Name : - Kushwaha Rahul Shri Ravindra

Roll No :- 16 (I.C.T)

Subject :- OSWD Practical Assignment-2

GitHub : - https://github.com/rahulkushwaha20/assignment_node_2

1) Develop a user registration form and store its data in any database using Express. Form should also contain file upload (single, multiple) with validations.

User.js

```
const mongoose = require("mongoose");
const userSchema = new mongoose.Schema({
    name : {
        type : String,
        required: true
    },
    email :{
        type : String,
        required : true
    },
    phone : {
        type : String,
        required : true
    },
    image :{
        type: String,
        required : true
    },
    created:{
        type : Date,
        required : true,
        default : Date.now
    }
});

module.exports = mongoose.model('user',userSchema);
```

route.js

```
const express = require("express");
const router = express.Router();
const User = require('../models/users');
const multer = require("multer");
const fs = require("fs");

//image upload
var storage = multer.diskStorage({
    destination: function(req, res, cb){
        cb(null,'./uploads');
    },
    filename : function(req,file,cb){
        cb(null,file.fieldname+"_"+Date.now()+"_"+file.ori
ginalname);
    },
});

//this is middleware for upload
var upload = multer({
    storage : storage,
}).single("image");

//insert data for database route

router.post("/add",upload,(req,res) => {
    const user = new User({
        name:req.body.name,
        email:req.body.email,
        phone:req.body.phone,
        image:req.file.filename,
    });
    user.save((err) =>{
        if(err){
            res.json({message: err.message, type:
'danger'});
        }else{
            req.session.message = {
                type : 'success',
                message: "User Add successfully"
            };
            res.redirect("/");
        }
    })
});


//display all the users
router.get('/', (req,res) => {
    User.find().exec((err, users) =>{
        if(err){
            res.json({ message : err.message})
        }else{
            res.render('index',{
                title : "Home Page",
                users : users,
            });
        }
    });
});
```

```
router.get('/add',(req,res) =>{
    res.render("add_user", { title : "Add User"})
})

//edit route get
router.get('/edit/:id', (req, res) =>{
    let id = req.params.id;
    User.findById(id,(err,user) =>{
        if(err){
            res.redirect("/");
        }else{
            if(user == null){
                res.redirect("/");
            }else{
                res.render('edit_users',{
                    title : "Edit Page",
                    user : user
                })
            }
        }
    })
})

//update route post method

router.post("/update/:id",upload, (req,res) =>{
    let id = req.params.id;
    let new_image = "";
    if(req.file){
        new_image = req.file.filename;
        try{
            fs.unlinkSync('./uploads/'+req.body.old_image)
;
```

```
        }catch(err){
            console.log(err);
        }
    }else{
        new_image = req.body.old_image;

    }

    User.findByIdAndUpdate(id, {
        name : req.body.name,
        email : req.body.email,
        phone : req.body.phone,
        image : new_image,
    }, (err, result) =>{
        if(err){
            res.json({message: err.message, type:
'danger'});
        }else{
            req.session.message = {
                type : 'success',
                message: "User Update successfully"
            };
            res.redirect("/");
        }
    })
})

//Delete route

router.get("/delete/:id", (req,res) =>{
    let id = req.params.id;
    User.findByIdAndRemove(id,(err,result) => {
        if(result.image != ""){
            try{
                fs.unlinkSync("./uploads/" +
result.image);
            }catch(err){
                console.log(err);
            }
        }
            if(err){
                res.json({message: err.message});
            }else{
                req.session.message = {
                    type : 'info',
                    message: "User Delete successfully"
                };
                res.redirect("/");
            }

    })
})

module.exports = router;
```

main.js

```
//imports
require("dotenv").config();
const express = require("express");
const mongoose = require("mongoose");
const session = require("express-session");

const app = express();
const PORT = process.env.PORT || 4000;
```

```
//database connection
mongoose.connect(process.env.DB_URI, {
    useNewUrlParser: true,
    useUnifiedTopology: true
});
const db = mongoose.connection;

db.on("error",(error) => console.log(error));
db.once("open", () => console.log("Connecte to
database"))

//middleware
app.use(express.urlencoded({extended : false}));
app.use(express.json());

app.use(session({
    secret : "my secret key",
    saveUninitialized : true,
    resave : false
}));

app.use((req,res,next) =>{
    res.locals,message = req.session.message;
    delete req.session.message;
    next();
});

//image load
app.use(express.static('uploads'));


//set template engine
app.set("view engine","ejs");
```

```
//route perfix
app.use("",require("./routes/routes"))

app.listen(PORT,() =>{
    console.log('Server start at 
http://localhost:${PORT}')
})
```

Add_user.ejs

```
<%- include('layout/header') %>

<div class="container">
    <div class="row">
        <div class="col-lg-6 mx-auto mt-4">
            <div class="card shadow">
                <div class="card-header bg-primary">
                    <h5 class="text-light">Add New 
User</h5>
                </div>
                <div class="card-body p-4">
                    <form action="/add" method="post" 
id="add-form" enctype="multipart/form-data">
                        <div class="mb-3">
                            <label 
for="name">Name</label>
                            <input type="text" 
name="name" class="form-control form-control-lg" 
placeholder="Enter Name" required/>
                        </div>
                        <div class="mb-3">
                            <label 
for="email">Email</label>
```

```
                            <input type="email"
name="email" class="form-control form-control-lg"
placeholder="Enter Email" required/>
                        </div>
                        <div class="mb-3">
                            <label
for="phone">Phone</label>
                            <input type="tel"
name="phone" class="form-control form-control-lg"
placeholder="Enter Phone" required/>
                        </div>
                        <div class="mb-3">
                            <label for="image"
class="form-label">Select Image</label>
                            <input type="file"
name="image" class="form-control form-control-lg"
required/>
                        </div>
                        <div class="mb-3">
                            <input type="submit"
name="submit" value="Add User" class="btn btn-primary
btn-lg" />
                        </div>
                    </form>
                </div>
            </div>
        </div>
    </div>
</div>

<%- include('layout/footer') %>
```

Edit_user.ejs

```
<%- include('layout/header') %>

<div class="container">
    <div class="row">
        <div class="col-lg-6 mx-auto mt-4">
            <div class="card shadow">
                <div class="card-header bg-primary">
                    <h5 class="text-light">Edit User
(<%= user.name  %>)</h5>
                </div>
                <div class="card-body p-4">
                    <form action="/update/<%=
user._id  %>" method="post" id="add-form"
enctype="multipart/form-data">
                        <div class="mb-3">
                            <label
for="name">Name</label>
                            <input type="text"
name="name" class="form-control form-control-lg"
placeholder="Enter Name" value="<%= user.name  %>"
required/>
                        </div>
                        <div class="mb-3">
                            <label
for="email">Email</label>
                            <input type="email"
name="email" class="form-control form-control-lg"
placeholder="Enter Email" value="<%= user.email  %>"
required/>
                        </div>
                        <div class="mb-3">
                            <label
for="phone">Phone</label>
```

```
                            <input type="tel"
name="phone" class="form-control form-control-lg"
placeholder="Enter Phone" value="<%= user.phone %>"
required/>
                        </div>
                        <div class="mb-3">
                            <label for="image"
class="form-label">Select Image</label>
                            <input type="file"
name="image" class="form-control form-control-lg"/>
                            <img src="/<%= user.image
%>" width="100" class="img-thumbnail mt-1">
                        </div>
                        <input type="hidden"
name="old_image" value="<%= user.image %>">
                        <div class="mb-3">
                            <input type="submit"
name="submit" value="Update User" class="btn btn-
primary btn-lg" />
                        </div>
                    </form>
                </div>
            </div>
        </div>
    </div>
</div>

<%- include('layout/footer') %>
```

Index.ejs

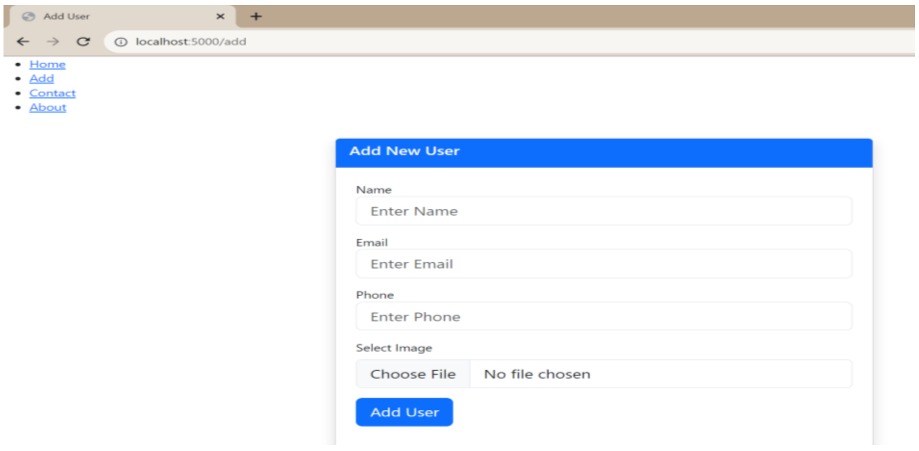
```
<%- include('layout/header') %>

<div class="container">
```

```
    <div class="row">
        <div class="col">
            <div class="col-lg-12">
                <% if (message) { %>
                    <div class="alert alert-dismissble
fade show alert-<%= message.type %>"
                        role="alert">
                        <button class="btn-close"
type="button" data-bs-dismiss="alert"
                        aria-label="close"></button>
                        <strong><%= message.message
%></strong>
                    </div>
                <% } %>
                <div class="table-responsive">
                <% if (users != '') { %>
                    <table class="table table-
striped">
                        <thead>
                         <tr>
                             <th>Id</th>
                             <th>Image</th>
                             <th>Name</th>
                             <th>Email</th>
                             <th>Phone</th>
                             <th>Action</th>
                         </tr>
                        </thead>
                        <tbody>
                         <% users.forEach((row, index)
=> { %>
                            <tr class="align-middle">
                                <td><%= index %></td>
                                <td><img src="<%=
row.image %>" width="50" class="img-thumbnail"></td>
                                <td><%= row.name
%></td>
                                <td><%= row.email
%></td>
                                <td><%= row.phone
%></td>
                                <td>
                                    <a href="/edit/<%=
row._id %>">Edit</a>
                                    <a
href="/delete/<%= row._id %>">Delete</a>
                                </td>
                            </tr>
                         <% }) %>
                        </tbody>
                     </table>
               <% } else{ %>
                    <h1 class="text-center text-
secondary mt-5">no user found in the database</h1>
               <% } %>

                </div>
            </div>
        </div>
    </div>
</div>
<%- include('layout/footer') %>
```

OUTPUT:



Add User
localhost:5000/add

- Home
- Add
- Contact
- About

Add New User

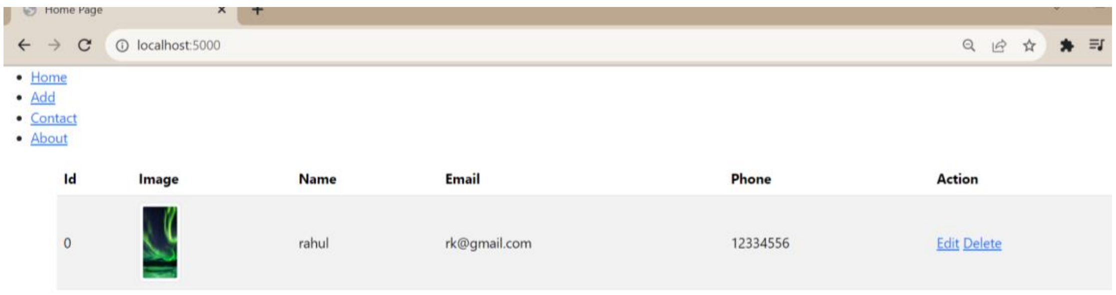
Name
Enter Name

Email
Enter Email

Phone
Enter Phone

Select Image
Choose File No file chosen

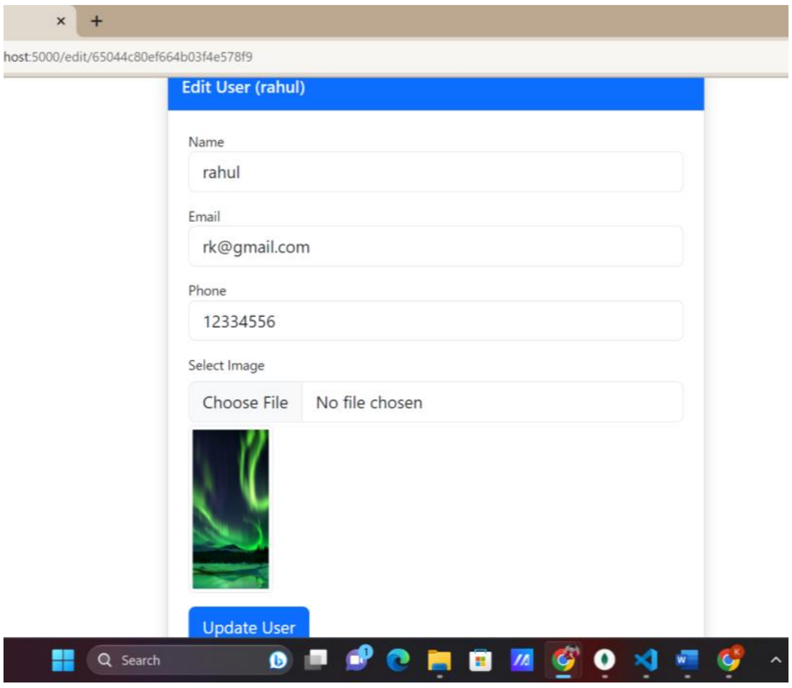
Add User



Home Page
localhost:5000

- Home
- Add
- Contact
- About

Id	Image	Name	Email	Phone	Action
0		rahul	rk@gmail.com	12334556	Edit Delete



host:5000/edit/65044c80ef664b03f4e578f9

Edit User (rahul)

Name
rahul

Email
rk@gmail.com

Phone
12334556

Select Image
Choose File No file chosen

Update User

2) **Express Login application with file session store.**

Index.js

```
const express = require('express')
const app = express()
const session = require('express-session')
const path = require('path')
const FileStore = require('session-file-
store')(session)

const PORT = 8080
app.use(express.static('public'))
app.use(express.urlencoded({ extended: false }))

app.use(session({
    secret: 'secret',
    resave: false,
    saveUninitialized: false,
    store: new FileStore({ path: './session-data' })
}))

app.get('/', (req, res, next) => {
    res.sendFile(__dirname + '/public/login.html')
})

app.post('/login', (req, res) => {
    var username = req.body.username;
    var password = req.body.password;

    if (username && password) {
        //credential validation logic here, if
credential matches than we store data in session
        req.session.loggedIn = true;
        req.session.username = username
        console.log(req.body)
        return res.redirect('/home')
    } else {
        return res.send('Please Enter Username And
Password!')
    }
})

app.get('/home', (req, res) => {
    if (req.session.loggedIn) {
        console.log('logged in')
        res.sendFile(__dirname + '/public/home.html')
    } else {
        console.log('not logged in')
        res.sendFile(__dirname + '/public/login.html')

    }
})

app.listen(PORT, () => {
    console.log(`server listening on
http://localhost:${PORT}`)
})
```

Login.html

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible"
content="IE=edge">
```

```html
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Log In Page</title>
    <style>
        .mb-2 {
            margin-bottom: 2rem;
        }

        .mt-2 {
            margin-top: 2rem;
        }

        .flex-center {
            display: flex;
            justify-content: center;
            flex-direction: column;
            align-items: center;
        }

        * {
            font-size: 1.2rem;
        }
    </style>
</head>

<body>
    <div class="container flex-center">
        <div class="header">
            <h3>Log In</h3>
        </div>
        <div class="form-container mt-2 ">
            <form class="flex-center" action="/login"
method="post">
                <div class="username mb-2">
```

```
                    <input type="text"
placeholder="enter username" name="username" required>
                </div>
                <div class="password mb-2">
                    <input type="password"
name="password" placeholder="enter password" id=""
required>
                </div>
                <div class="submit">
                    <input type="submit" value="Log
In">
                </div>
            </form>
        </div>
    </div>
</body>

</html>
```

Home.html

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible"
content="IE=edge">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Home Page</title>
</head>

<body>
    <div class="container">
```

```
        <h2>Home Page, You Are Authentic User</h2>
    </div>
</body>

</html>
```

OUTPUT:

```
{"cookie":{"originalMaxAge":null,"expires":null,"httpO
nly":true,"path":"/"},"loggedIn":true,"username":"neti
","__lastAccess":1691475158438}
```

3) Express Login application with redis session store.
Route.js

```
const router = require("express").Router()


 router.get("/", (req, res) => {
    res.render("Login.hbs")
  });

  const isLoggedIn = (req, res, next) => {
    if (req.session.userId) {
      next();
    } else {
      res.redirect("/login");
    }
  };
  router.post("/login", (req, res) => {
    try {
        const password = req.body.password;

        req.session.userId = Date.now();

        return res.render("dashboard.hbs");
    } catch (error) {
        console.log(error);
    }




  });
  router.get("/logout", (req, res) => {
    // Destroy session and redirect to login page
    req.session.destroy();
    res.redirect("/");
  });

module.exports = router
```

Server.js

```
const express = require("express");
const session = require("express-session");
const { createClient } = require("redis");
const hbs = require("hbs")
const path = require("path")

const app = express();

// Initialize client.

const redisClient = createClient();
redisClient.connect().then(()=>{
   console.log("redis client is connected");
}).catch((error)=>{
    console.log("not connected ",error)
});

// Initialize store.
const RedisStore = require("connect-redis").default
const redisStore = new RedisStore({
  client: redisClient,
  prefix: "Expressredis:",
});
```

```
// Initialize session storage.
app.use(
  session({
    store: redisStore,
    resave: false,
    saveUninitialized: false,
    secret: "keyboard cat",
  })
);

app.use(express.json())
app.use(express.urlencoded({extended:true}))
app.set("views",path.join(__dirname,"/src/views"))
app.set("view engine",hbs)



const routes = require("./src/Routes/routes")
app.use("/",routes)




// Start the server
app.listen(3000, () => {
  console.log("Server started on port 3000");
});
```

4) **Login with JWT, CRUD operations for students table with mongoose, express and any one template engine, Logout.**
Register.js

```
const mongoose = require("mongoose");
const jwt = require("jsonwebtoken");
const newRegisterSchema = new mongoose.Schema({
    name :{
        type : String,
        required : true
    },
    username :{
        type : String,
        required : true
    },
    password :{
        type : String,
        required : true
    },
    address :{
        type : String,
        required : true
    }
});

module.exports =
mongoose.model("register",newRegisterSchema);
```

user.js

```
const mongoose = require("mongoose");
const newUserSchema = new mongoose.Schema({
    name :{
        type : String,
        required : true
    },
    roll_no :{
        type : String,
        required : true
    }
});

module.exports = mongoose.model("user",newUserSchema);
```

loginres.js

```
const express = require("express");
const router = express.Router();
const users = require("../models/register");
const bcrypt = require("bcrypt");
const jwt = require("jsonwebtoken");


router.get('/',(req,res) =>{
    res.render("login");
 })

// Rendering of buttons Done Now handling post
requests
router.post("/login",async(req,res)=>{
    const {username,password}=req.body;
    let user= await users.findOne({username});
    if(!user)
    return res.redirect("/register");
    // Make sure this compare function is await else
it will go on the remaining procedure without actually
comparing whether we have the right user or not and it
will login regardless.
    const isUser=await
bcrypt.compare(password,user.password);
```

```
    if(!isUser){
        // We need to do return res.render to make
sure further statements do not get executed and we
redirect from here itself.
       return res.render("login",{message:"Invalid
Password"});
    }
    else{
        const
token=jwt.sign({_id:user._id},"jsiufbdiufbuibfIU");
        res.cookie("token",token,{
            httpOnly: true,
            expires: new Date(Date.now()+60*1000),
        })
        res.redirect("index");
    }
})



router.get("/register",(req,res) =>{
    res.render("register");
 });


 router.post("/register",async(req,res)=>{
    const {name,username,password,address}=req.body;
    let user= await users.findOne({username});//
Inside findOne u need to send an object
    if(user){
        res.redirect("/login");
    }
    console.log(name,username,password,address);
    const hashedPassword= await
bcrypt.hash(password,10); // 10 is a salt that we need
to mention
    user=await users.create({
        name,
        username,
        address,
        password: hashedPassword,

    }); // It is an async function

    // This is done to hide the exact id of the
document that will be created inside the users
collection everytime a new user registers. We pass an
object having key of _id with value of that particular
document's id and we encrypt it using jwt and an
algorithn which is a string in this case.
    const
token=jwt.sign({_id:user._id},"jsiufbdiufbuibfIU");
    res.cookie("token",token,{
        httpOnly: true,
        expires: new Date(Date.now()+60*1000),
    })
    res.redirect("/");
})



// Authentication function
const authentication=async(req,res)=>{
    const {token}=req.cookies;
    if(token){
        // Extracting the original ID form the cookie,
we decode using the token
```

```
        const
iD=jwt.verify(token,"jsiufbdiufbuibfIU");
        const user= await users.findById(iD);
        if(user)
           return
res.render("logout",{name:user.name});
    }
    return res.redirect("/login");
}

router.post("/logout",(req,res)=>{
    res.cookie("token",null,{
        httpOnly: true,
        expires: new Date(Date.now()),
    });
    res.redirect("/login");
})


router.use("",require("./router"));
module.exports = router;
```

route.js

```
const express = require("express");
const router = express.Router();
const User = require("../models/user");

router.get('/index',(req,res) =>{
    User.find().exec((err,user) => {
        if(err){
            console.log("error");
        }else{
            res.render('index',{
                user : user
            })
        }
    })
})

router.get("/add",(req,res) =>{
    res.render("add_user");
});

router.post("/add",(req,res) => {
    const user = new User({
        name: req.body.name,
        roll_no : req.body.roll_no
    });
    user.save((err) =>{
        if(err){
            console.log("Error");
        }else{
            console.log("Sucsess");
            res.redirect("/");
        }
    })
});



router.get("/edit/:id",(req,res) => {
    let id = req.params.id;
    User.findById(id,(err,user) =>{
        if(err){
            console.log("Erroer");
        }else{
            if(user == null){
                res.redirect("/");
            }else{
                res.render("edit_user",{
                    user : user
                })
            }
        }
    })
});

router.post("/update/:id",(req,res) => {
    let id = req.params.id;
    User.findByIdAndUpdate(id,{
        name : req.body.name,
        roll_no : req.body.roll_no
    }, (err,result) =>{
        if(err){
            console.log("error");
        }else{
            console.log("success");
            res.redirect("/");
        }

    })
});

router.get("/delete/:id",(req,res) =>{
    let id = req.params.id;
    User.findByIdAndRemove(id,(err,result) =>{
        if(err){
            console.log("Error");
        }else{
            console.log("Success");
            res.redirect("/");
        }
    })
 })


module.exports = router;
```

main.js

```
require("dotenv").config();
const express = require("express");
const session = require("express-session");
const mongoose = require("mongoose");
const app = express();
const FileStore = require('session-file-
store')(session)


const PORT = process.env.PORT || 8000;

//Database connection code
mongoose.connect(process.env.DB_URL,{
    useNewUrlParser : true,
    useUnifiedTopology : true

});

const db = mongoose.connection;

db.on("error", (error) =>{console.log(error)});
db.once("open",() =>{console.log("Connecte to
databse")});

//middleware
```

```
app.use(express.urlencoded({extended:true}));
app.use(express.json());

app.use(session({
    secret : "My secret",
    saveUninitialized : true,
    resave : false,
    store: new FileStore({ path: './session-data' })
}))

app.use((req,res,next) =>{
    res.locals.message = req.session.message;
    delete req.session.message;
    next();
})

app.post("/logout",(req,res)=>{
    res.render("login");
})

app.post("/logout",(req,res)=>{
    res.cookie("token",null,{
        httpOnly: true,
        expires: new Date(Date.now()),
    });
    res.redirect("/login");
})

//tamplate engine
app.set("view engine","ejs");

//prefix routes
app.use("",require("./routes/loginres"));
app.listen(PORT,() =>{
    console.log('Server start at
http://localhost:${PORT}')
});
```

Index.ejs

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h1>Hello, <%=locals.name%></h1>
    <li><a href="/">Home</a></li>
    <li><a href="/add">Add</a></li>
    <table>
        <thead>
            <tr>
                <th>Id</th>
                <th>Name</th>
                <th>Roll No</th>
                <th>Action</th>
            </tr>
        </thead>
        <tbody>
            <% user.forEach((row, index) => { %>
                <tr>
                    <td><%= index %></td>
                    <td><%= row.name %></td>
                    <td><%= row.roll_no %></td>
                    <td>
```

```
                        <a href="/edit/<%= row._id
%>">Edit</a>
                        <a href="/delete/<%= row._id
%>">Delete</a>
                    </td>
                </tr>
            <% }) %>
        </tbody>
    </table>
    <form action="/logout" method="post">
        <button type="submit" name="submit"
class="btn">Log Out</button>
    </form>
   </body>
</html>
```

Register.ejs

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <form action="/register" method="post" id="add-
form">
        <div class="mb-3">
            <label for="name">Name</label>
            <input type="text" name="name"
class="form-control form-control-lg"
placeholder="Enter Name" />
        </div>
        <div class="mb-3">
```

```
            <label for="username">UserName</label>
            <input type="text" name="username"
class="form-control form-control-lg"
placeholder="Enter Email" required/>
        </div>
        <div class="mb-3">
            <label for="username">Password</label>
            <input type="text" name="password"
class="form-control form-control-lg"
placeholder="Enter Email" required/>
        </div>
        <div class="mb-3">
            <label for="username">Address</label>
            <input type="text" name="address"
class="form-control form-control-lg"
placeholder="Enter Email" required/>
        </div>
        <div class="mb-3">
            <input type="submit" name="submit"
value="Add User" class="btn btn-primary btn-lg" />
        </div>
    </form>
</body>
</html>
```

Login.ejs

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Login</title>
</head>
```

```
<body>
    <form action="/login" method="post" id="add-form">
        <div class="mb-3">
            <label for="name">Username</label>
            <input type="text" name="username"
class="form-control form-control-lg"
placeholder="Enter Name" />
        </div>
        <div class="mb-3">
            <label for="email">Password</label>
            <input type="pwd" name="password"
class="form-control form-control-lg"
placeholder="Enter Email" required/>
        </div>
        <div class="mb-3">
            <input type="submit" name="submit"
value="Add User" class="btn btn-primary btn-lg" />
        </div>
        <a href="/register">Register</a>
    </form>
</body>
</html>
```

Add_user.ejs

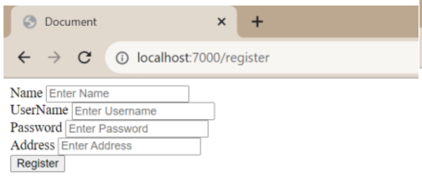
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <form action="/add" method="post" id="add-form">
```

```
        <div class="mb-3">
            <label for="name">Name</label>
            <input type="text" name="name"
class="form-control form-control-lg"
placeholder="Enter Name" />
        </div>
        <div class="mb-3">
            <label for="email">Email</label>
            <input type="text" name="roll_no"
class="form-control form-control-lg"
placeholder="Enter Email" required/>
        </div>
        <div class="mb-3">
            <input type="submit" name="submit"
value="Add User" class="btn btn-primary btn-lg" />
        </div>
    </form>
</body>
</html>
```

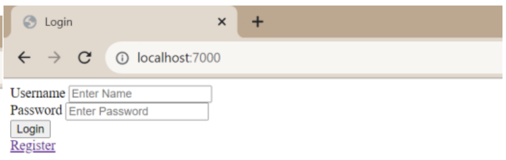
Edit_user.ejs

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <form action="/update/<%= user._id %>"
method="post" id="add-form">
        <div class="mb-3">
            <label for="name">Name</label>
            <input type="text" name="name"
class="form-control form-control-lg"
placeholder="Enter Name" value="<%=user.name %>" />
        </div>
        <div class="mb-3">
            <label for="email">Roll No:</label>
            <input type="text" name="roll_no"
class="form-control form-control-lg"
placeholder="Enter Email" value="<%= user.roll_no
%>"/>
        </div>
        <div class="mb-3">
            <input type="submit" name="submit"
value="Update User" class="btn btn-primary btn-lg" />
        </div>
    </form>
</body>
</html>
```

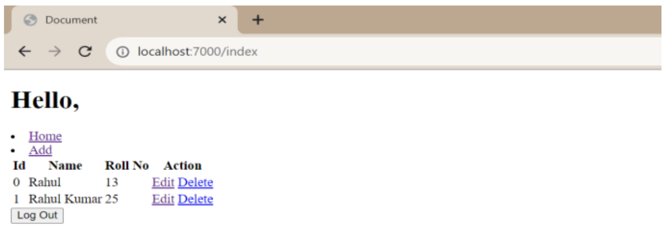
OUTPUT:



Name Enter Name
UserName Enter Username
Password Enter Password
Address Enter Address
Register



Username Enter Name
Password Enter Password
Login
Register

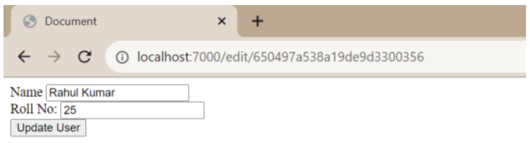


Hello,

- Home
- Add

Id	Name	Roll No	Action
0	Rahul	13	Edit Delete
1	Rahul Kumar	25	Edit Delete

Log Out



Name Rahul Kumar
Roll No: 25
Update User

5) Login with JWT, CRUD operations for students table with mongoose, express and
frontend(html,css,javascript/jquery/angularjs), Logout.
Register.js

```
const mongoose = require("mongoose");
const jwt = require("jsonwebtoken");
const newRegisterSchema = new mongoose.Schema({
    name :{
        type : String,
        required : true
    },
    username :{
        type : String,
        required : true
    },
    password :{
        type : String,
        required : true
    },
    address :{
        type : String,
        required : true
    }
});

module.exports =
mongoose.model("register",newRegisterSchema);
```

Student.js

```
const mongoose = require("mongoose");
const newUserSchema = new mongoose.Schema({
    name :{
        type : String,
        required : true
    },
    dept :{
        type : String,
        required : true
    },
    roll_no :{
        type : String,
        required : true
    },
    address :{
        type : String,
        required : true
    }
});

module.exports = mongoose.model("user",newUserSchema);
```

loginres.js

```
const express = require("express");
const router = express.Router();
const users = require("../models/register");
const bcrypt = require("bcrypt");
const jwt = require("jsonwebtoken");


router.get('/',(req,res) =>{
    res.render("login");
 })

// Rendering of buttons Done Now handling post
requests
router.post("/login",async(req,res)=>{
    const {username,password}=req.body;
    let user= await users.findOne({username});
    if(!user)
    return res.redirect("/register");
    // Make sure this compare function is await else
it will go on the remaining procedure without actually
comparing whether we have the right user or not and it
will login regardless.
    const isUser=await
bcrypt.compare(password,user.password);
    if(!isUser){
        // We need to do return res.render to make
sure further statements do not get executed and we
redirect from here itself.
        return res.render("login",{message:"Invalid
Password"});
    }
    else{
        const
token=jwt.sign({_id:user._id},"jsiufbdiufbuibfIU");
        res.cookie("token",token,{
            httpOnly: true,
            expires: new Date(Date.now()+60*1000),
        })
        res.redirect("index");
    }
})



router.get("/register",(req,res) =>{
    res.render("register");
 });


 router.post("/register",async(req,res)=>{
    const {name,username,password,address}=req.body;
    let user= await users.findOne({username});//
Inside findOne u need to send an object
    if(user){
        res.redirect("/login");
    }
    console.log(name,username,password,address);
    const hashedPassword= await
bcrypt.hash(password,10); // 10 is a salt that we need
to mention
    user=await users.create({
        name,
        username,
        address,
        password: hashedPassword,

    }); // It is an async function

    // This is done to hide the exact id of the
document that will be created inside the users
collection everytime a new user registers. We pass an
object having key of _id with value of that particular
document's id and we encrypt it using jwt and an
algorithn which is a string in this case.
    const
token=jwt.sign({_id:user._id},"jsiufbdiufbuibfIU");
    res.cookie("token",token,{
        httpOnly: true,
        expires: new Date(Date.now()+60*1000),
    })
    res.redirect("/");
})
```

```
// Authentication function
const authentication=async(req,res)=>{
    const {token}=req.cookies;
    if(token){
        // Extracting the original ID form the cookie,
we decode using the token
        const
iD=jwt.verify(token,"jsiufbdiufbuibfIU");
        const user= await users.findById(iD);
        if(user)
           return
res.render("logout",{name:user.name});
    }
    return res.redirect("/login");
}

router.post("/logout",(req,res)=>{
    res.cookie("token",null,{
        httpOnly: true,
        expires: new Date(Date.now()),
    });
    res.redirect("/login");
})


router.use("",require("./router"));
module.exports = router;
```

router.js

```
const express = require("express");
const router = express.Router();
const User = require("../models/student");
```

```
router.get('/index',(req,res) =>{
    User.find().exec((err,user) => {
        if(err){
            console.log("error");
        }else{
            res.render('index',{
               user : user
            })
        }
    })
 })

 router.get("/add",(req,res) =>{
    res.render("add_user");
 });

 router.post("/add",(req,res) => {
    const user = new User({
        name: req.body.name,
        dept : req.body.dept,
        roll_no : req.body.roll_no,
        address : req.body.address
    });
    user.save((err) =>{
        if(err){
            console.log("Error");
        }else{
            console.log("Sucsess");
            res.redirect("index");
        }
    })
 });
```

```
router.get("/edit/:id",(req,res) => {
   let id = req.params.id;
   User.findById(id,(err,user) =>{
       if(err){
           console.log("Erroer");
       }else{
           if(user == null){
               res.redirect("/");
           }else{
               res.render("edit_user",{
                   user : user
               })
           }
       }
   })
});

router.post("/update/:id",(req,res) => {
   let id = req.params.id;
   User.findByIdAndUpdate(id,{
       name : req.body.name,
       dept : req.body.dept,
       roll_no : req.body.roll_no,
       address : req.body.address
   }, (err,result) =>{
       if(err){
           console.log("error");
       }else{
           console.log("success");
           res.redirect("index");
       }

   })
});
```

```
 router.get("/delete/:id",(req,res) =>{
     let id = req.params.id;
     User.findByIdAndRemove(id,(err,result) =>{
         if(err){
             console.log("Error");
         }else{
             console.log("Success");
             res.redirect("index");
         }
     })
 })


module.exports = router;
```

.env

```
PORT = 9000
DB_URl = mongodb://127.0.0.1:27017/studentDB
```

Main.js

```
require("dotenv").config();
const express = require("express");
const session = require("express-session");
const mongoose = require("mongoose");
const app = express();
const FileStore = require('session-file-
store')(session)


const PORT = process.env.PORT || 8000;

//Database connection code
mongoose.connect(process.env.DB_URL,{
    useNewUrlParser : true,
    useUnifiedTopology : true

});

const db = mongoose.connection;

db.on("error", (error) =>{console.log(error)});
db.once("open",() =>{console.log("Connecte to
databse")});

//middleware

app.use(express.urlencoded({extended:true}));
app.use(express.json());

app.use(session({
    secret : "My secret",
    saveUninitialized : true,
    resave : false,
    store: new FileStore({ path: './session-data' })
}))

app.use((req,res,next) =>{
    res.locals.message = req.session.message;
    delete req.session.message;
    next();
})

app.post("/logout",(req,res)=>{
    res.render("login");
})

app.post("/logout",(req,res)=>{
    res.cookie("token",null,{
        httpOnly: true,
        expires: new Date(Date.now()),
    });
    res.redirect("/login");
})

//tamplate engine
app.set("view engine","ejs");

//prefix routes
app.use("",require("./routes/loginres"));

app.listen(PORT,() =>{
    console.log('Server start at
http://localhost:${PORT}')
});
```

Index.ejs

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h1>Hello, <%=locals.name%></h1>
    <li><a href="/">Home</a></li>
    <li><a href="/add">Add</a></li>
    <table>
        <thead>
            <tr>
                <th>Id</th>
                <th>Name</th>
                <th>Roll No</th>
                <th>Department</th>
                <th>Address</th>
                <th>Action</th>
            </tr>
        </thead>
        <tbody>
            <% user.forEach((row, index) => { %>
                <tr>
                    <td><%= index %></td>
                    <td><%= row.name %></td>
                    <td><%= row.roll_no %></td>
                    <td><%= row.dept %></td>
                    <td><%= row.address %></td>

                    <td>
                        <a href="/edit/<%= row._id
%>">Edit</a>
                        <a href="/delete/<%= row._id
%>">Delete</a>
                    </td>
                </tr>
            <% }) %>
        </tbody>
    </table>
    <form action="/logout" method="post">
        <button type="submit" name="submit"
class="btn">Log Out</button>
    </form>
   </body>
</html>
```

Login.ejs

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Login</title>
</head>
<body>
    <h>Login Page</h>
    <form action="/login" method="post" id="add-form">
        <div class="mb-3">
            <label for="name">Username</label>
            <input type="text" name="username"
class="form-control form-control-lg"
placeholder="Enter user" />
        </div>
        <div class="mb-3">
            <label for="email">Password</label>
            <input type="pwd" name="password"
class="form-control form-control-lg"
placeholder="Enter password" required/>
        </div>
        <div class="mb-3">
            <input type="submit" name="submit"
value="Add User" class="btn btn-primary btn-lg" />
        </div>
        <a href="/register">Register</a>
    </form>
</body>
</html>
```

Register.ejs

```html
<!DOCTYPE html>
```

```html
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h1>Register page</h1>
    <form action="/register" method="post" id="add-
form">
        <div class="mb-3">
            <label for="name">Name</label>
            <input type="text" name="name"
class="form-control form-control-lg"
placeholder="Enter Name" />
        </div>
        <div class="mb-3">
            <label for="username">UserName</label>
            <input type="text" name="username"
class="form-control form-control-lg"
placeholder="Enter Username" required/>
        </div>
        <div class="mb-3">
            <label for="username">Password</label>
            <input type="password" name="password"
class="form-control form-control-lg"
placeholder="Enter Password" required/>
        </div>
        <div class="mb-3">
            <label for="username">Address</label>
            <input type="text" name="address"
class="form-control form-control-lg"
placeholder="Enter Address" required/>
```

```
        </div>
        <div class="mb-3">
            <input type="submit" name="submit"
value="Add User" class="btn btn-primary btn-lg" />
        </div>
    </form>
</body>
</html>
```

Add_user.ejs

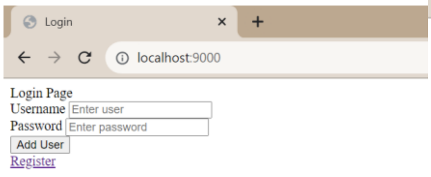
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h1>Add Page</h1>
    <form action="/add" method="post" id="add-form">
        <div class="mb-3">
            <label for="name">Name:</label>
            <input type="text" name="name"
class="form-control form-control-lg"
placeholder="Enter Name" />
        </div>
        <div class="mb-3">
            <label for="email">Roll No:</label>
            <input type="text" name="roll_no"
class="form-control form-control-lg"
placeholder="Enter roll no" required/>
        </div>
        <div class="mb-3">
```

```
            <label for="email">Department:</label>
            <input type="text" name="dept"
class="form-control form-control-lg"
placeholder="Enter department" required/>
        </div>
        <div class="mb-3">
            <label for="email">Address:</label>
            <input type="text" name="address"
class="form-control form-control-lg"
placeholder="Enter address" required/>
        </div>
        <div class="mb-3">
            <input type="submit" name="submit"
value="Add User" class="btn btn-primary btn-lg" />
        </div>
    </form>
</body>
</html>
```

Edit_user.ejs

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h1>Update Page</h1>
    <form action="/update/<%= user._id %>"
method="post" id="add-form">
        <div class="mb-3">
            <label for="name">Name</label>
            <input type="text" name="name"
class="form-control form-control-lg"
placeholder="Enter Name" value="<%=user.name %>" />
        </div>
        <div class="mb-3">
            <label for="email">Roll No:</label>
            <input type="text" name="roll_no"
class="form-control form-control-lg"
placeholder="Enter roll no" value="<%= user.roll_no
%>"/>
        </div>
        <div class="mb-3">
            <label for="email">Department</label>
            <input type="text" name="dept"
class="form-control form-control-lg"
placeholder="Enter department" value="<%= user.dept
%>"/>
        </div>
        <div class="mb-3">
            <label for="email">Address</label>
            <input type="text" name="address"
class="form-control form-control-lg"
placeholder="Enter Address" value="<%= user.address
%>"/>
        </div>
        <div class="mb-3">
            <input type="submit" name="submit"
value="Update User" class="btn btn-primary btn-lg" />
        </div>
    </form>
</body>
</html>
```

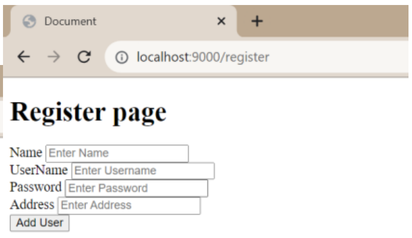
OUTPUT:



Login

localhost:9000

Login Page
Username Enter user
Password Enter password
Add User
Register

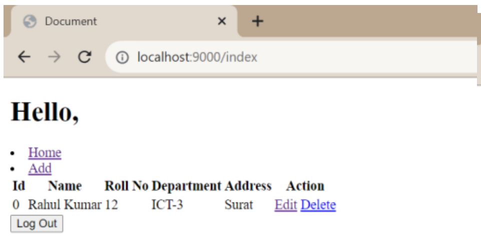


Document

localhost:9000/register

Register page

Name Enter Name
UserName Enter Username
Password Enter Password
Address Enter Address
Add User



Document

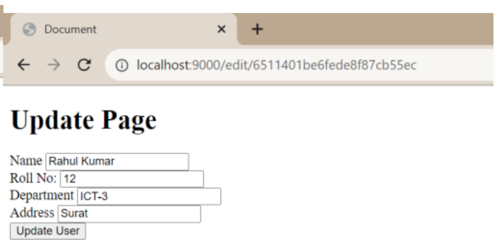
localhost:9000/index

Hello,

- Home
- Add

Id	Name	Roll No	Department	Address	Action
0	Rahul Kumar	12	ICT-3	Surat	Edit Delete

Log Out

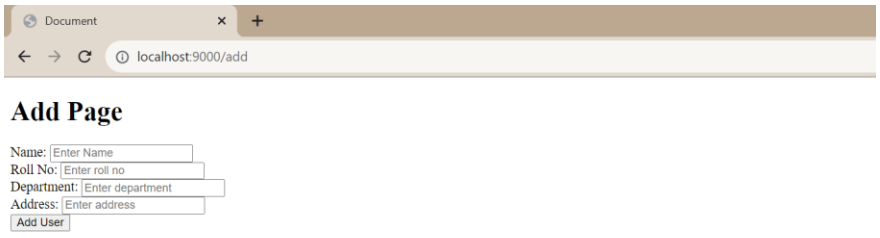


Document

localhost:9000/edit/6511401be6fede8f87cb55ec

Update Page

Name Rahul Kumar
Roll No: 12
Department ICT-3
Address Surat
Update User



Document

localhost:9000/add

Add Page

Name: Enter Name
Roll No: Enter roll no
Department: Enter department
Address: Enter address
Add User