SERVER ALLOCATION APP

CONTENTS

APP DESCRIPTION

1. WHAT IS SERVER ALLOCATION APP?

- ***Server_Allocation app is based on the concept of Inventory Management, where user can request for particular resource for specific amount which will be automatically fulfilled after the SuperAdmin/Admin's Approval.***

- ***This Application will be useful for anyone who likes to know how Inventory Management works? and how was it going to help your work done!***

Let us see more brief in upcoming topics

Who is User?

1. User is a person who raises requests and needs to know whether his request's was responded back to him.(accepted/rejected/proceed later)

USER-CONTROLLER :-

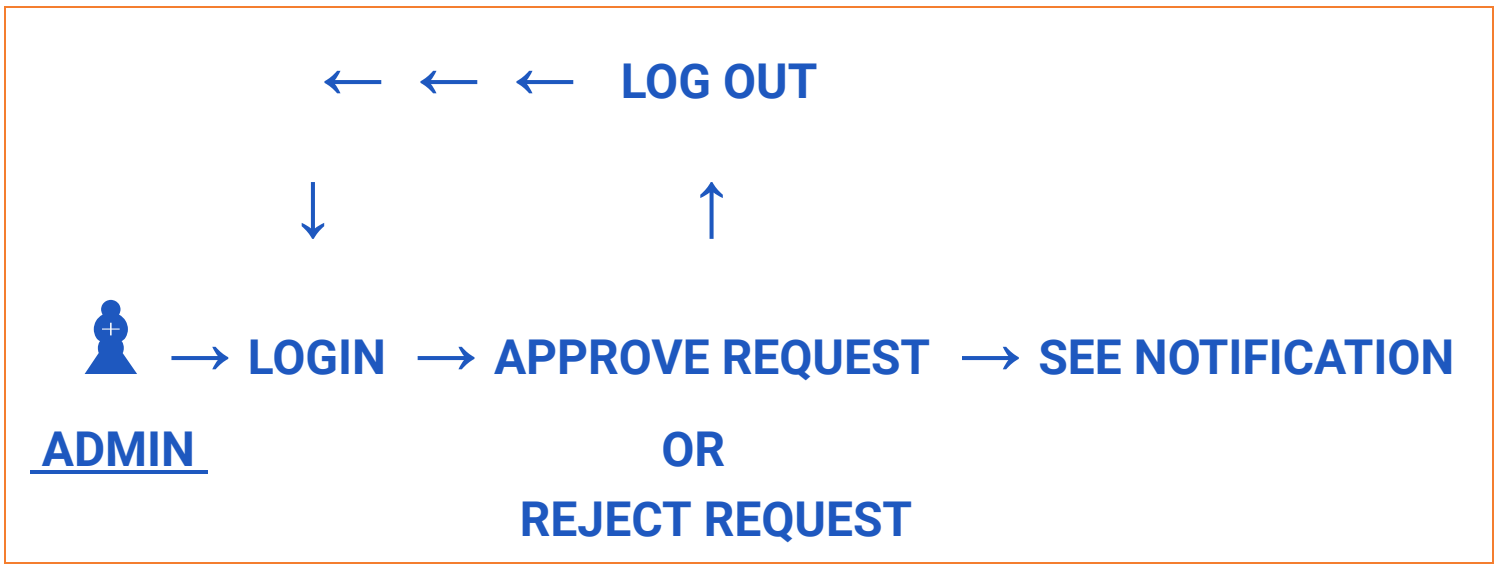
REQUEST :-

1. Request in this app means the amount of storage the user needs from the Other person (which must be in GB (gigabytes)).
2. Request can either be Approved or rejected by the higher person (Admin/SuperAdmin) but whatever the higher person does to specific user will be reflected to that user.

Who is Admin?

Admin is a person who can Approve or Reject the request raised by the users.

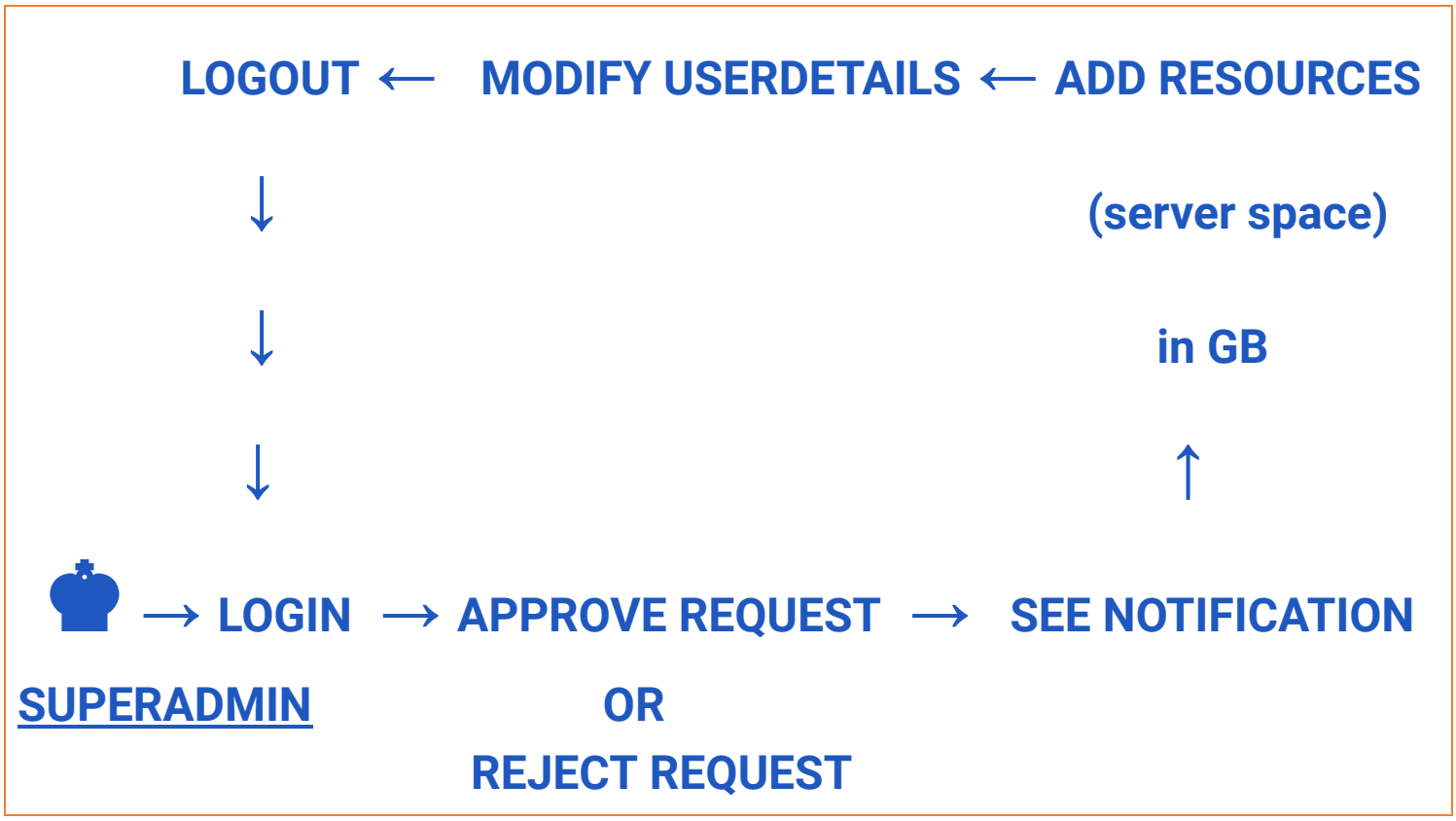
Admin-controller



Admin is the intermediate person between the user and the Super admin who has approve or reject the requests raised by the multiple user's.

Who is SUPERADMIN?

SuperAdmin is a person who can create,delete and update the user's status ,can Approve or reject the user's request and can add resources while he/she wanted to.



SuperAdmin is the person who have the main power's (As Queen in the chess) who can handle with user's request ,change user profile to admin or super-admin or vice-versa and can add resources to the server whenever they need to or whenever the server's resource went down below 10%.

HOW THE APP WORKS?

AUTHENTICATION :-

SignIn page :-

- if you are already a user, you can proceed with the sign-in page.

Signup page :-

- if you are new to this Application you should enter your presence here else you cannot explore this Application.
- After signup everyone is redirected to the user's page where they can request for the storage space.

APP APPLICATION'S :-

USER'S ROLE :-

- user can add requests for their needs,which may or may not be approved by either Admin nor Super-admin.

ADMIN'S role :-

- Admin can view all requests raised by the multiple users, which can be Approved or rejected by him based on the decisions he makes.
- Admin can also monitor the notification, which is alarmed when the server went below 10%.

SUPER-ADMIN'S ROLE :-

- Super-admin is the super power, he can add, delete or update users and he can also add, delete or update admins and super-admins too.
- Super-admin concentrates with the server resources when it went down below 10% they will be notified after seeing those message he will add new servers with different resource amount(in Gb).
- After the Super-Admin adding the resources to the server, the FILLED_PARTIALLY status will be fulfilled automatically to the users.

PAGES IN THE SERVER_APP

(Common pages not included - ONLY 60%(included))

USER PAGES :-

type	jsp pages	Servlet pages
USER	userDetails.jsp sorted_userDetails.jsp	-

ADMIN PAGES :-

type	jsp pages	Servlet pages
ADMIN	adminNotification.jsp sorted_adminNotification.jsp adminview.jsp sorted_adminview.jsp ProfileRejectInfo.jsp	ApproveOperations.java

SUPERADMIN PAGES :-

type	jsp pages	Servlet pages
SUPERADMINVIEW	delete.jsp editbook.jsp index.jsp Notification.jsp sorted_Notification.jsp superadminview.jsp sorted_superadminview.jsp resourceview.jsp ProfileRejectInfo.jsp	SuperApproveOperations.java AddResourceServlet.java NotifyTable.java

DISCRIPTION OF THE PAGES :-

- I had used with page without sorting feature and again used the page with sorted feature so I used two jsp pages with the same functions but differs in sorted and non-sorted manner.

  eg. Notification.jsp -> Non-sorted (All the requests will be displayed)
  sorted_Notification.jsp -> sorted (PENDING/REJECTED/APPROVED....)

- Approveoperations.jsp and SuperApproveOperations.jsp are two different pages but employed in same concepts where ApproveOperation.jsp will concentrate for pending requests and tries to fill the request from the server resources. whereas, SuperAdminOperation.jsp mostly concentrates on partially filled requests whenever the resource is added it checks for partially filled and tries to fill those requests first and then goes with the pending requests.

ADDITIONAL ENHANCEMENTS :-

- This app uses otp configuration for person who forgot the password and trying to login.
- For easy view and Action to be performed this app was fixed with sorting features.
- The search option used in finding username will find user-details easily.
- Notification will alert the super-admin and admin before giving Approval to the user requests which was pending.

AUTHENTICATION AFTER LOG-OUT :-

TO PREVENT BACK BUTTON AFTER LOG-OUT :-

```
<script>
  function preventBack(){window.history.forward();}
  setTimeout("preventBack()", 0);
  window.onunload=function(){null};
</script>
```

TO PREVENT RELOADING AFTER INSERTION OR LOG-OUT :-

```
response.setHeader("Cache-Control", "no-cache, no-store,
must-revalidate");
response.setHeader("Pragma", "no-cache");
response.setHeader("Expires", "0");
```

OTHER AUTHENTICATIONS :-

- To check for the User cookie, when it was available then we can allow the user to the main page else the request was blocked and the user was redirected to the Error-page.

- We should not allow the user to log in with the same browser but with a different account. If he/she tries it we should redirect them to the already logged in page.

- Filtering the values (inputs) given by the user before it pass to the database.

OTHER APPLICATIONS AND MODELS :-

OTHER APPLICATION :-

MAIL_FOR _OTP :-

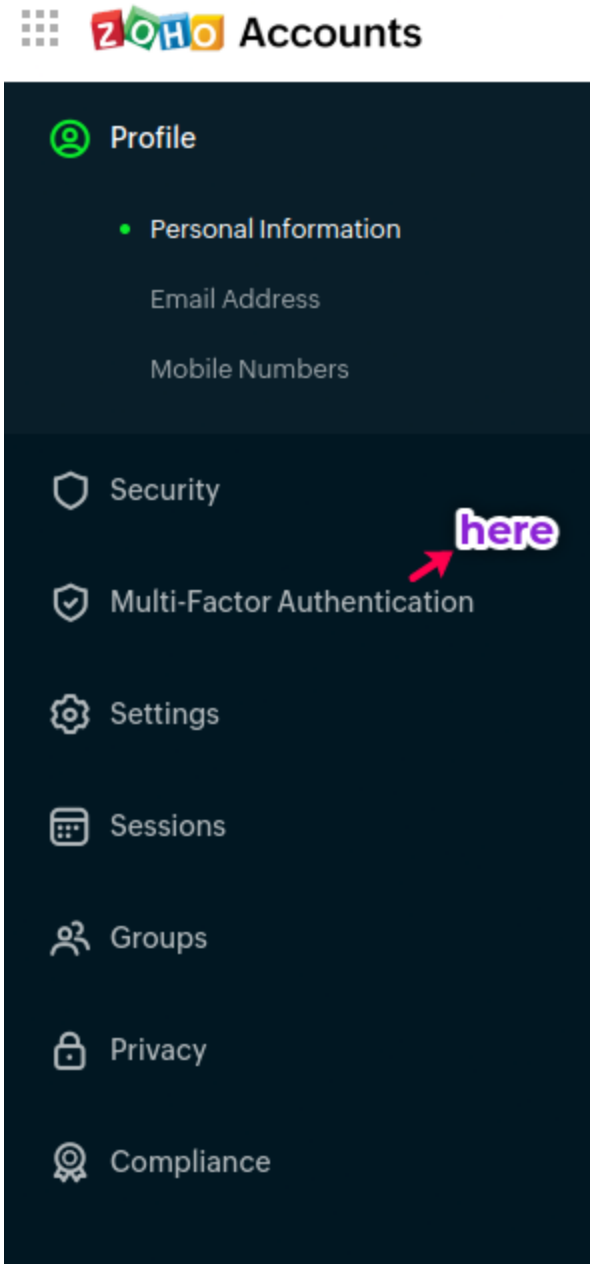
1. **Activation.jar**
2. **javax.mail.jar**

- **These jar files are required to stimulate the mail API which was commonly stored in SMTP format.**

STEPS FOR GETTING AUTHENTICATED FOR ZOHO MAIL :-

- Click on your Profile -> MY ACCOUNT -> MULTI-FACTOR-AUTHENTICATION
  -> ZOHOONE-AUTH

where you can give your device permission and generate a random password (note it) which can later be used as a password in your code.



- After that set up and adding jar files to your lib folder, you should follow the below code calling different methods and classes.(like Transport)

code :-

```
1 package common;
2
3 import javax.mail.*;
4 import javax.mail.internet.InternetAddress;
5 import javax.mail.internet.MimeMessage;
6 import java.util.Properties;
7
8 public class JavaSendEmail {
9     public static void sendMail(String recepient,String
  sub,String text){
10         System.out.println("Preparing to send Email");
11
12         Properties properties=new Properties();
13         properties.put("mail.smtp.auth","true");
14
  properties.put("mail.smtp.starttls.enable","true");
15
  properties.put("mail.smtp.host","smtp.zoho.com");
16         properties.put("mail.smtp.port","587");
17
18         final String
  username="gokulnath.ks@zohocorp.com";
19         final String password="BEWopdgeU1ajd";
  //here enter your's password
20
21         Session
  session=Session.getInstance(properties, new
  Authenticator() {
22             @Override
23             protected PasswordAuthentication
```

```
  getPasswordAuthentication() {
24                  return new
  PasswordAuthentication(username,password);
25              }
26          });
27
28          Message
  message=prepareMessage(session,username,recepient,sub,t
  ext);
29          try{
30              Transport.send(message);
31          }catch (Exception e){
32              e.printStackTrace();
33          }
34          System.out.println("Email sent Successfully
  to"+recepient);
35      }
36      private static Message prepareMessage(Session
  session,String username,String recepient,String
  sub,String text){
37          Message message=new MimeMessage(session);
38          try{
39              message.setFrom(new
  InternetAddress(username));
40
  message.setRecipient(Message.RecipientType.TO,new
  InternetAddress(recepient));
41              message.setSubject(sub);
42              message.setText(text);
43          }catch (Exception e){
44              e.printStackTrace();
```

```
45          }
46          return message;
47      }
48  }
49
```

TRY WITH RESOURCES :-

- It was one of the advancement of the try-catch method which was introduced java SE7.

- It implements AutoCloseable for following Class's Object declared inside the resource block.link (oracle document for try-with-resource)

It automatically closes the connection after it gets exit with the resource block with independent of the success or failure of the object created(Exception/without Exception it will be auto closed after the end of the block).

Eg :-

```
1       public HashMap<String,Object> validate(String s){
2
3               HashMap<String,Object> hMap=new HashMap<>();
4
5  try(
6               Connection con=DriverManager.getConnection(jdbcHosting,user,pass);
7               PreparedStatement pStatement=con.prepareStatement("select * from
   SERVER_MANAGEMENT where C_EMAIL=?");
8       ){
9       pStatement.setString(1,s);
10
11      ResultSet rSet=pStatement.executeQuery();
12
13              while(rSet.next()) {
14
15              hMap.put("id", rSet.getInt(1));
16              hMap.put("name", rSet.getString(2));
17              hMap.put("pass", rSet.getString(3));
18              hMap.put("email", rSet.getString(4));
19              hMap.put("utype", rSet.getInt(5));
20
21              }
22      }catch(Exception e){
23
24      }
25 return hMap;
26      }
```

ADVANTAGES OF COOKIES :-

- we can set particular time for cookie's life but we can't in session.
- Using sessions and handling with multiple servers will lead to session failure, which affects the value we tend to pass.
- Using sessions will affect the loading time of server.
- Cookies will prevent logging to multiple pages at same browser.
- Using Cookie's we can achieve user experience.(like how many times in a day,week... the user accessed the site)
- Cookie's reduces time than session.

Considering this reason, I prefer with Cookie's for Authentication and user operations.

USAGE OF POJO CLASS AND DAO PATTERNS :-

- I had Written getters setters for accessing the variables which is declared in private in order to maintain privacy for getting and setting the value to the pojo class.

- It is used for achieving code readability purposes and to avoid repeated code.

MAVEN :-

- Maven is a build tool used to convert files into certain structure,where we can add dependencies in pom.xml instead of adding jar files to the lib folder.

TOMCAT :-

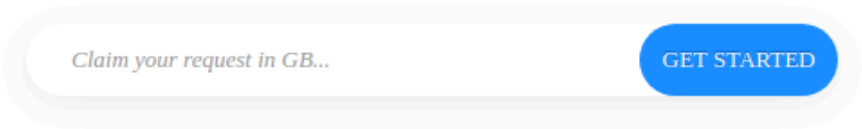
- Apache Tomcat is an Application server that acts as intermediate which used to receive requests from the client as http format and sends response to the client as a dynamic content getting from the servletApplication and Database.

MODELS :-

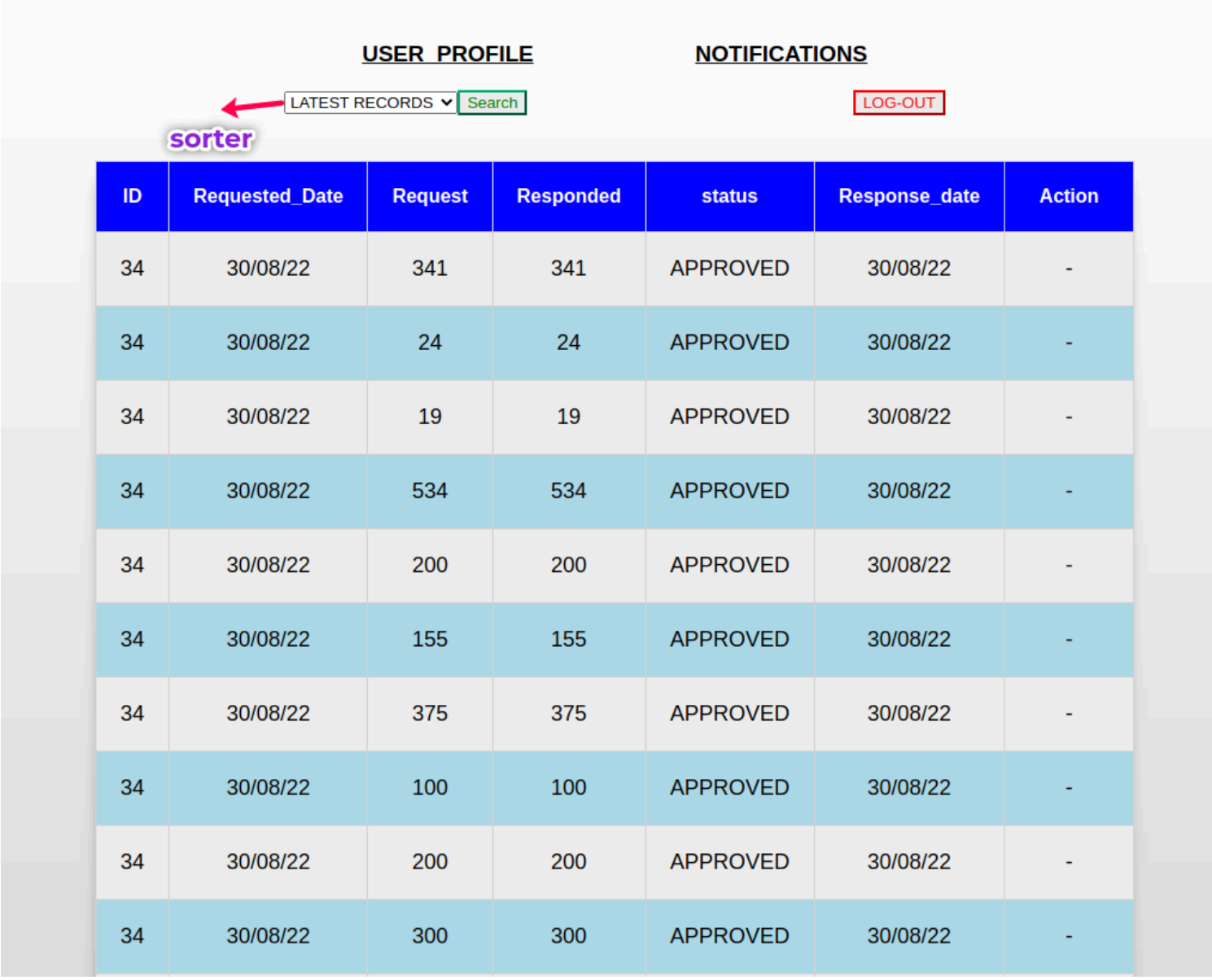
(only few pages are included to show an overview)

USER_MODELPAGES :-

History

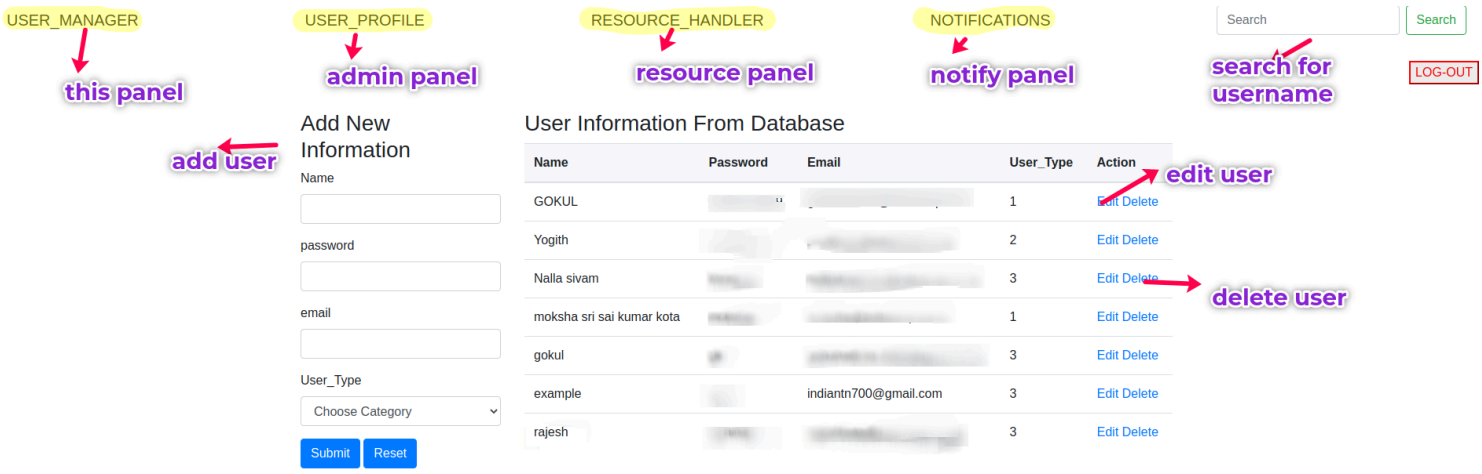


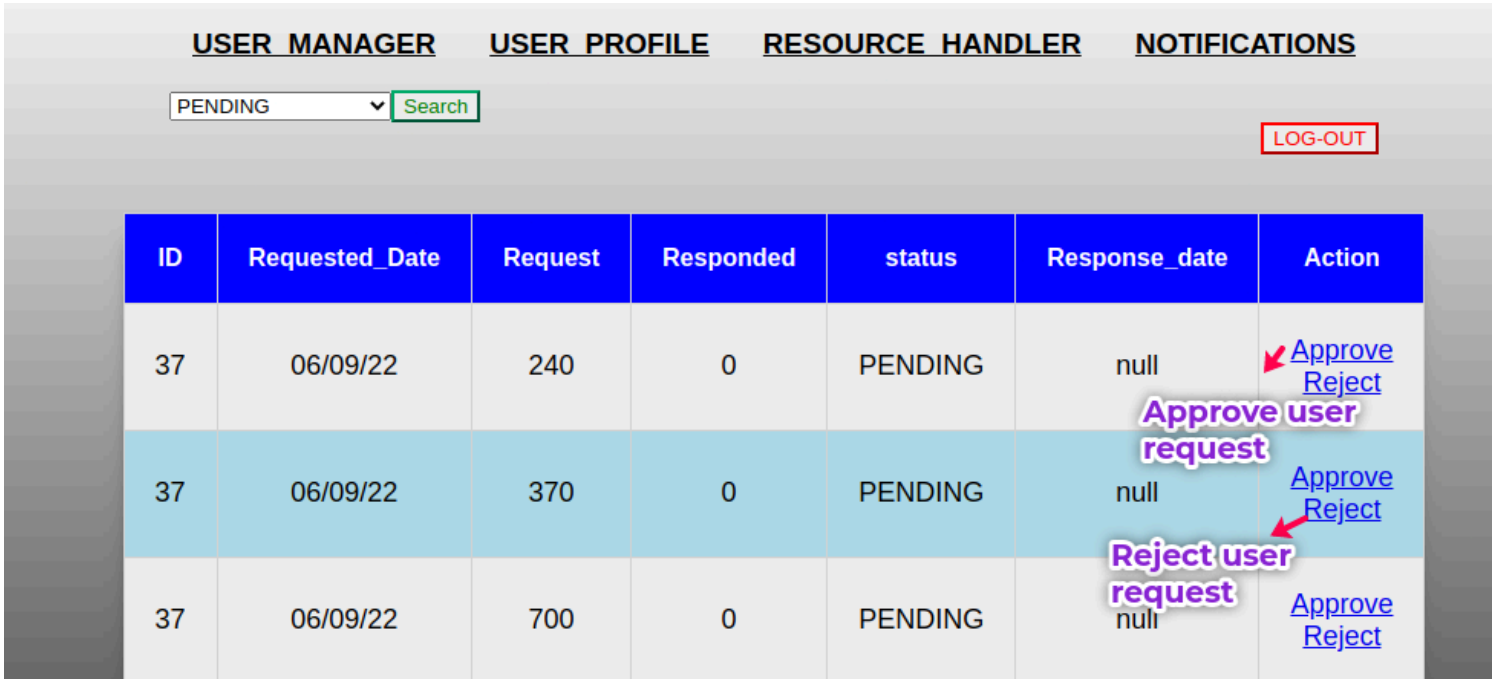
ADIMIN_MODELPAGE :-



ID	Requested_Date	Request	Responded	status	Response_date	Action
34	30/08/22	341	341	APPROVED	30/08/22	-
34	30/08/22	24	24	APPROVED	30/08/22	-
34	30/08/22	19	19	APPROVED	30/08/22	-
34	30/08/22	534	534	APPROVED	30/08/22	-
34	30/08/22	200	200	APPROVED	30/08/22	-
34	30/08/22	155	155	APPROVED	30/08/22	-
34	30/08/22	375	375	APPROVED	30/08/22	-
34	30/08/22	100	100	APPROVED	30/08/22	-
34	30/08/22	200	200	APPROVED	30/08/22	-
34	30/08/22	300	300	APPROVED	30/08/22	-

SUPERADMIN_MODELPAGES :-


USER_MANAGER
this panel
USER_PROFILE
admin panel
RESOURCE_HANDLER
resource panel
NOTIFICATIONS
notify panel
Search
Search
search for username
LOG-OUT
add user
Add New Information
Name
password
email
User_Type
Choose Category
Submit
Reset
User Information From Database
Name
Password
Email
User_Type
Action
GOKUL
1
Edit Delete
Yogith
2
Edit Delete
Nalla sivam
3
Edit Delete
moksha sri sai kumar kota
1
Edit Delete
gokul
3
Edit Delete
example
indiantn700@gmail.com
3
Edit Delete
rajesh
3
Edit Delete
edit user
delete user


USER_MANAGER
USER_PROFILE
RESOURCE_HANDLER
NOTIFICATIONS
PENDING
Search
LOG-OUT
ID
Requested_Date
Request
Responded
status
Response_date
Action
37
06/09/22
240
0
PENDING
null
Approve
Reject
37
06/09/22
370
0
PENDING
null
Approve
Reject
37
06/09/22
700
0
PENDING
null
Approve
Reject
Approve user request
Reject user request

USER_MANAGER USER_PROFILE RESOURCE_HANDLER NOTIFICATIONS

LOG-OUT

Add New Information

Name

Resource name

Value

Resource value

Submit Reset

User Information From Database

NAME	CAPACITY	REMAINING	User_Type	DATE
A	100	0	1	06/09/22
B	200	0	1	06/09/22
C	300	0	1	06/09/22
D	1000	0	1	06/09/22
E	500	0	1	07/09/22
Y	800	140	1	07/09/22
Z	200	200	1	07/09/22