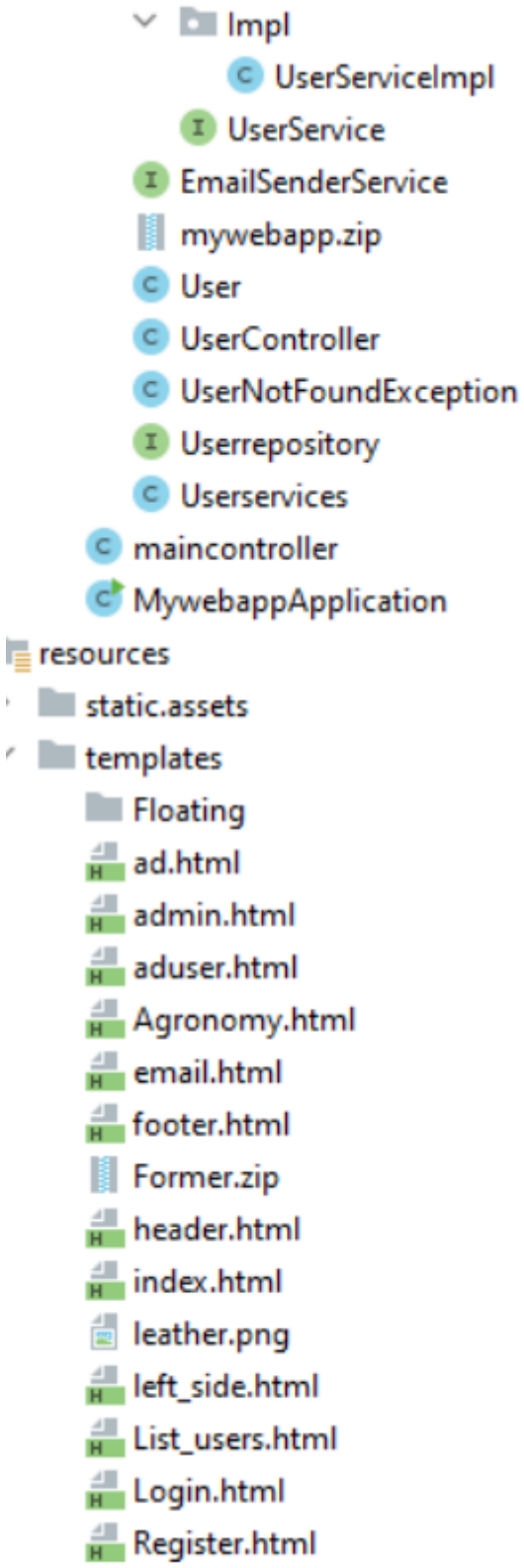
[PART 2]

Using MVC architecture implement each feature, and for each feature provide a screenshots demonstrating the implementation of Model, View and Controller components. Use Ajax at least in sending one request.

Click on resolve to import them

This project is using SPRING AND THERE IS IMPLEMENTATIONS OF MVC on all the above functional requirement


src
main
java
com.mycompany.mywebapp
com.mycompany.mywebapp.user
config
SpringSecurity
Controller
EmailController
dto
UserDto
impl
EmailSenderServiceImpl
EmailSenderServiceImplBuilde
model
Role
Userss
Repository
RoleRepository
UserRepository
resource
EmailMessage
Services
Impl
UserServiceImpl
UserService
EmailSenderService
mywebapp.zip
User
UserController


Impl
UserServiceImpl
UserService
EmailSenderService
mywebapp.zip
User
UserController
UserNotFoundException
Userrepository
Userservices
maincontroller
MywebappApplication
resources
static.assets
templates
Floating
ad.html
admin.html
aduser.html
Agronomy.html
email.html
footer.html
Former.zip
header.html
index.html
leather.png
left_side.html
List_users.html
Login.html
Register.html

Technical documentation: In this project I used the spring boot framework and spring initializr and interlj idea for programming part

https://github.com/Martin32323/webtechfinal.git