



DBMS Project

Mid-Term Report

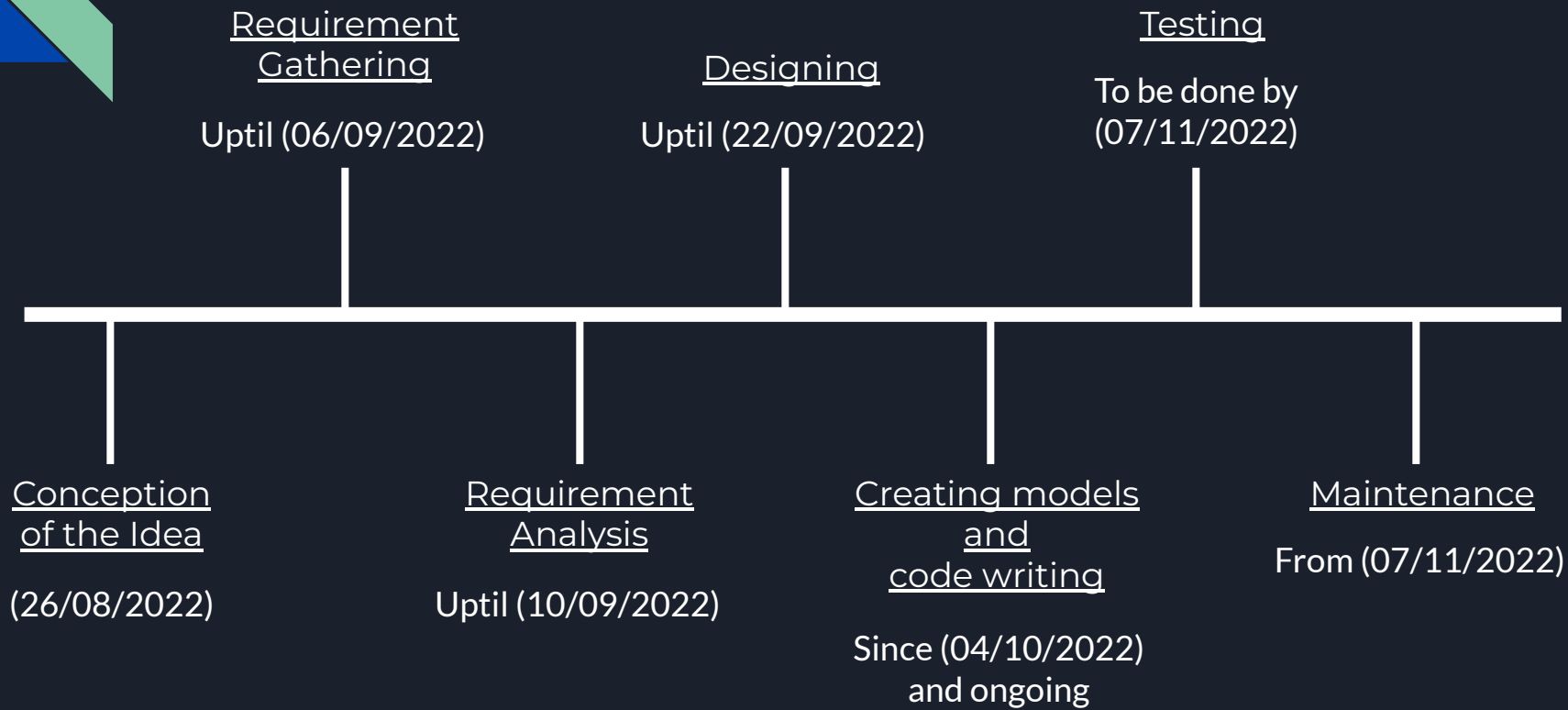
STeMS website

by

- Yagna Patel (20JE1102)
- Vaibhav Patel (20JE1053)
- Kathan Patel (20JE0669)
- Varun Balwani (20JE1061)

Project Timeline





Work Done till now





Conception of the Idea

To make a website which makes it easy for the students to get information and data from a single platform.

Also, providing teachers with a single platform to upload and update information and data.

Hence, STeMS (**S**tudent **T**eacher **M**anagement **S**ystem) website.



Requirement Gathering

Students requirements :

We need a website using which we can access our attendance, course materials, marks of various different subjects. We also want to have the facility of making a complaint regarding marks obtained, etc.


Teachers requirements :

We need a website using which we can upload attendance, course materials, marks of our own subjects. We also want to have the facility to upload notices which are visible to everyone in the institute. Students should not have the facility to upload any documents except for complaints.



Requirement Analysis

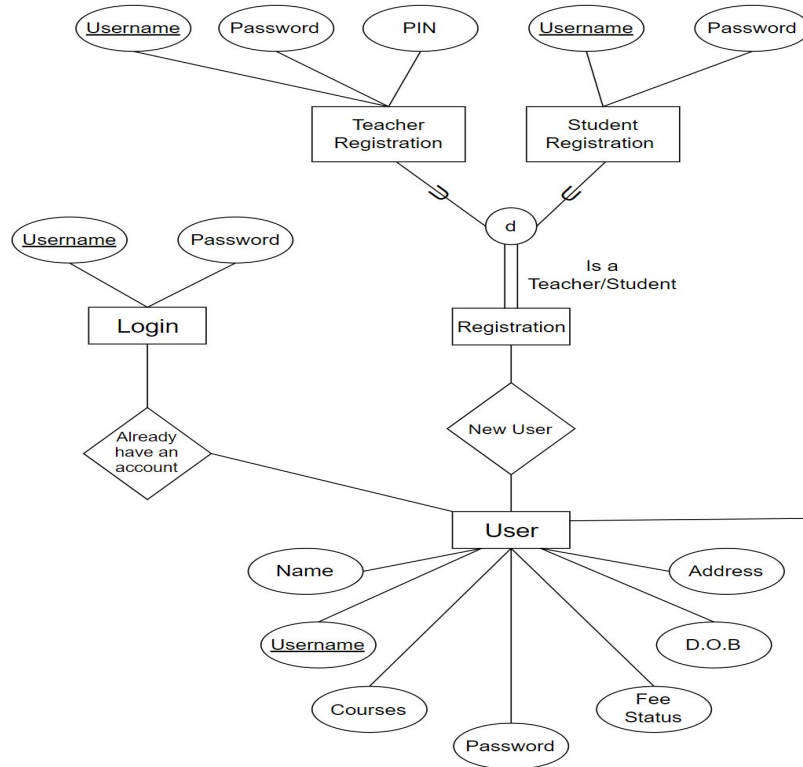
- Who is the User?
- Who is a student of the institute?
- Who is a teacher of the institute?
- How are the student and teacher different from one another?
- Does a student has some unique ID to distinguish him/her from other students? Same goes with the teachers as well?
- What are the various types of complaints a student can make?

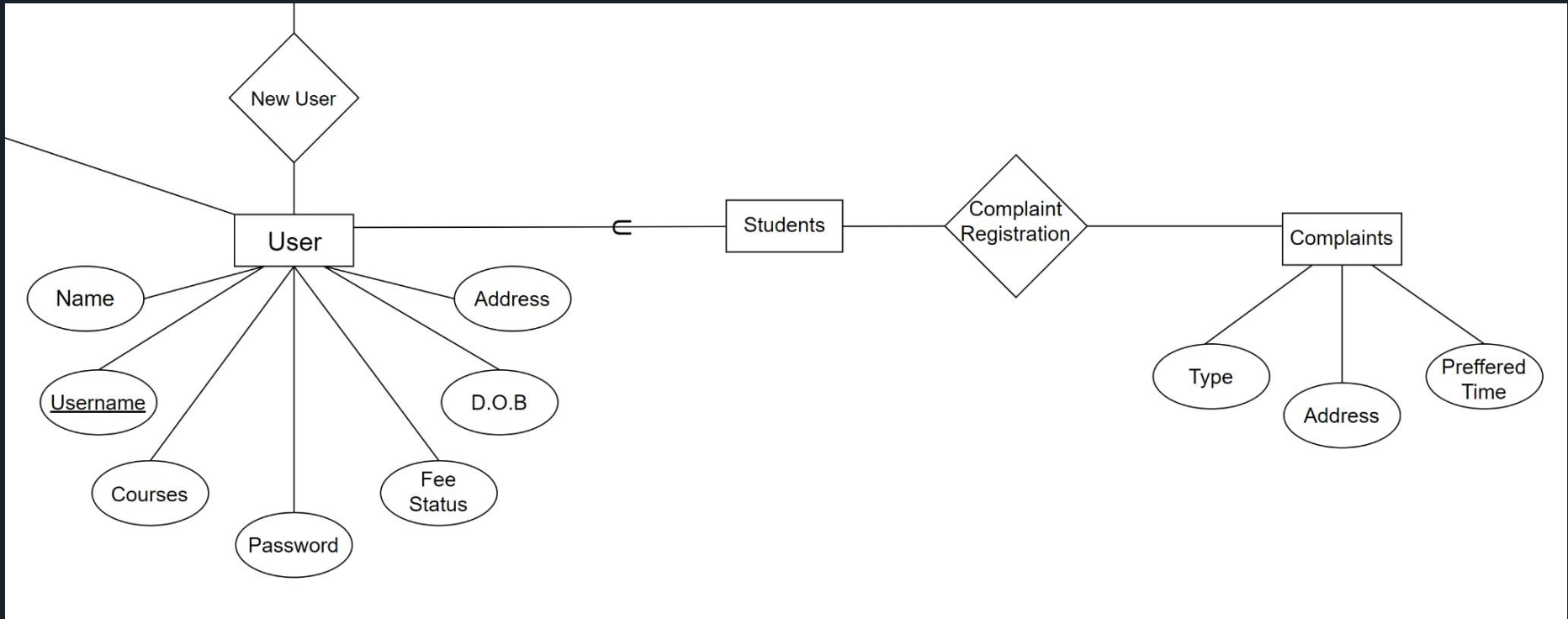
- 
- Which details regarding a particular complaint does a user provides?
 - What does attendance mean?
 - How frequently should the attendance be updated?
 - What does a notice mean?
 - Are there different types of notices? If so then how are they different from one another?

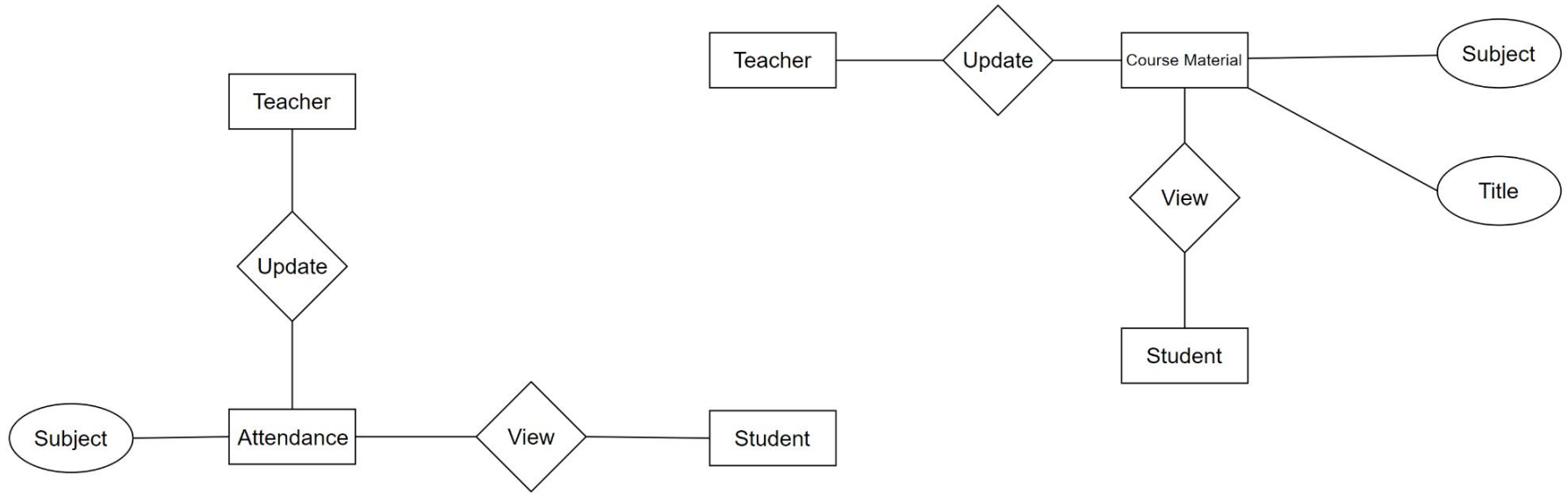
Designing

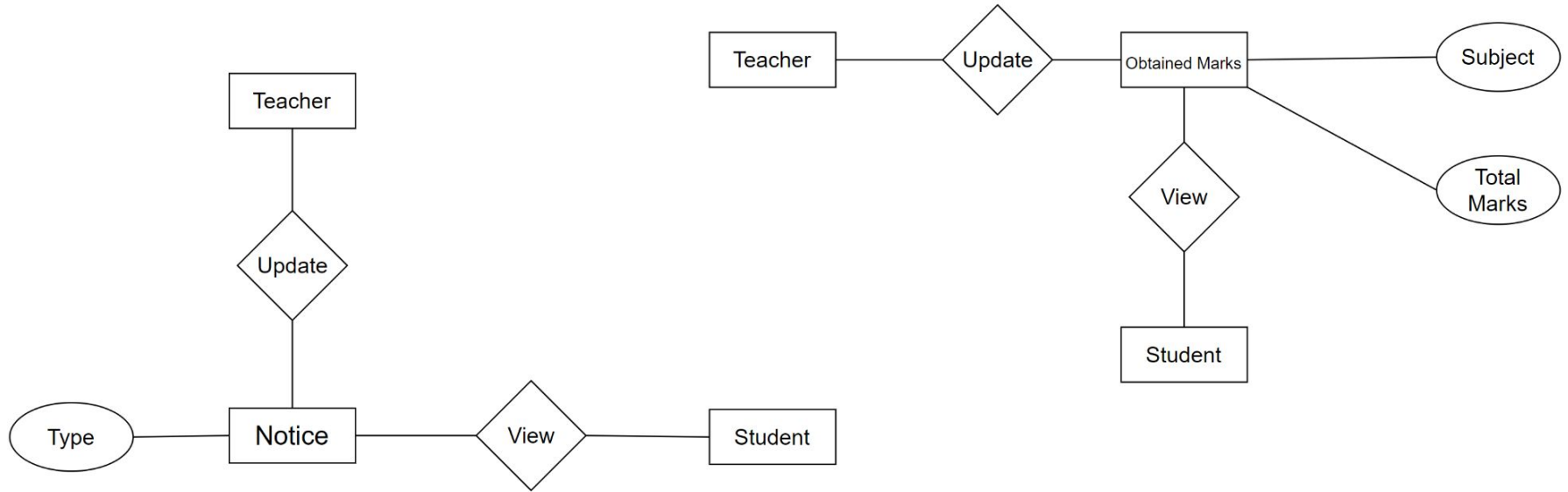


ER Diagram

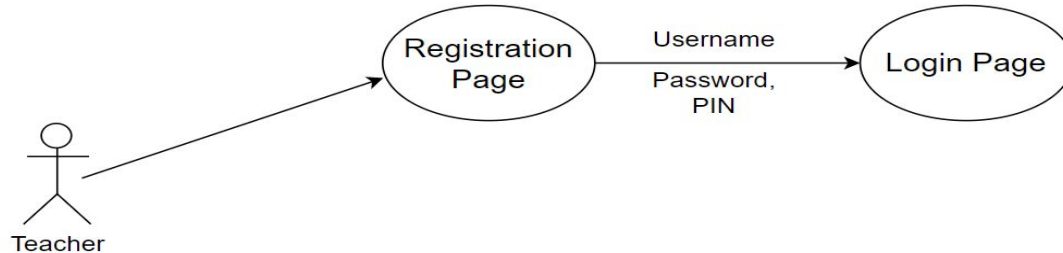
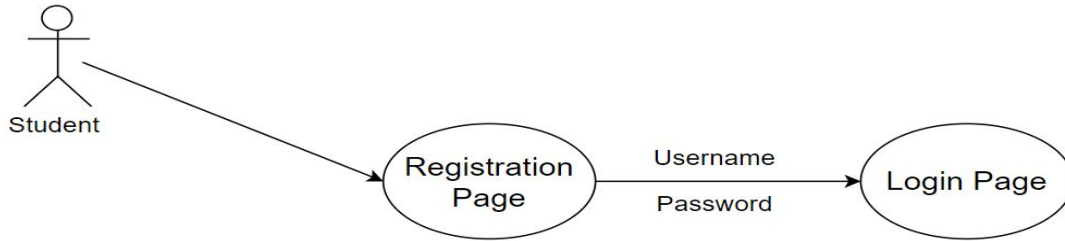




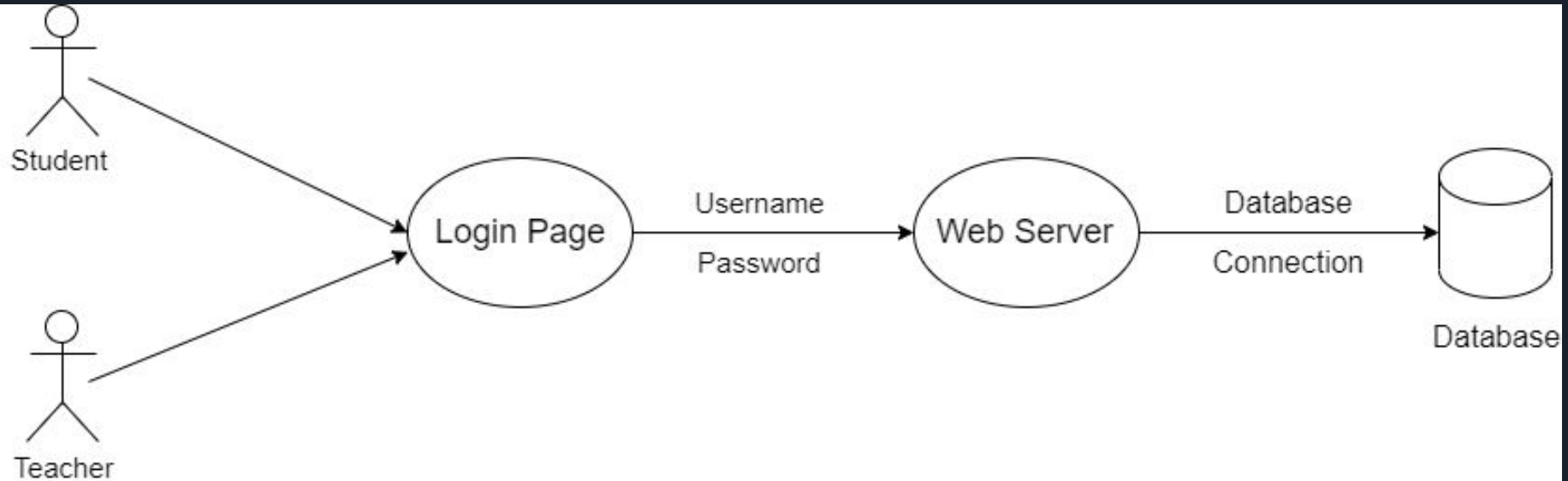




Registration Page Design



Login Page Design



Code Writing




```

index.html X signupprof.html signupstudent.html login.html # app.css
index.html > html > body
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="UTF-8" />
5     <meta http-equiv="X-UA-Compatible" content="IE=edge" />
6     <meta name="viewport" content="width=device-width, initial-scale=1.0" />
7     <link rel="stylesheet" href="app.css" />
8     <title>Home Page</title>
9   </head>
10  <body>
11    <h1>Register</h1>
12    <div class="container">
13      <button>Sign Up as Student</button>
14      <br />
15      <br />
16      <button>Sign Up as Teacher</button>
17      <br />
18      <br />
19      <button>Already have an account?</button>
20    </div>
21  </body>
22 </html>
23
24

```

```

index.html signupprof.html signupstudent.html login.html X # app.css
login.html > ...
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="UTF-8" />
5     <meta http-equiv="X-UA-Compatible" content="IE=edge" />
6     <meta name="viewport" content="width=device-width, initial-scale=1.0" />
7     <link rel="stylesheet" href="app.css" />
8     <title>Login</title>
9   </head>
10  <body>
11    <h1>Login Details</h1>
12    <form action="backend.php">
13      <div class="studentlog">
14        <label for="name"><b>Username</b></label>
15        <input type="text" name="username" />
16        <br />
17        <br />
18        <label for="passwd"><b>Password</b></label>
19        <input type="password" name="passwd" />
20        <br>
21        <br>
22        <button class="submitbutton">Submit</button>
23      </div>
24    </form>
25  </body>
26 </html>

```

```

index.html  signupprof.html X  signupstudent.html  login.html  # app.css
signupprof.html > html > body > form > div.proflong > br
1  <!DOCTYPE html>
2  <html lang="en">
3    <head>
4      <meta charset="UTF-8" />
5      <meta http-equiv="X-UA-Compatible" content="IE=edge" />
6      <meta name="viewport" content="width=device-width, initial-scale=1.0" />
7      <link rel="stylesheet" href="app.css" />
8      <title>Prof Sign Up</title>
9    </head>
10   <body>
11     <h1>Sign Up Details</h1>
12     <form action="backend.php">
13       <div class="proflog">
14         <label for="name"><b>Name</b></label>
15         <input type="text" name="name" />
16         <br />
17         <br />
18         <label for="name"><b>Username</b></label>
19         <input type="text" name="username" />
20         <br />
21         <br />
22         <label for="passwd"><b>Password</b></label>
23         <input type="password" name="passwd" />
24       </div>
25       <br />
26       <label for="PIN"><b>Pin</b></label>
27       <input type="password" name="PIN" />
28       <br>
29       <br>
30       <button class="submitbutton">Submit</button>
31     </div>
32   </form></body></html>

```

```

index.html  signupprof.html  signupstudent.html X  login.html  # app.css
signupstudent.html > html > head > title
1  <!DOCTYPE html>
2  <html lang="en">
3    <head>
4      <meta charset="UTF-8" />
5      <meta http-equiv="X-UA-Compatible" content="IE=edge" />
6      <meta name="viewport" content="width=device-width, initial-scale=1.0" />
7      <link rel="stylesheet" href="app.css" />
8      <title>Student Sign Up</title>
9    </head>
10   <body>
11     <h1>Sign Up Details</h1>
12     <form action="backend.php">
13       <div class="studentlog">
14         <label for="name"><b>Name</b></label>
15         <input type="text" name="name" />
16         <br />
17         <br />
18         <label for="name"><b>Username</b></label>
19         <input type="text" name="username" />
20         <br />
21         <br />
22         <label for="passwd"><b>Password</b></label>
23         <input type="password" name="passwd" />
24         <br>
25         <br>
26         <button class="submitbutton">Submit</button>
27       </div>
28     </form>
29   </body>
30 </html>
31

```

JS app.js X

```

JS app.js > app.get("/registerTeacher") callback
1  const express = require('express');
2  const path = require('path');
3  const mongoose = require('mongoose');
4  const ejsMate = require('ejs-mate');
5  const passport = require('passport');
6  const localStrategy = require('passport-local');
7  const User = require("../models/user");
8
9  const dbUrl = process.env.DB_URL || 'mongodb://localhost:27017/mis'
10 mongoose.connect(dbUrl, {
11   useNewUrlParser: true,
12   //useCreateIndex: true,
13   useUnifiedTopology: true,
14   //useFindAndModify: false,
15 });
16 const db= mongoose.connection;
17 db.on("error", console.error.bind(console, "connection error:"));
18 db.once("open", () => {
19   console.log("Database connected");
20 });
21
22 const app=express();
23 app.engine('ejs',ejsMate);
24 app.set('view engine','ejs');
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```

[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
listening on port 3000!!
Database connected
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
listening on port 3000!!
Database connected
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
listening on port 3000!!
Database connected
```

JS app.js X JS user.js

```

JS app.js > app.get("/registerTeacher") callback
25 app.use(express.urlencoded({extended : true}));
26 //app.use(methodOverride('_method'));
27
28 app.use(passport.initialize());
29 //app.use(passport.session());
30 passport.use(new LocalStrategy(User.authenticate()));
31 passport.serializeUser(User.serializeUser());
32 passport.deserializeUser(User.deserializeUser());
33
34
35
36 app.get('/', (req,res)=>{
37   res.send('hello');
38 })
39
40 app.get("/login",function(req,res){
41   res.render("login");
42 });
43
44 app.get("/registerStudent", (req,res) =>{
45   res.render("registerStudent");
46 });
47
48 app.get("/registerTeacher", (req,res) =>{
49   res.render("registerTeacher");
50 });
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```

[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
listening on port 3000!!
Database connected
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
listening on port 3000!!
Database connected
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
listening on port 3000!!
Database connected
```

```

JS app.js JS user.js X
models > JS user.js > [X] UserSchema > feeStatus > type
1  const mongoose = require('mongoose');
2  const Schema = mongoose.Schema;
3  const passport = require('passport');
4  const passportLocalMongoose = require('passport-local-mongoose');
5
6
7  const UserSchema = new Schema({
8    username:{
9      type: String
10    },
11
12    name:{
13      type: String
14    },
15
16    dob:{
17      type: String
18    },
19
20    address:{
21      type: String
22    },
23
24    // courses:[

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```

[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
listening on port 3000!!
Database connected
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
listening on port 3000!!
Database connected
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
listening on port 3000!!
Database connected

```

```

JS app.js JS user.js X
models > JS user.js > [X] UserSchema > isVarified
25 // {
26 //   type: String
27 // }
28 // ],
29
30 password:{
31   type: String
32 },
33
34 role:{
35   type : String
36 },
37
38
39 isVarified:{
40   type: String,
41   default: 'unpaid',
42 }
43
44 });
45
46
47 UserSchema.plugin(passportLocalMongoose);
48 module.exports =mongoose.model("User",UserSchema);

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```

[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
listening on port 3000!!
Database connected
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
listening on port 3000!!
Database connected
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
listening on port 3000!!
Database connected

```


Work to be done





Creating Models and Code Writing

- We have to make an attendance model which will have the data about attendance of all students. It would be updated only by the teacher.
- The course material, grade sheet will be uploaded by the teacher and it can be seen by the students.
- We have to provide the facility of uploading notices to the teacher.

- 
- Providing facility to the student to register complaints of various types.
 - The user model is to be updated in such a way that a user can update his/her username, password and personal details.
 - Estimated completion date : 05/11/2022.



Testing

- The testing phase will begin after that and is to be completed by 07/11/2022.
- We will consider certain scenarios and test our website based on the desired output and the obtained output.
- If we don't obtain the desired output, we might need to make some changes.