

Name: Muskan Vinod Aneja

Roll no: 2

Semester: 7th

Subject: Application development using Full Stack

Assignment: Practical assignment 2

1) Express File upload (single, multiple) with validations.

app.js

```
const express = require("express");
const app = express();
const path = require("path");

const {PORT = 3000} = process.env;
const multer = require("multer");

const storage = multer.diskStorage({
  destination:(req,file,callback)=>{
    callback(null,path.join(__dirname,"uploads"))
  },
  filename:(req,file,callback)=>{
    callback(null,file.originalname)
  }
});

const maxFileSize = 10 * 1024 * 1024;

const upload = multer({

  storage:storage,
  fileFilter:(req,file,callback)=>{

    if(file.mimetype=="image/jpg" || file.mimetype == "image/jpeg" ||
file.mimetype == "image/png")
    {
      callback(null,true);
    }
    else
    {

      callback(null,false);

      return callback(new Error("Only .png,.jpeg,.jpg images are
allowed.."));
    }
  }
});
```

```
    }  
  },  
  limits: maxFileSize  
  
}).array("images", 12);  
  
const single = multer({  
  storage: storage,  
  fileFilter: (req, file, callback) => {  
    if (file.mimetype === "image/jpg" || file.mimetype === "image/jpeg" ||  
file.mimetype === "image/png")
```

```

        {
            callback(null,true);
        }
        else
        {
            callback(null,false);

            return callback(new Error("Only .png,.jpeg,.jpg images are
allowed.."));
        }
    },
    limits:maxFileSize
}).single("images");

app.set("view engine","ejs");
app.use(express.static(path.join(__dirname,"assets")));

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

app.get("/application",(req,res)=>{

    res.render("index",{
        title:" Multiple File Upload"
    });

});

app.post("/sendFile",(req,res)=>{

    upload(req,res,(err)=>{
        if(err instanceof multer.MulterError)

        {

            //      Error i.e. occured while uploading(Multer error);
            res.status(400).json({err,message:"File upload
failed....",status:false});
        }
        else if(err)

        {
            //Incase error come...

```

```
        res.status(400).json({message:"Sorry bro, an unknow error is  
occurred..",status:false});  
    }
```

```
        res.status(200).json({message:"Files have been uploaded  
successfully",status:true});  
    });  
});
```

```
app.get("/single",(req,res)=>{  
    res.render("single",{  
  
        title:"File Upload App"  
    });  
});
```

```

app.post("/sendSingleFile",single,(req,res)=>{

    upload(req,res,(err)=>{
        if(err instanceof multer.MulterError)
        {

            //      Error i.e. occured while uploading(Multer error);
            res.status(400).json({err,message:"File upload
failed....",status:false});

        }
        else if(err)
        {

            console.log(err);
            //Incase error come...

            res.status(400).json({message:"Sorry bro, an unknow error is
occured..",status:false});
        }

        res.status(200).json({message:"File has been
uploaded...",status:true});
    });
});

```

```

app.listen(PORT,()=>{
    console.log(`Server is listening to PORT: ${PORT}`);
});

```

index.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>
<%=title%> </title>

```

```
</head>
```

```
<body>
```

```
  <form action="sendFile" method="post" enctype="multipart/form-data">
```

```
    <input type="file" name="images" id="images" multiple>
```

```
    <input type="submit" value="Send" class="btn btn-primary">
  </form>
```

```
<hr>
```

```
<div id="preview">
```

```
</div>
```

</body>

<script src="js/script.js"></script>

</html>

single.ejs

<!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>

<%=title%> </title>

</head>

<body>

<form action="sendSingleFile" method="post" enctype="multipart/form-data">

<input type="file" name="images" id="images">

<input type="submit" value="Send" class="btn btn-primary">

</form>

<hr>

<div id="preview">

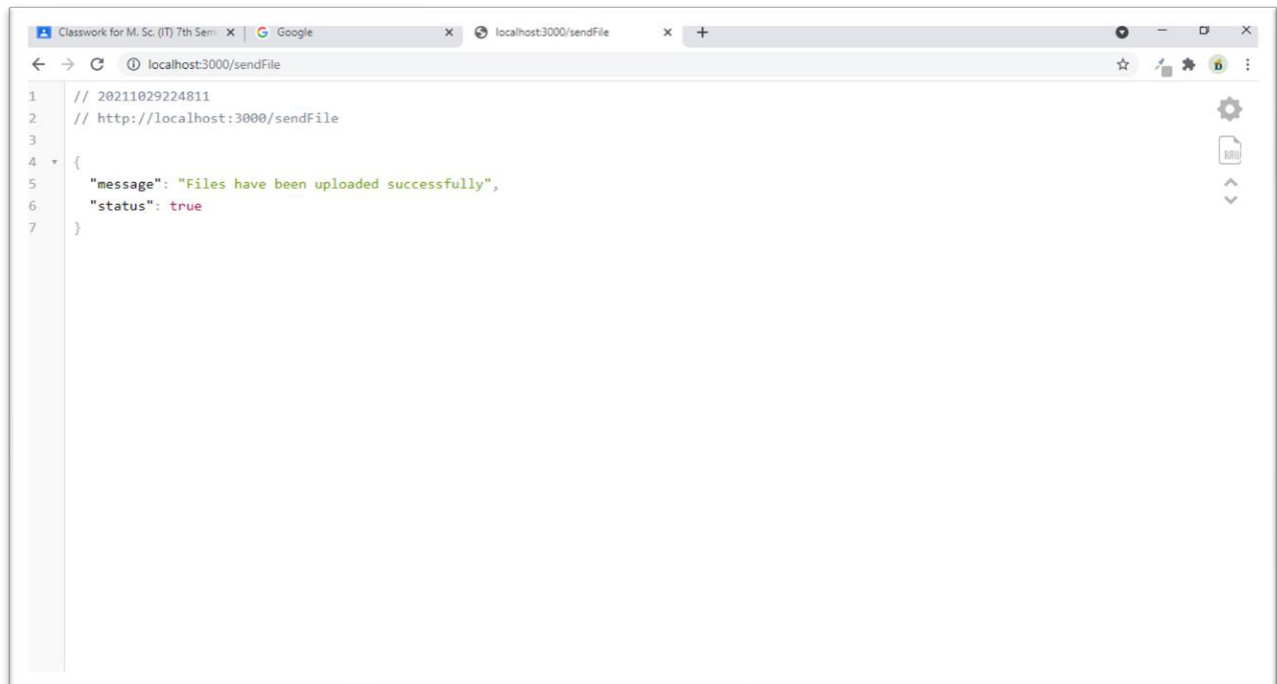
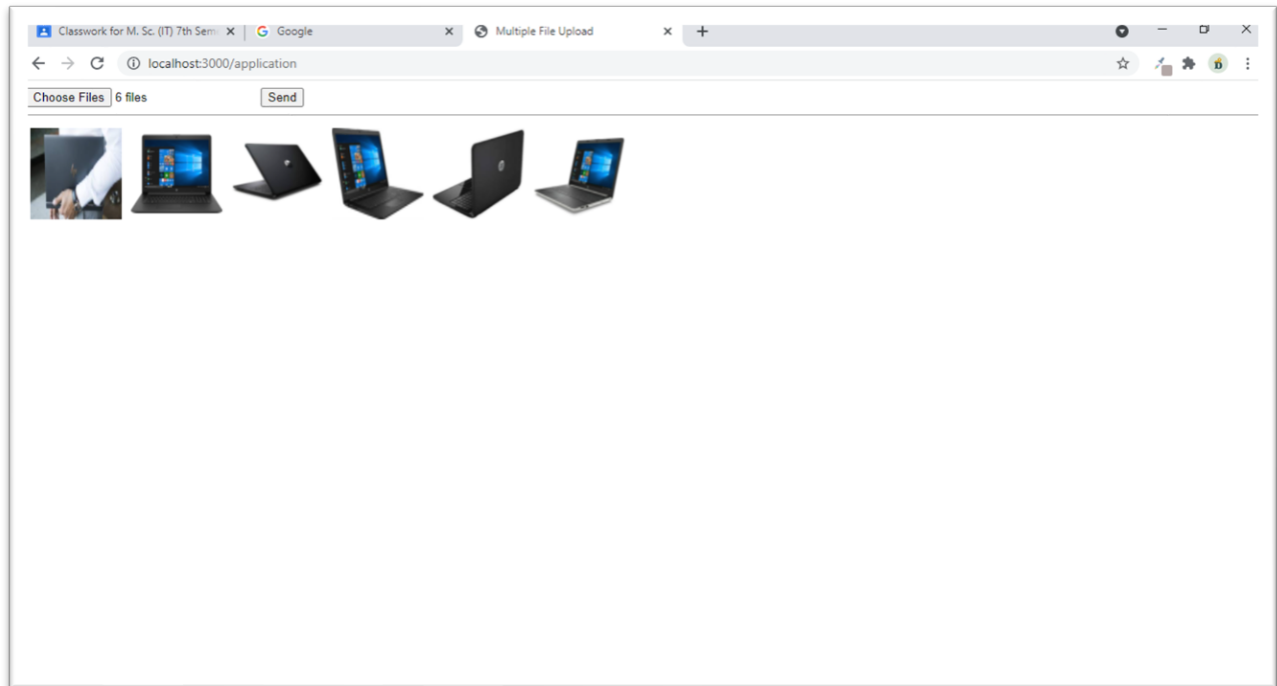
</div>

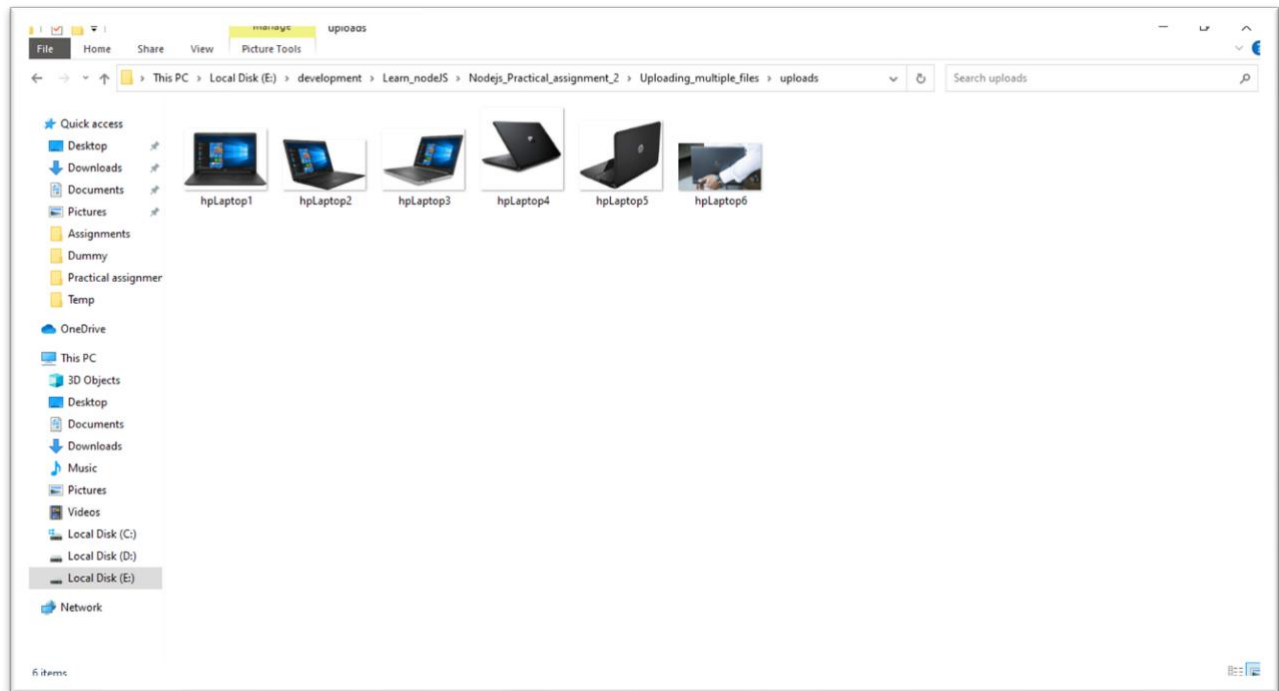
</body>

<script src="js/script.js"></script>

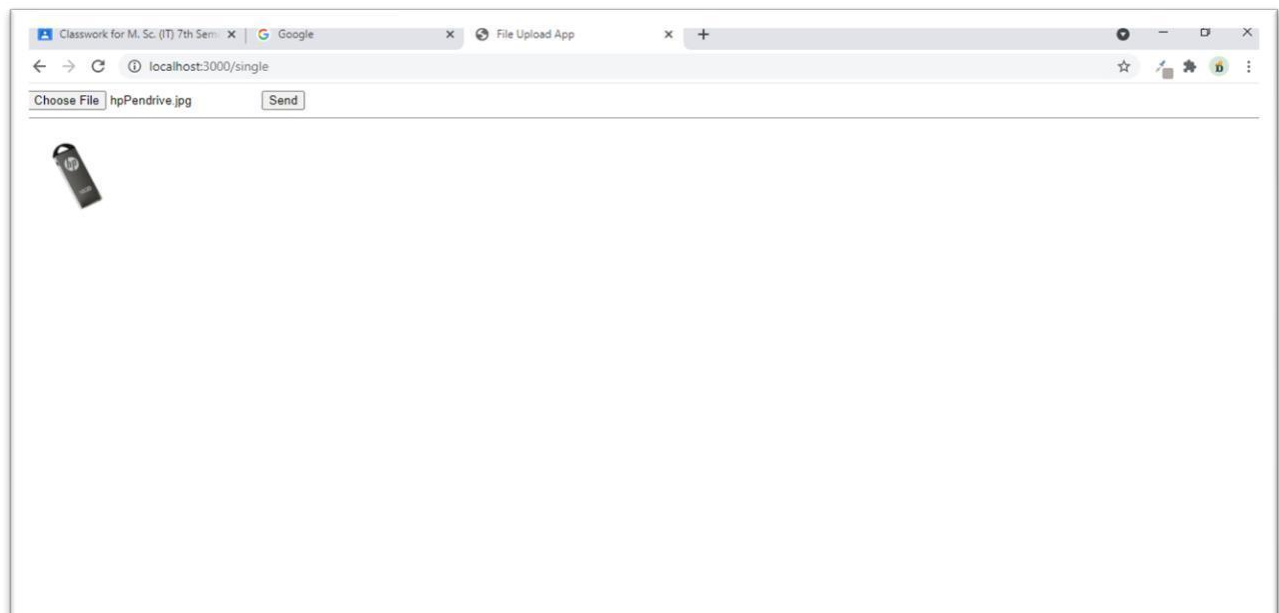
</html>

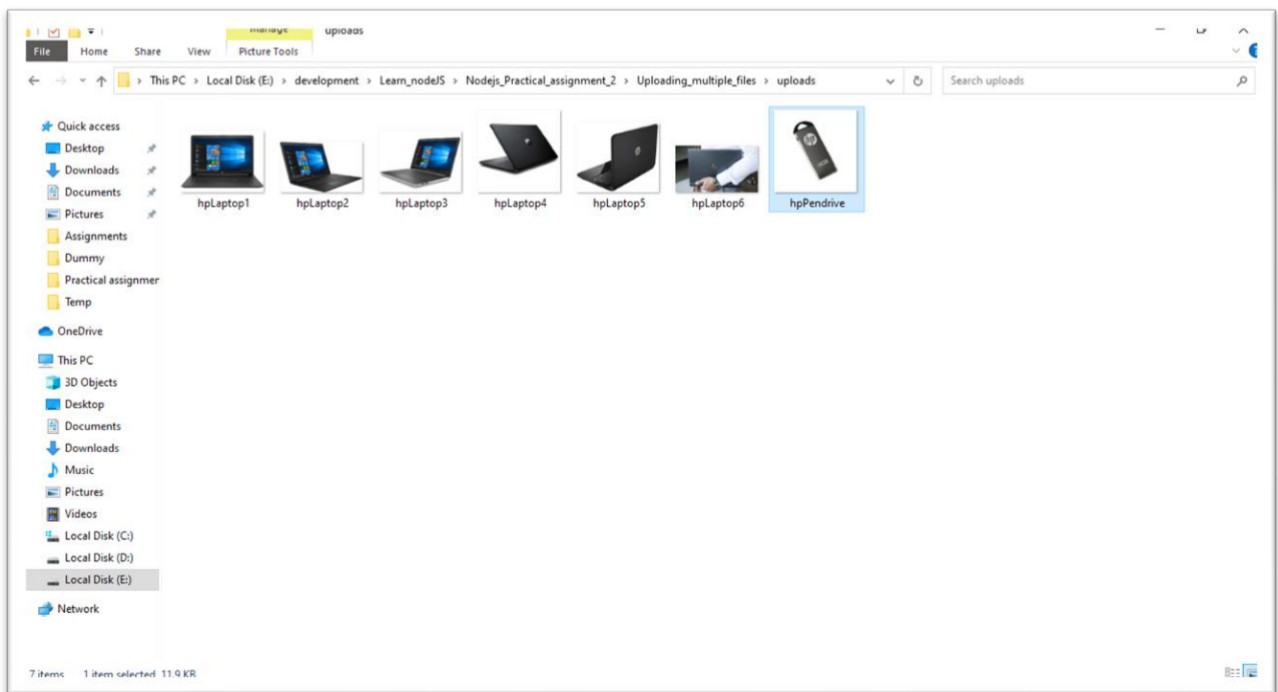
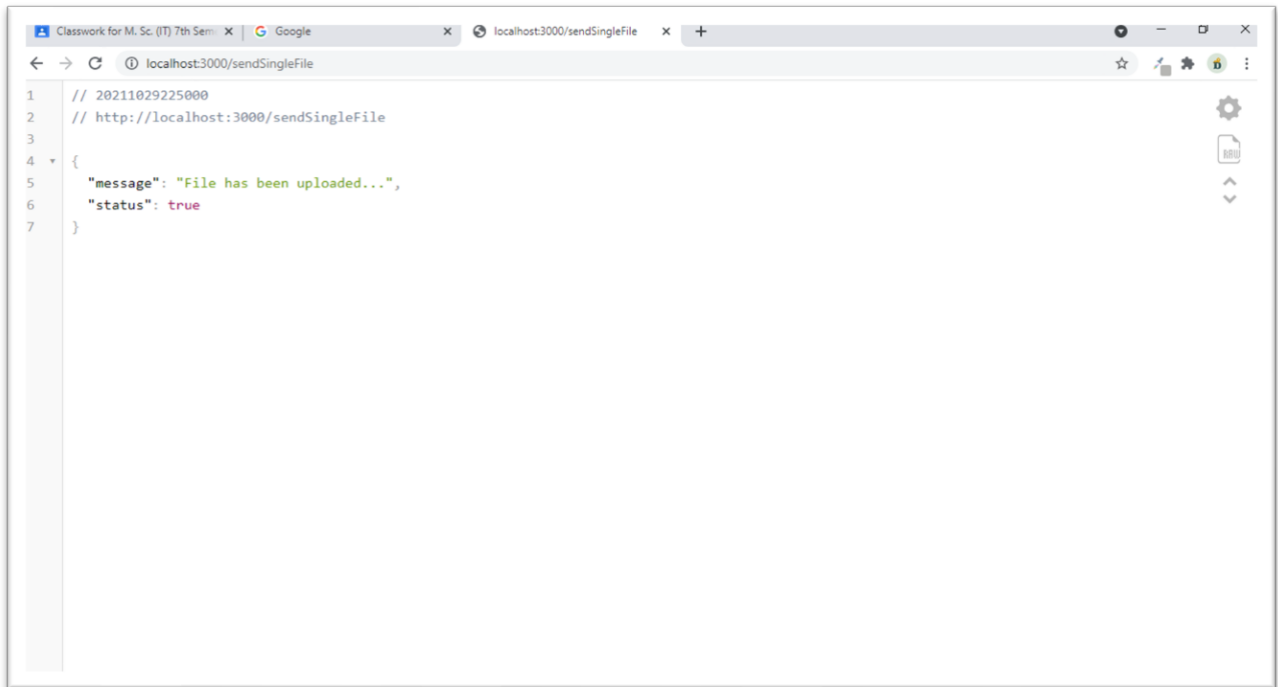
Upload Multiple Files





Upload Single Files





2) Express Login application with file session store

app.js

```
const express = require("express");
const app = express();
const mongoose = require("mongoose");

require("dotenv/config");
const path = require("path");

const session = require("express-session"); const
cookieParser = require("cookie-parser");

const FileStore = require("session-file-store")(session);

const SESSION_HOUR = 2 * 60 * 60;

const {
  PORT = 3000,

  SOME_SECRET = "You know this is a secret",
  SESSION_NAME = "connect.sid", SESSION_LIFE =
  SESSION_HOUR,

  NODE_ENV = "development",
} = process.env;

const IN_PROD = NODE_ENV === "production";

//   Middleware
app.use(express.urlencoded({extended:false}));
app.use(cookieParser());
app.use(express.static(path.join(__dirname,"assets")));
app.set("view engine","ejs");

// session setting

app.use(session({
  store:new FileStore,
  secret:SOME_SECRET,

  name:SESSION_NAME,
  resave:false,
```

```
    saveUninitialized:false,  
  
    cookie:{  
      maxAge: SESSION_LIFE,  
      secure:IN_PROD,  
  
      sameSite:true  
    }  
  });  
  
  app.get("/login",(req,res)=>{
```

```

        res.render("index",{
            title:"Login"

        });
    })

    app.post("/send",(req,res)=>{
        const username = "Dhawal@gmail.com";
        const password = "123456";

        if(username==req.body.email && password==req.body.password)
        {

            req.session.username = req.body.email;
            res.redirect("/home");
        }

        else
        {

            res.status(400).json({message:"Invalid username or
password",status:false});
        }

    });

    app.get("/home",(req,res)=>{

        if(req.session.username)
        {

            req.session.reload((err)=>{
                res.render("home",{
                    title:"Home",

                    username:req.session.username
                });
            });

        }
        else
        {

            res.redirect("/login");
        }
    }

```

```
});  
  
app.get("/logout",(req,res)=>{  
  if(req.session.username!="")  
  {  
  
    req.session.destroy(()=>{ console.log(`Session  
      destroyed...`);  
  
    });  
  
  }  
  
  res.redirect("/login");  
});
```

```

});

mongoose.connect(process.env.db_url,{useNewUrlParser:true,useUnifiedTopology:
true}).
then(()=>{

    // Listening to server...
    app.listen(PORT,()=>{
        console.log(`Server is listening to PORT: ${PORT}`);
    });
}).catch(err=>{
    console.log(`Something went wrong while connecting to db...`);

})

const db = mongoose.connection;
db.once("open",()=>console.log("Connected to DB..."));

```

partials/header.ejs

```

<!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><%=title%></title>

    <link rel="stylesheet" href="/bootstrap/bootstrap.min.css">
</head>
<body>

```

partials/footer.ejs

```

</body>

</html>

```

Index.ejs

```

<!-- Header Start-->

```

```
<%- include('partials/header.ejs')%>
<!-- Header End -->
```

```
<div class="container">
  <form action="send" method="post">
    <div class="form-group">

      <label for="email">Email</label>

      <input type="text" name="email" id="email" class="form-control"> </div>

    <div class="form-group">
```



```

        <label for="password">Password</label>

        <input type="password" name="password" id="password" class="form-
control">
      </div>
      <div class="form-group text-center">

        <button type="submit" class="btn btn-primary">Submit</button>
      </div>
    </form>

  </div>

  <!-- Footer Start -->

  <%- include('partials/header.ejs')%>
  <!-- Footer End -->

```

home.ejs

```

  <!-- Header Start-->
  <%- include('partials/header.ejs')%>
  <!-- Header End -->

  <nav class="navbar bg-dark navbar-dark "> <ul
    class="navbar-nav d-flex">

      <li class="nav-item">

        <span class="navbar-brand"><%= username%></span>
      </li>

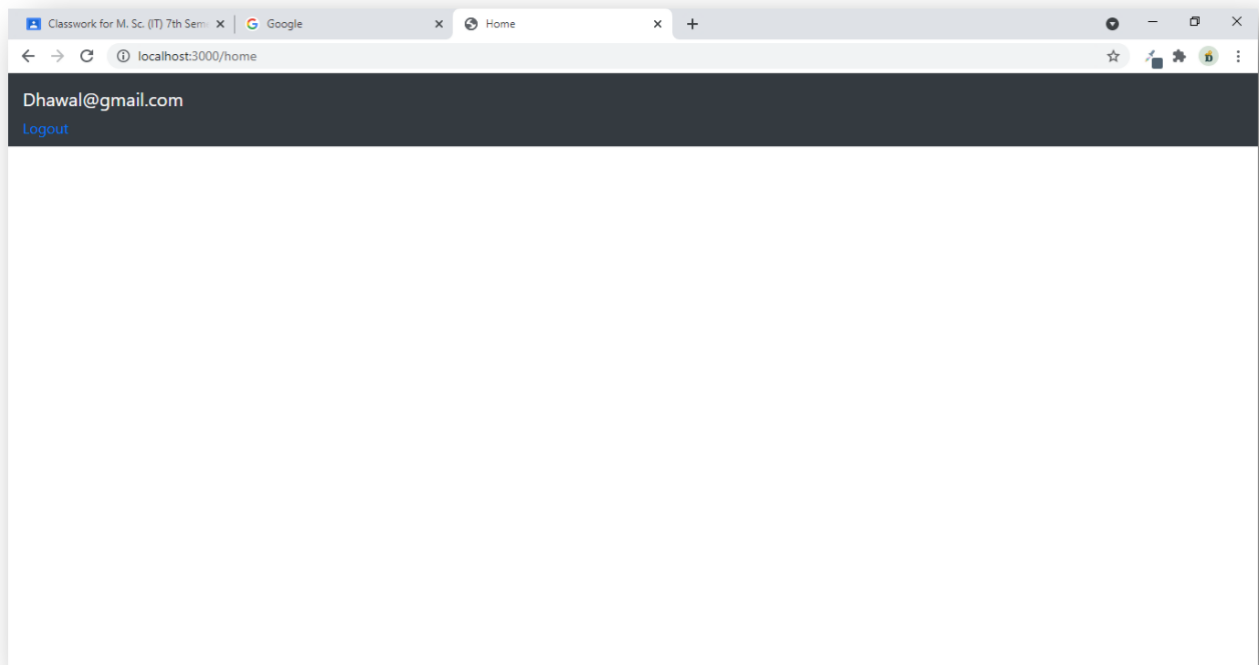
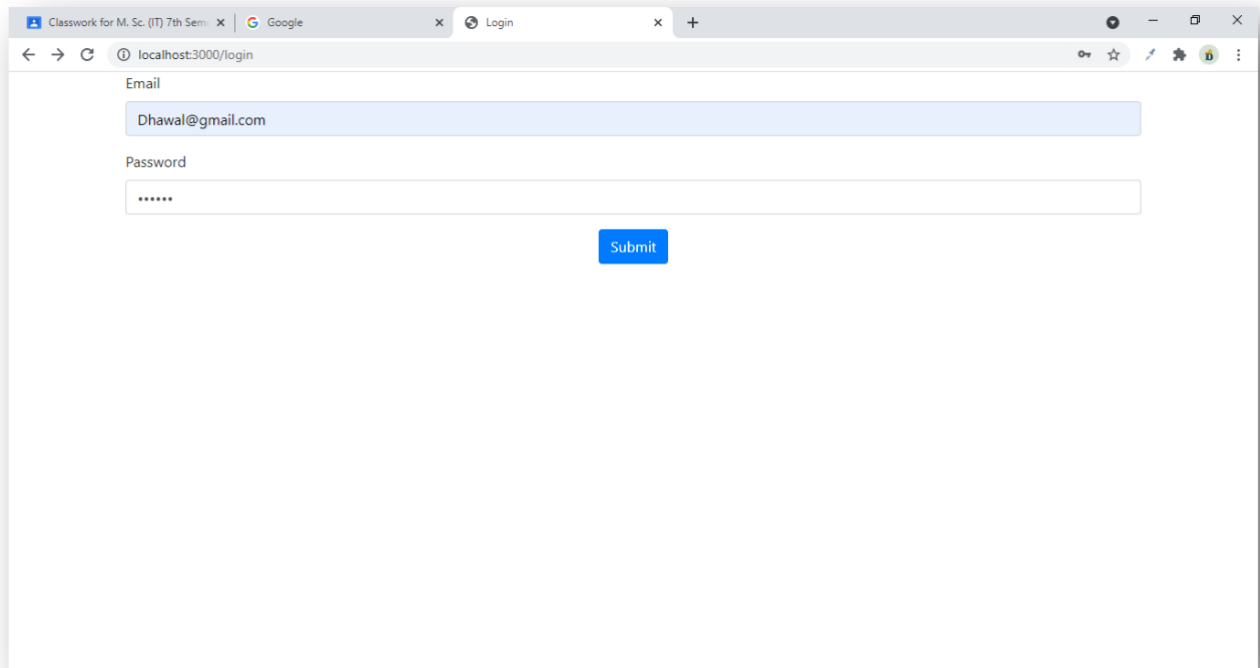
      <li class="nav-item">

        <a href="/logout" class="navbar-link">Logout</a> </li>

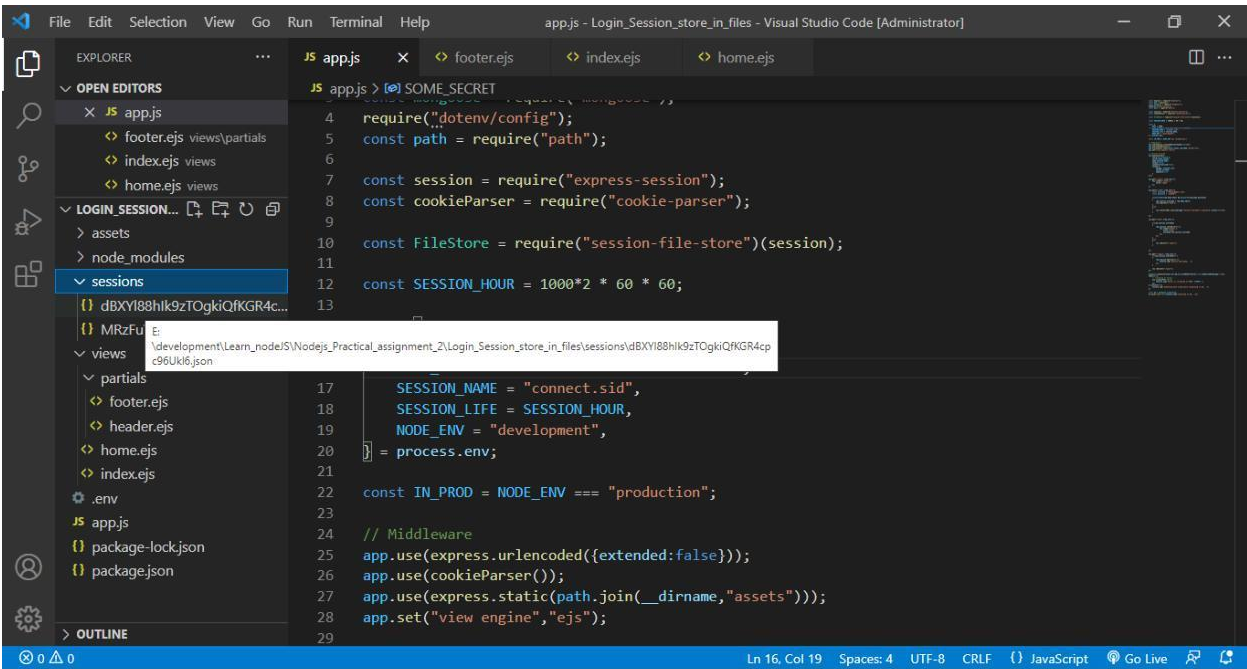
    </ul>
  </nav>

  <!-- Footer Start -->
  <%- include('partials/header.ejs')%>
  <!-- Footer End -->

```



Session Files



```
File Edit Selection View Go Run Terminal Help app.js - Login_Session_store_in_files - Visual Studio Code [Administrator]

EXPLORER
  OPEN EDITORS
    JS app.js
    footer.ejs views\partials
    index.ejs views
    home.ejs views
  LOGIN_SESSION...
    assets
    node_modules
    sessions
      {} dBXyI88hk9zTOgkiQfKGR4c...
      {} MRzFuE:
        \development\Learn_node\5\Nodejs_Practical_assignment_2\Login_Session_store_in_files\sessions\dBXyI88hk9zTOgkiQfKGR4cpz96Ukl6.json
    views
      partials
        footer.ejs
        header.ejs
        home.ejs
        index.ejs
      .env
      JS app.js
      {} package-lock.json
      {} package.json
  OUTLINE

JS app.js
1  const SOME_SECRET = require("crypto").randomBytes(64).toString("hex");
2  require("dotenv/config");
3  const path = require("path");
4  const session = require("express-session");
5  const cookieParser = require("cookie-parser");
6
7  const FileStore = require("session-file-store")(session);
8
9  const SESSION_HOUR = 1000 * 2 * 60 * 60;
10
11
12
13
14
15
16  const IN_PROD = process.env.NODE_ENV === "production";
17  const SESSION_NAME = "connect.sid",
18  SESSION_LIFE = SESSION_HOUR,
19  NODE_ENV = "development",
20  = process.env;
21
22  const IN_PROD = NODE_ENV === "production";
23
24  // Middleware
25  app.use(express.urlencoded({extended:false}));
26  app.use(cookieParser());
27  app.use(express.static(path.join(__dirname,"assets")));
28  app.set("view engine", "ejs");
29
```

3) Login with JWT, CRUD operations for students table with mongoose, express and any one template engine, Logout.

app.js

```
const express = require("express");

const router = express.Router();
const path = require("path");

const Student = require("../Models/Student"); const
verify = require("../Models/Verify");

router.get("/",verify, async (req,res)=>{ await
  Student.find().then(result=>{
    var data = {
      title:"Student List",

      username:req.session.username,
      result
    };

    res.render("../views/Students/index",{
      data

    });

  }).catch(err=>{

    console.log("Error while fetching the record..");
  });
});

router.get("/edit/:student_id",verify,async (req,res)=>{

  const _id = req.params.student_id;

  var student;

  await Student.findOne({_id}).then((result)=>{
    student=result;

  }).catch((err)=>{
    return res.status(400).json({message:"Bad Request",status:false});
```

```
});  
  
if(student==null)  
{  
  
    res.redirect("/student");  
}  
else  
  
{  
    var data = {  
        title: `Edit ${student.student_name}`,
```

```

        student,
    }

    res.render("../Views/Students/update",{ data

    });

}

});

router.get("/remove/:student_id",verify,async (req,res)=>{ const _id
    = req.params.student_id;

    await Student.deleteOne({_id}).then(result=>{
        res.redirect("/student");

    }).catch(err=>{
        res.status(400).json({message:"Something went
wrong..." ,status:false});

    });

});

router.post("/insert",verify,async (req,res)=>{

    const

    {student_name,student_email,student_age,student_gender,student_phone} =
    req.body;

    const student = new Student({

        student_name,student_email,student_age,student_gender,student_phone
    });

    await student.save().then(result=>{
    }).catch(err=>{

        console.log(err);
    });

    return res.redirect("/student");

});

router.post("/save",verify,async (req,res)=>{

```

```
const
{ _id, student_name, student_email, student_age, student_gender, student_phone } =
req.body;

Student.updateOne({ _id }, { $set: {
    student_name, student_email, student_age, student_gender, student_phone
}}).then(result => {
    console.log(result);
    return res.redirect("/student");
});
```

```

    }).catch(err=>{
      console.log(err);

      res.status(400).json({message:"Something went wrong",status:false});
    });
  });
});

router.get("/logout",(req,res)=>{

  if(req.session.username!=="")
  {
    req.session.destroy(()=>{

      console.log(`Logged out...`);
    });
  }

  res.redirect("/user/login");
});

module.exports = router;

```

The following files are from Models directory

Student.js

```

const mongoose = require("mongoose");
const Schema = mongoose.Schema;
const StudentSchema = new Schema({

  student_name:{
    type:String,
    required:[true,"Name is required"]
  },
  student_email:{
    type:String,

    unique:true,
    required:[true,"Email is required"]
  },
  student_age:{
    type:Number,

```



```
    min:[17,"Age should be more than 17"],
    max:[24,"Age should not be more than 24"],
    required:[true,"Age is required"]
  },

  student_gender:{
    type:String,
    enum:{
      values:["Male","Female"],
      message:`{value} invalid gender`
    }
  },
  student_phone:{
```

```

        type:String,
        validate:{
            validator:function(value)
            {
                return /^\d{10}$/.test(value);
            },
            message:entered=>`${entered.student_phone} is not a valid phone
number`
        },
        required:[true,"Phone is required"]
    }
});

module.exports = mongoose.model("student",StudentSchema);

```

users.js

```

const mongoose = require("mongoose");
const Schema = mongoose.Schema;

const UserSchema = new Schema({
    firstname:{
        type:String,

        required:[true,"First name is required"]
    },
    lastname:{
        type:String,
        required:[true,"Last name is required"]
    },
    email:{
        type:String,
        unique:true,

        required:[true,"Email is required"]
    },
    password:{
        type:String,

```

```
      required:[true,"Password is required"]
    },
    created_at:{
      type:Date,
      default:Date.now
    }
  });
```

```
module.exports = mongoose.model("users",UserSchema);
```

Verify.js

```
const jwt = require("jsonwebtoken");
module.exports = (req,res,next)=>{
```

```

const token = req.session.token;

if(!token)
{
    return res.redirect("/user/login");
}

try
{
    const verified = jwt.verify(token,process.env.JWT_SECRET);

    req.user = verified;
    next();
}

catch(err)
{
    console.log("Error from Verify.js" + err + ""); return
    res.redirect("/user/login");
}
}

```

The following files are from Controllers directory

StudentController.js

```

const express = require("express");
const router = express.Router();
const path = require("path");

const Student = require("../Models/Student"); const
verify = require("../Models/Verify");

router.get("/",verify, async (req,res)=>{ await
    Student.find().then(result=>{
        var data = {

            title:"Student List",
            username:req.session.username,
            result

```

```
};

res.render("../views/Students/index",{

    data

});

}).catch(err=>{
    console.log("Error while fetching the record..");
});

});
```

```

router.get("/edit/:student_id",verify,async (req,res)=>{

    const _id = req.params.student_id;

    var student;

    await Student.findOne({_id}).then((result)=>{
        student=result;

    }).catch((err)=>{
        return res.status(400).json({message:"Bad Request",status:false});
    });

    if(student==null)
    {

        res.redirect("/student");
    }
    else

    {
        var data = {
            title: `Edit ${student.student_name}`,

            student,
        }

        res.render("../Views/Students/update",{ data

        });

    }
});

router.get("/remove/:student_id",verify,async (req,res)=>{ const _id
= req.params.student_id;

    await Student.deleteOne({_id}).then(result=>{
        res.redirect("/student");

    }).catch(err=>{

        res.status(400).json({message:"Something went
wrong...",status:false});
    });

```

```
});
```

```
router.post("/insert",verify,async (req,res)=>{
```

```
    const  
    {student_name,student_email,student_age,student_gender,student_phone} =
```

```
req.body;
```

```
    const student = new Student({
```

```
        student_name,student_email,student_age,student_gender,student_phone  
    });
```

```

    await student.save().then(result=>{
    }).catch(err=>{

        console.log(err);
    });

    return res.redirect("/student");

});

router.post("/save",verify,async (req,res)=>{

    const
    {_id,student_name,student_email,student_age,student_gender,student_phone} =
    req.body;

    Student.updateOne({_id},{ $set:{
        student_name,student_email,student_age,student_gender,student_phone

    }}).then(result=>{
        console.log(result);
        return res.redirect("/student");

    }).catch(err=>{
        console.log(err);

        res.status(400).json({message:"Something went wrong",status:false});

    });

});

router.get("/logout",(req,res)=>{
    if(req.session.username!=="")

    {
        req.session.destroy(()=>{
            console.log(`Logged out...`);

        });
    }
    res.redirect("/user/login");

});

module.exports = router;

```


UserController.js

```
const express = require("express"); const
router = express.Router(); const bcrypt =
require("bcryptjs"); const User =
require("../Models/Users"); const mongoose =
require("mongoose"); const jwt =
require("jsonwebtoken");
```

```
router.get("/create",(req,res)=>{
```

```

        res.render("../Views/Users/Create",{
            title:"Create"

        });
    });

    router.get("/login",(req,res)=>{
        res.render("../Views/Users/Login",{
            title:"Login"

        });
    });

    router.post("/login",async (req,res)=>{
        const {email,password:plainTextPassword} = req.body;

        const user = await User.findOne({email});

        //    Check user is existed or not
        if(!user)
        {
            return res.status(200).json({message:"Invalid
Username/Password",status:false});
        }

        // Compare plainTextPassword with hash...
        if(!await bcrypt.compare(plainTextPassword,user.password))
        {
            return res.status(200).json({message:"Invalid
password",status:false});
        }

        const token = jwt.sign({
            id:user.id,

            email:user.email
        },process.env.JWT_SECRET);

        req.session.username = user.email;
        req.session.token = token;

        res.status(200).json({message:"Got the token",status:true});
    });

```

```
router.post("/create", async (req, res) => {  
  const {firstname, lastname, email, password: plainTextPassword} = req.body;  
  
  //Check email is existed or not  
  const existed = await User.findOne({email});  
  
  if(existed)  
  {
```

```

        return res.status(200).json({message:"Email is already
existed...",status:false})

    }

    const salt = await bcrypt.genSalt(10);

    const hashPassword = await bcrypt.hash(plainTextPassword,salt);

    console.log(hashPassword);

    const user = new User({
        firstname,

        lastname,
        email,
        password:hashPassword

    });

    await user.save().then((result)=>{

        if(result)
        {

            res.status(200).json({message:"Account has been
created...",status:true});
        }
    }).catch(err=>{

        res.status(400).json({message:"Bad Request",status:false});
    })

    //res.status(200).json({message:"Under the development
stage...",status:false});

});

module.exports = router;

```

The following files are from Views Directory

Student/index.ejs

<!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">
  <link rel="stylesheet" href="/bootstrap/bootstrap.min.css">

  <title><%=data.title%></title>
  <style>
    .cancel_button{

      background-color: #616A6B;
      color: white;
```

```

    }
    .cancel_button:hover{

        background-color: #424949;
        color: white;
    }

</style>
</head>
<body>

    <nav class="navbar navbar-expand-sm bg-dark navbar-dark"> <ul
        class="navbar-nav">
        <li class="navbar-item">

            <span class="navbar-text"><%=data.username%></span>
        </li>
        </ul>

        <ul class="navbar-nav ml-auto">
        <li class="navbar-item">

            <a href="student/logout" class="nav-link">Logout</a> </li>
        </ul>
    </nav>

    <div class="container">

        <div class="float-right">

            <button id="add" class="btn btn-outline-primary"
data-toggle="modal" data-target="#student_modal" data-backdrop="static"
type="button">Add Student</button>

        </div>
    </div>

    <form id="add-student" action="/student/insert" method="post"> <div
class="modal fade" id="student_modal" role="dialog">

        <div class="modal-dialog modal-lg modal-dialog-centered"> <div
class="modal-content">

            <!-- Modal header -->
            <div class="modal-header">

```

```
        <h4 class="modal-title">Add Student</h4>
        <button class="close" type="submit" data-
dismiss="modal">&times;</button>
    </div>

    <!-- Modal Body -->
    <div class="modal-body">

        <div class="form-group">
            <label for="student_name">Name</label>
            <input type="text" name="student_name"
id="student_name" class="form-control">
        </div>
        <div class="form-group">
```

```

        <label for="student_email">Email</label>
        <input type="email" name="student_email"

id="student_email" class="form-control">
    </div>
    <div class="form-group">

        <label for="student_age">Age</label> <input
        type="number" name="student_age"
id="student_age" class="form-control">

    </div>
    <div class="form-group">

        <label for="student_gender">Gender</label>
        <input type="radio" name="student_gender"

id="student_gender" value="Male">Male

        <input type="radio" name="student_gender"
id="student_gender" value="Female">Female
    </div>
    <div class="form-group">

        <label for="student_phone">Phone</label>
        <input type="text" name="student_phone"

id="student_phone" class="form-control">
    </div>
</div>

<!-- Modal Footer -->

<div class="modal-footer justify-content-start"> <button
    type="submit" class="btn btn-

primary">Submit</button>

    <button type="button" id="cancelBtn" data-dismiss="modal"
class="btn btn-light cancel_button">Cancel</button>

    </div>
</div>
</div>
</div>

```



```
</form>
```

```
<div class="container mt-3">
```

```
  <table class="table">
    <thead class="thead-light">
      <tr>
        <th>Name</th>
        <th>Email</th>
        <th>Age</th>
        <th>Gender</th>
        <th>Phone</th>
        <th></th>
      </tr>
    </thead>
```

```

<tbody id="tbody">
  <% if(data.result.length>0) { %>

    <%data.result.forEach(student=>{%>
      <tr>
        <td><%=student.student_name%></td>

        <td><%=student.student_email%></td>
        <td><%=student.student_age%></td>
        <td><%=student.student_gender%></td>

        <td><%=student.student_phone%></td>
        <td>
          <a href="student/edit/<%=student._id%>"
class="btn btn-primary">Edit</a>
          |
          <a href="student/remove/<%=student._id%>"
onclick="return confirm('Are you sure?')" class="btn btn-danger">Remove</a> </td>
        </tr>

        <%}}%>
      <% } else { %>
        <tr>

          <td colspan="6" align="center">No records found</td>
        </tr>
      <%}}%>

    </tbody>
  </table>
</div>

```

```

</body>

```

```

<script src="/jquery/jquery.js"></script>

```

```

<script src="/bootstrap/bootstrap.min.js"></script>

```

```

<script src="/jquery/popper.min.js"></script>

```

```

</html>

```

[Student/update.ejs](#)

```
<!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><%= data.title%></title>
  <link rel="stylesheet" href="/bootstrap/bootstrap.min.css">

</head>
<body>
  <form id="add-student" action="/student/save" method="post">
```

```

<div class="modal fade" id="student_modal" data-backdrop="static"
data-keyboard="false" role="dialog">

    <div class="modal-dialog modal-lg modal-dialog-centered"> <div
class="modal-content">

        <!-- Modal header -->
        <div class="modal-header">

            <h4 class="modal-title">Edit Student</h4>
        </div>

        <!-- Modal Body -->

        <div class="modal-body">
            <div class="form-group">
                <input type="hidden" name="_id"
value="<%=data.student._id%>">
                <label for="student_name">Name</label>
                <input type="text" name="student_name"
id="student_name" class="form-control"
value="<%=data.student.student_name%>">
            </div>

            <div class="form-group">

                <label for="student_email">Email</label>
                <input type="email" name="student_email"
id="student_email" class="form-control"
value="<%=data.student.student_email%>">
            </div>

            <div class="form-group">

                <label for="student_age">Age</label> <input
type="number" name="student_age"
id="student_age" class="form-control" value="<%=data.student.student_age%>"> </div>
                <div class="form-group">

                    <label for="student_gender">Gender</label>
                    <input type="radio" name="student_gender"

```

id="student_gender" value="Male"

<%=data.student.student_gender=="Male"? "Checked":""%>>Male

<input type="radio" name="student_gender"

id="student_gender" value="Female"

<%=data.student.student_gender=="Female"? "Checked":""%>>Female

</div>

<div class="form-group">

<label for="student_phone">Phone</label>

<input type="text" name="student_phone"

id="student_phone" class="form-control"

value="<%=data.student.student_phone%>">

</div>

</div>

<!-- Modal Footer -->

<div class="modal-footer justify-content-start">

```
        <button type="submit" class="btn btn-  
primary">Save</button>
```

```
    </div>  
  </div>  
</div>  
  
</div>  
</form>  
</body>
```

```
<script src="/jquery/jquery.js"></script>
```

```
<script src="/bootstrap/bootstrap.min.js"></script>  
<script src="/jquery/popper.min.js"></script> <script  
type="text/javascript">
```

```
    $(document).ready(function(){  
        $("#student_modal").modal("show");  
  
    });  
</script>
```

```
</html>
```

Users/create.ejs

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

```
    <meta charset="UTF-8">
```

```
    <meta name="viewport" content="width=device-width, initial-scale=1.0">  
    <title><%=title%></title>
```

```
    <link rel="stylesheet" href="/bootstrap/bootstrap.min.css">  
</head>  
<body>
```

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

```
            <h3 class="text-center text-dark">Register</h3>  
        </div>  
        <form id="register-user">  
            <div class="form-group">
```

```
        <label for="firstname">First name</label>

        <input type="text" name="firstname" id="firstname"
class="form-control">

    </div>

    <div class="form-group">

        <label for="lastname">Last name</label>
        <input type="text" name="lastname" id="lastname" class="form-
control">

    </div>

    <div class="form-group">

        <label for="email">Email</label>
```

```

        <input type="email" name="email" id="email" class="form-
control">

    </div>

    <div class="form-group">

        <label for="password">Password</label>

        <input type="password" name="password" id="password"
class="form-control">

    </div>

    <div class="form-group">

        <label for="cpassword">Confirm Password</label>

        <input type="cpassword" name="cpassword" id="cpassword"
class="form-control">

    </div>

    <div class="form-group">

        <button type="submit" class="btn btn-primary">Submit</button> <a
href="login">Already have an account?</a>

    </div>

</form>
</div>
</body>

<script src="/jquery/jquery.js"></script>
<script type="text/javascript">

    $(document).ready(=>{

        $("#register-user").on("submit",(e)=>{
            e.preventDefault();

            var formData = new FormData(e.target);
            var object = {};
            formData.forEach((value,key)=>object[key]=value);

            $.ajax({

```



```
url:`create`,

type:"POST",
data:JSON.stringify(object),
contentType:"application/json",

processData:false,
dataType:"JSON",
success:(result)=>{

    alert(result.message);
    if(result.status)
    {

        //    Redirect to Login Page
        window.location.href = "login";

    }

},
error:(err)=>{
    if(!err.responseJSON.status)
```

```

        {
            alert(err.responseJSON.message);
        }
    }
});

});
</script>

</html>

Users/login.ejs

<!DOCTYPE html>

<html lang="en">
<head>
    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title><%=title%></title>
    <link rel="stylesheet" href="/bootstrap/bootstrap.min.css">

</head>
<body>
    <div class="container">

        <div class="header">

            <h3 class="text-center text-dark">Login</h3>
        </div>

        <form id="login-form">

            <div class="form-group">

                <label for="email">Email</label>

                <input type="email" name="email" class="form-control"
placeholder="Email" id="email">

            </div>

            <div class="form-group">

```

```
        <label for="password">Password</label>
        <input type="password" name="password"
placeholder="Password" id="password" class="form-control">
    </div>

    <div class="form-group mt-3">

        <input type="submit" value="Login" class="btn btn-primary"> <a
href="create">Don't have an account?</a>
    </div>

</form>
</div>
</body>

<script src="/jquery/jquery.js"></script>
<script type="text/javascript">

    $(document).ready(=>{

        $("#login-form").on("submit",function(e){
```

```

e.preventDefault();

var formData = new FormData(e.target);
var object = {};

    formData.forEach((value,key)=>object[key]=value);

$.ajax({

    url:'login',
    type:"POST",
    data:JSON.stringify(object),

    processData:false,
    contentType:"application/json",
    dataType:"json",

    success:function(result)
    {
        alert(result.message);

        if(result.status)
        {
            window.location.href="/student";

        }
    },
    error:function(err)

    {
        if(!err.responseJSON.status)
        {

            alert(err.responseJSON.message);

        }
    }

});
});

```

```

</script>
</html>

```

Creating an account

The screenshot shows a web browser window with the address bar displaying `localhost:3000/user/create`. The form contains the following fields and values:

- First name: Dhawal
- Last name: Parmar
- Email: Dhawal@gmail.com
- Password: 123456
- Confirm Password: 123456

A blue "Submit" button is at the bottom left, and a link "Already have an account?" is at the bottom right. A modal dialog box is open in the center, displaying the message: "localhost:3000 says: account has been created." with an "OK" button.

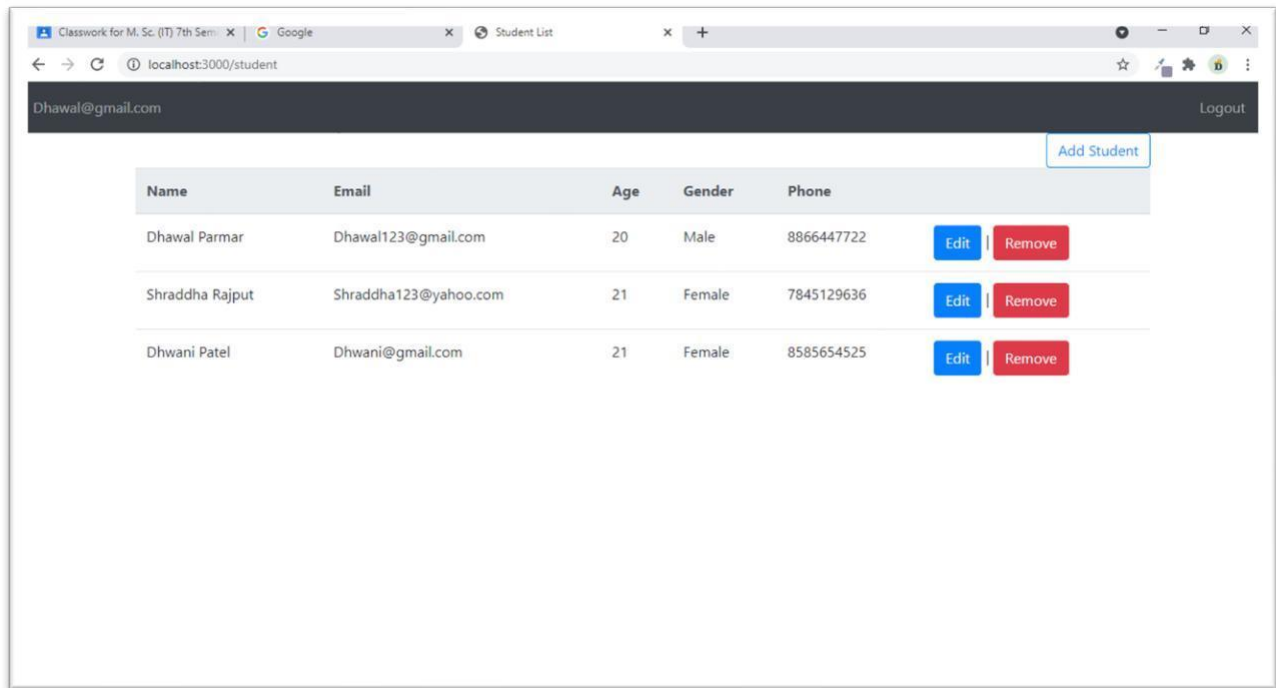
Login

The screenshot shows a web browser window with the address bar displaying `localhost:3000/user/login`. The form contains the following fields and values:

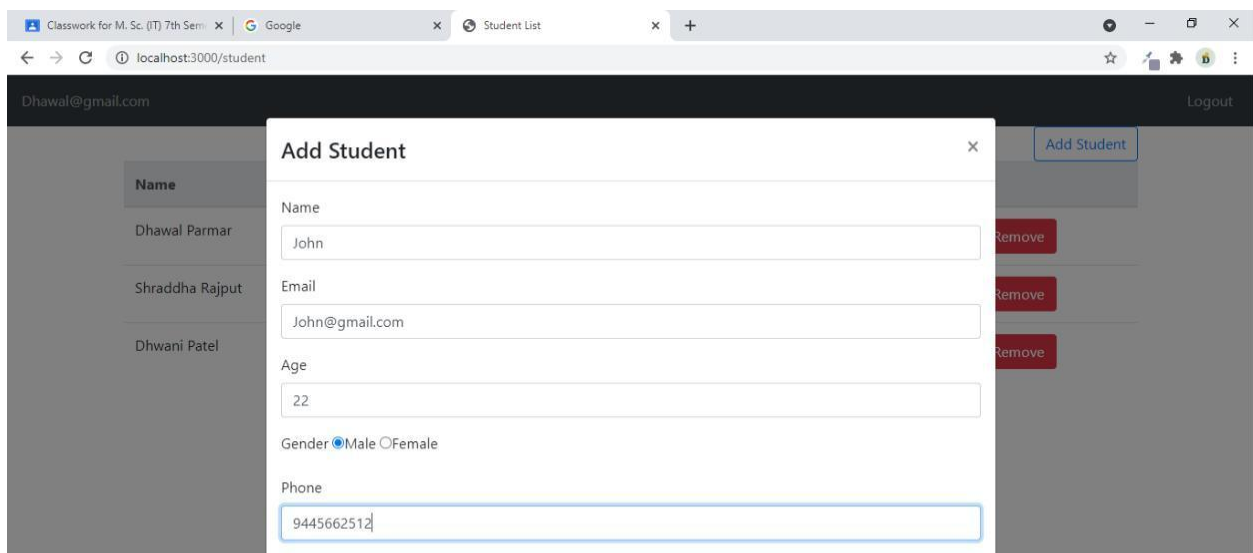
- Email: Dhawal@gmail.com
- Password: 123456

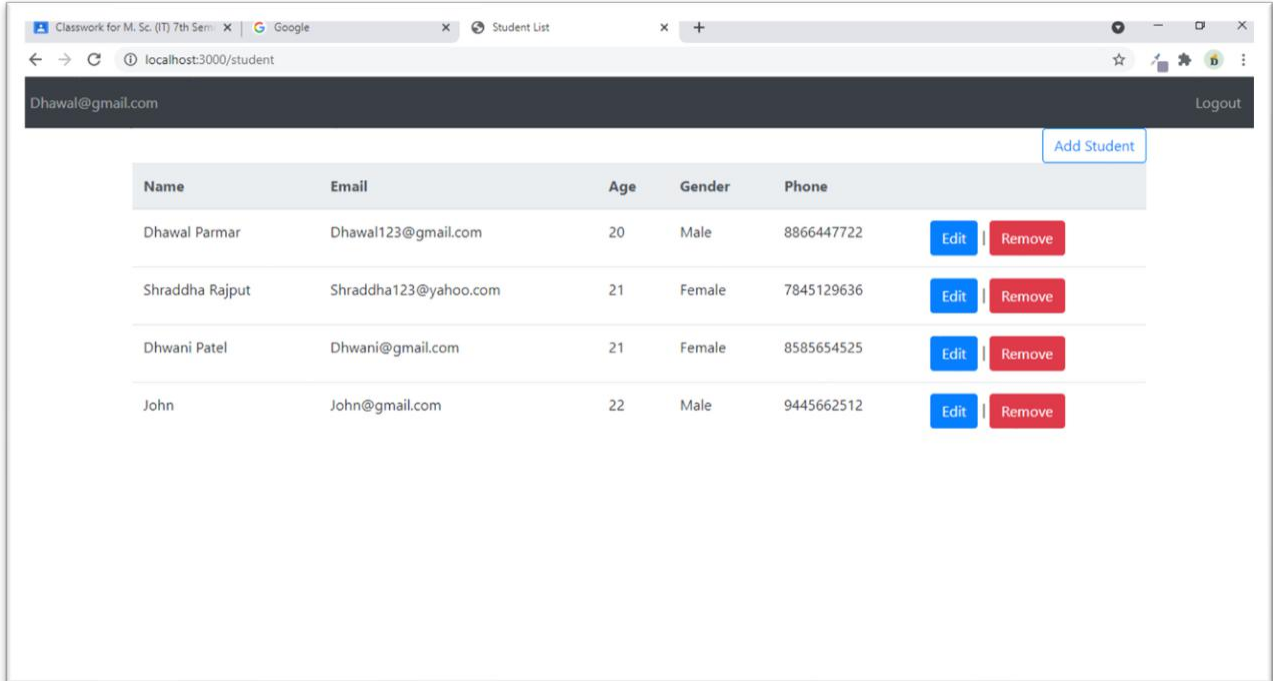
A blue "Login" button is at the bottom left, and a link "Don't have an account?" is at the bottom right. A modal dialog box is open in the center, displaying the message: "localhost:3000 says: Got the token" with an "OK" button.

List of students

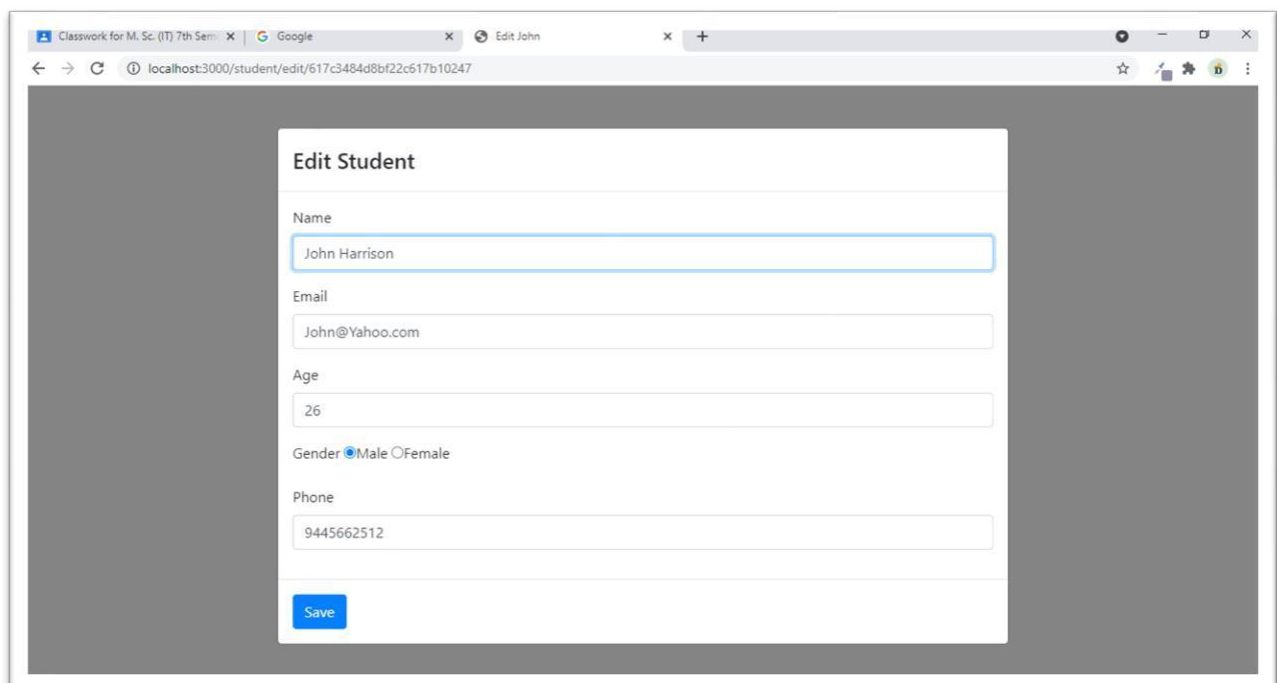


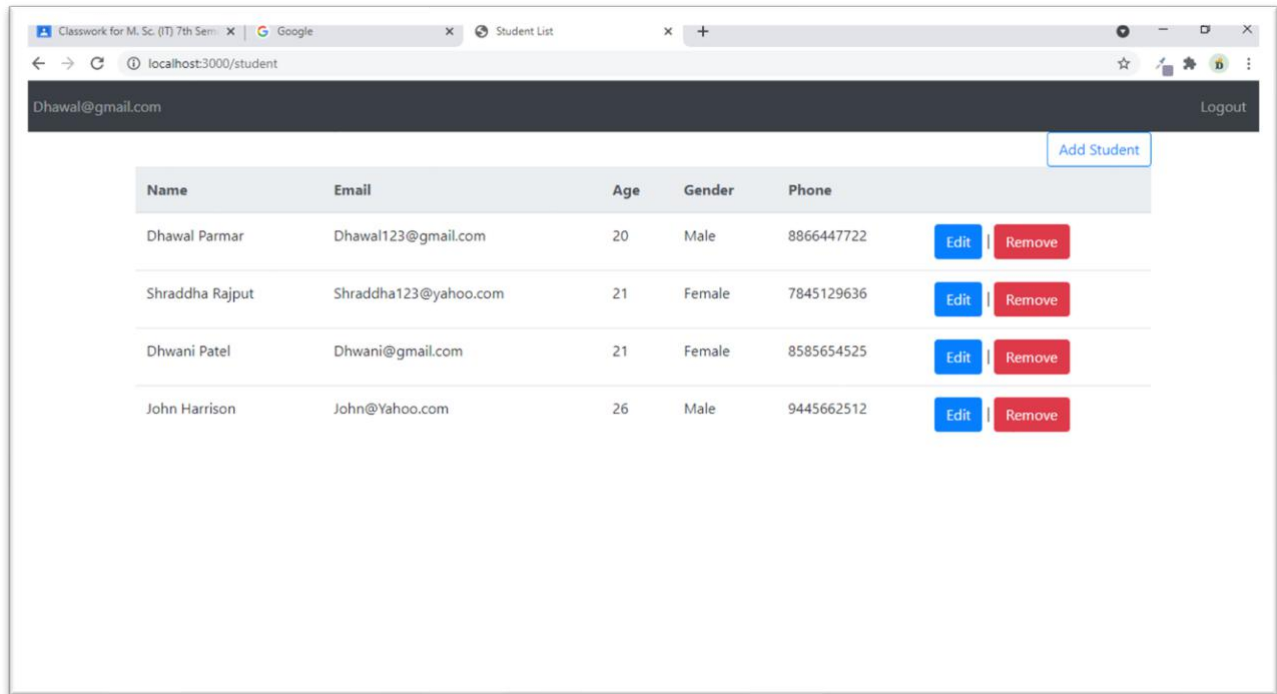
Adding the student



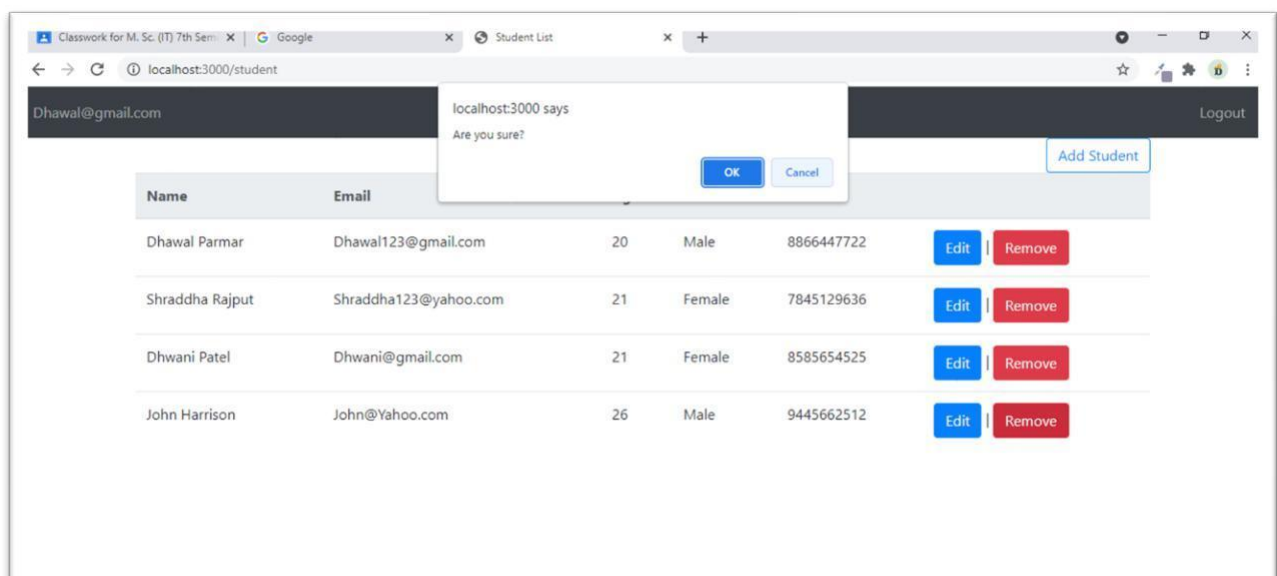


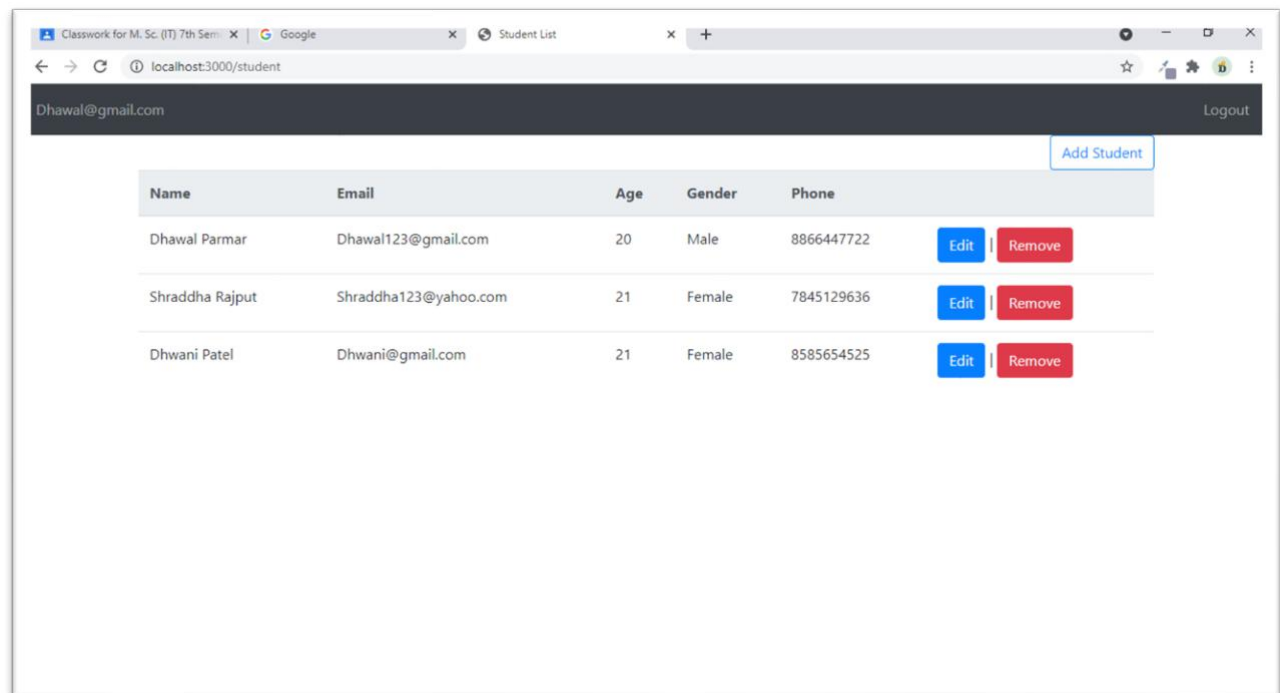
Edit the student





Removing the student





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

In frontend I've used html,css,jquery

app.js

```
const express = require("express");
const app = express();
const mongoose = require("mongoose");

const {PORT = 3000} = process.env;
const path = require("path");
const cors = require("cors");

require("dotenv/config");

const UserController = require("./Controllers/UserController"); const
StudentController = require("./Controllers/StudentController");

const cookieParser = require("cookie-parser");

app.use(cookieParser());

app.use(express.static(path.join(__dirname,"Public")));
app.use(express.json());

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

// Controller middleware app.use(cors());
app.use("/api/user",UserController);
app.use("/api/student",StudentController);

mongoose.connect(process.env.db_url,{useNewUrlParser:true,useUnifiedTopology:
true})

.then()=>{
  app.listen(PORT,()=>{
    console.log(`Server is listening to PORT ${PORT}`);
```

```
    });  
  })  
  .catch((err)=>{  
  
    console.log(`An unknown error is occurred while connecting to database...  
    ${err}`);  
  });  
  
const db = mongoose.connection;  
db.once("open",()=>console.log(`Connected to DB....`));
```

The following files are from modals directory

Students.js

```
const mongoose = require("mongoose");
const Schema = mongoose.Schema;
const StudentSchema = new Schema({

  student_name:{
    type:String,
    required:[true,"Student name is required"],

  },
  student_email:{
    type:String,

    unique:[true,"Email should be unique"],
    required:[true,"Student name is required"]

  },
  student_age:{
    type:Number,

    min:[17,"Age should be more than 17"],
    max:[24,"Age should not be more than 24"],
    required:[true,"Student name is required"]

  },
  student_gender:{
    type:String,
    enum:{

      values:["Male","Female"],
      message:'{VALUE} invalid Gender'

    },

    required:[true,"Student name is required"]
  },
  student_phone:{

    type:String,
    validate:{
```

```

        validator:function(value)
        {
            return /^\d{10}$/.test(value);
        },
        message:entered=>`${entered.student_phone} is not valid Phone
number`
    },
    required:[true,"Student name is required"]
},
});

module.exports = mongoose.model("student",StudentSchema);

```

users.js

```
const mongoose = require("mongoose");
const Schema = mongoose.Schema;

const UserSchema = new Schema({
  firstname:{
    type:String,
    required:[true,"First name is required"]
  },
  lastname:{
    type:String,
    require:[true,"Last name is required"]
  },
  email:{
    type:String,

    unique:true,
    require:[true,"Email is required"]
  },
  password:{
    type:String,
    require:[true,"Password is required"]
  },
  created_at:{
    type:Date,

    default:Date.now
  }
});

module.exports = mongoose.model("users",UserSchema);
```

verifyToken.js

```
const jwt = require("jsonwebtoken");

module.exports = (req,res,next) =>{
  const token = req.cookies["jwt_token"];

  if(!token)
```

```
{  
  
    return res.status(401).json({message:"Access denied",status:false});  
}  
  
try  
{  
    const verified = jwt.verify(token,process.env.JWT_SECRET);  
  
    req.user = verified;  
    next();  
}
```

```

        catch(err)
        {

            return res.status(401).json({message:"Invalid
token...",status:false});

        }

    }
}

```

The following files are from Controllers directory

UserController.js

```

const express = require("express");
const router = express.Router();
const path = require("path");

const bcrypt = require("bcryptjs");
const jwt = require("jsonwebtoken");
require("dotenv/config");

const User = require("../Models/users");
const verifyToken = require("../Models/verifyToken");

// One hour
const {MAX_AGE = 1000 * 60 * 60 * 1} = process.env;

router.post("/login",async (req,res)=>{

    const {email,password:plainTextPassword} = req.body;

    // Verify whether user is existed or not...

    const user = await User.findOne({email:email});
    console.log(user);

    if(!user)

    {

        return res.status(400).json({message:"Invalid
username/Password",status:false});

    }

    // Compare plainTextPassword with hash...

```



```
if(!(await bcrypt.compare(plainTextPassword,user.password)))
{
    return res.status(400).json({message:"Invalid
password",status:false});
}

const token = jwt.sign({
    id:user_id,
    email:user.email
},process.env.JWT_SECRET);

res.cookie("jwt_token",token,{httpOnly:true,secure:true,maxAge:MAX_AGE});
```

```
    return res.status(200).json({message:"Got the  
token",data:token,status:true}); });
```

```
router.get("/login",(req,res)=>{  
    res.sendFile(path.join(__dirname,"../Views/Users/index.html"));  
});
```

```
router.get("/create",(req,res)=>{  
  
    res.sendFile(path.join(__dirname,"../Views/Users/create.html"));  
  
});
```

```
router.post("/create",async (req,res)=>{  
  
    const existed = await User.findOne({email:req.body.email});  
  
    if(existed)  
  
    {  
  
        return res.status(400).json({message:"Email is already  
existed",status:false});  
  
    }  
  
    const salt = await bcrypt.genSalt(10);  
  
    const hashPassword = await bcrypt.hash(req.body.password,salt);  
  
    var user = new User({  
  
        firstname :req.body.firstname,  
        lastname:req.body.lastname,  
        email:req.body.email,  
  
        password:hashPassword  
    });  
  
    await user.save().then(result=>{  
        console.log(result);  
  
        return res.status(200).json({message:"Account has been  
created...",status:true});  
    }).catch(err=>{
```

```
        return res.status(400).json({message:"Unable to create an  
account...",status:false});  
    });
```

```
    //return res.status(200).json({message:"Account has been  
created...",status:true});
```

```
    /* res.status(200).json({message:"Okay",status:true});*/  
});
```

```
router.get("/test",(req,res)=>{
```

```
    if(!req.cookies.jwt_token)
```

```

    {
        res.clearCookie("jwt_token");

        return res.redirect("login");
    }
    res.sendFile(path.join(__dirname, "../Views/Users/test.html"));
});

```

```

router.post("/action",verifyToken,(req,res)=>{
    res.status(200).json({message:"Success",status:true});
});

```

```

module.exports = router;

```

StudentController.js

```

const express = require("express");

```

```

const router = express.Router();
const path = require("path");

```

```

const verify = require("../Models/verifyToken"); const
Student = require("../Models/students");

```

```

router.post("/",verify,async(req,res)=>{
    Student.find().then(result=>{
        if(result)

            {

                res.status(200).json({message:"Record
found",data:result,status:true});

            }
            else
            {

                res.status(200).json({message:"No data found",status:false});

            }
        }).catch(err=>{

            res.status(400).json({message:"Unable retrieve the
data",status:false});
        });
    });
}

```

```
});
```

```
router.delete("/:id",verify,async (req,res)=>{
```

```
    if(req.params.id!=undefined)
    {
```

```
        Student.deleteOne({_id:req.params.id}).then(result=>{
            console.log(result);
            if(result.deletedCount==1)
```

```
        {
```

```
            res.status(200).json({message:"Record has been
deleted...",status:true});
```

```

        }
        else

        {

            res.status(200).json({message:"Can't delete the
record...",status:false});

        }

    }).catch(err=>{

        res.status(400).json({message:"An unknow error is
occured...",status:false});
    });

    }
    else
    {

        res.status(200).json({message:"Something went wrong",status:false});
    }

    // res.status(200).json({message:req.params.id,status:false});
});

router.patch("/:id",verify, async (req,res)=>{

    const

    {student_id,student_name,student_email,student_age,student_gender,student_pho
ne} = req.body;

    await Student.updateOne(
        {_id:req.params.id},
        {

            $set:{
                student_name,
                student_email,

                student_age,
                student_gender,
                student_phone

            }
        }
    )
}

```

```
    ).then(result=>{  
        res.status(200).json({message:"Record has been  
updated...",status:true});  
    }).catch(err=>{  
        res.status(200).json({message:"An unknown error is  
occured...",status:false});  
    });  
});  
  
router.get("/get/:id",verify,async(req,res)=>{  
    await Student.findOne({_id:req.params.id}).then(result=>{ data =  
        ";
```

```

        data+=`<div class="modal fade" id="student_edit_modal"
role="dialog">`;

        data+=`<div class="modal-dialog modal-lg modal-dialog-centered">`;
        data+=`<div class="modal-content">`;

        // Header Start
        data+=`<div class="modal-header">`;

        data+=`<h4 class="modal-title text-center">Edit student</h4>
<button class="close" type="button" data-
dismiss="modal">&times;</button>`;
        data+=`</div>`;

        // Header End

        // Body Start

        data+=`<div class="modal-body">`;

        data+=`<div class="form-group">`; data+=`<label
for="student_name">Name</label>

        <input type="text" name="student_name" id="student_name"
value="${result.student_name}" class="form-control">`;

        data+=`</div>`;

        data+=`<div class="form-group">`; data+=`<label
for="student_email">Email</label>

        <input type="email" name="student_email" id="student_email"
value="${result.student_email}" class="form-control">`;

        data+=`</div>`;

        data+=`<div class="form-group">`; data+=`<label
for="student_age">Age</label>

        <input type="number" name="student_age" id="student_age"
value="${result.student_age}" class="form-control">`;

        data+=`</div>`;

        data+=`<div class="form-group">`;

```



```
data+=`<label for="student_gender">Gender</label>`;

if(result.student_gender=="Male")

{
    data+=`<input type="radio" name="student_gender"
id="student_gender" value="Male" checked>Male

        <input type="radio" name="student_gender" id="student_gender"
value="Female">Female`;
    }

    else
    {
        data+=`<input type="radio" name="student_gender"

id="student_gender" value="Male">Male

        <input type="radio" name="student_gender" id="student_gender"
value="Female" checked>Female`;
    }
```

```

    }

    data+='</div>';

    data+=`<div class="form-group">`; data+=`<label
for="student_phone">Phone</label>

    <input type="text" name="student_phone" id="student_phone"
value="${result.student_phone}" class="form-control">`;

    data+='</div>';

    data+='</div>';

    // Body End

    // Footer start

    data+=`<div class="modal-footer justify-content-start">

    <button class="btn btn-primary" type="submit">Save Changes</button>
    <button class="btn btn-light cancel_button" data-dismiss="modal"

id="close_edit_form" type="button">Cancel</button>
    </div>`;
    // Footer end

    data+='</div>';
    data+='</div>';

    data+=`</div>`;

    res.status(200).json({message:"Record found...",data,status:true});
    }).catch(err=>{

        res.status(200).json({message:"Something went
wrong...",status:false});
    });

});

router.get("/index",(req,res)=>{

    if(!req.cookies.jwt_token)
    {

```

```
        res.clearCookie("jwt_token");

        res.redirect("/api/user/login");
    }
    res.sendFile(path.join(__dirname, "../Views/Student/StudentList.html"));
});

router.post("/create", verify, async(req,res)=>{

    // console.log(req.body);

    //res.status(200).json({message:"Under the development
stage",status:false});

    const

    {student_name,student_email,student_age,student_gender,student_phone} =
    req.body;
```

```

const existed = await Student.findOne({student_email:student_email});

if(existed)
{
    return res.status(400).json({message:"Student's email is already
existed...",status:false});
}

const student = new Student({
    student_name,
    student_email,

    student_gender,
    student_age,
    student_phone

});

await student.save().then(result=>{

    if(result)
    {

        return res.status(200).json({message:"Record has been
inserted...",status:true});
    }
}).catch(err=>{

    console.log(err);

    return res.status(200).json({message:"Unable to insert the
record...",status:false});

});

});

router.get("/logout",(req,res)=>{
    if(req.cookies.jwt_token)

    {
        res.clearCookie("jwt_token");
    }
}

```

```
        res.redirect("/api/user/login");  
    });  
  
    module.exports = router;
```

The following files are from Views Directory

Student/StudentList.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">
  <link rel="stylesheet" href="/bootstrap/bootstrap.min.css">

  <title>Student</title>
  <style>
    .cancel_button{

      background-color: #616A6B;
      color: white;
    }

    .cancel_button:hover{
      background-color: #424949;
      color: white;
    }
  </style>
</head>

<body>

  <nav class="navbar navbar-expand-sm bg-dark navbar-dark"> <ul
    class="navbar-nav">

      <li class="navbar-item">

        <span class="navbar-text">Username</span>
      </li>

    </ul>
    <ul class="navbar-nav ml-auto">
      <li class="navbar-item">

        <a href="logout" class="nav-link">Logout</a>
      </li>
    </ul>
```

```
</nav>
```

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

```
  <div class="float-right">
```

```
    <button id="add" class="btn btn-outline-primary"
data-toggle="modal" data-target="#student_modal" data-backdrop="static"
type="button">Add Student</button>
```

```
  </div>
```

```
</div>
```

```
<form id="add-student">
```

```
<div class="modal fade" id="student_modal" role="dialog"> <div
class="modal-dialog modal-lg modal-dialog-centered">
```

```
  <div class="modal-content">
```

```

<!-- Modal header -->

<div class="modal-header">

    <h4 class="modal-title">Add Student</h4>
    <button class="close" type="submit" data-
dismiss="modal">&times;</button>
</div>

<!-- Modal Body -->
<div class="modal-body">
    <div class="form-group">

        <label for="student_name">Name</label>
        <input type="text" name="student_name"
id="student_name" class="form-control">
    </div>
    <div class="form-group">

        <label for="student_email">Email</label>
        <input type="email" name="student_email"
id="student_email" class="form-control">
    </div>

    <div class="form-group">

        <label for="student_age">Age</label> <input
type="number" name="student_age"
id="student_age" class="form-control">
    </div>
    <div class="form-group">

        <label for="student_gender">Gender</label>
        <input type="radio" name="student_gender"
id="student_gender" value="Male">Male

        <input type="radio" name="student_gender"
id="student_gender" value="Female">Female
    </div>

```



```

        <div class="form-group">

            <label for="student_phone">Phone</label>
            <input type="text" name="student_phone"

id="student_phone" class="form-control">
        </div>
    </div>

    <!-- Modal Footer -->

    <div class="modal-footer justify-content-start"> <button
        type="submit" class="btn btn-

primary">Submit</button>

        <button type="button" id="cancelBtn" data-dismiss="modal"
class="btn btn-light cancel_button">Cancel</button>
    </div>
</div>
</div>
</form>

```

```
<form id="edit-form">
```

```
</form>
```

```
<div class="container mt-3">
```

```
    <table class="table">
```

```
        <thead class="thead-light">
```

```
            <tr>
```

```
                <th>Name</th>
```

```
                <th>Email</th>
```

```
                <th>Age</th>
```

```
                <th>Gender</th>
```

```
                <th>Phone</th>
```

```
                <th></th>
```

```
            </tr>
```

```
        </thead>
```

```
        <tbody id="tbody">
```

```
        </tbody>
```

```
    </table>
```

```
</div>
```

```
</body>
```

```
<script src="/jquery/jquery.js"></script>
```

```
<script src="/bootstrap/bootstrap.min.js"></script>
```

```
<script src="/jquery/popper.min.js"></script> <script  
type="text/javascript">
```

```
    $(document).ready(function(){
```

```
        var student_id=undefined;
```

```
        $.ajax({
```

```
            url:"/api/student/",
```

```
            type:"POST",
```

```
dataType:"json",
processData:false,
success:function(result)

{
    var data = "";
    if(result.status)

    {
        var ArrayOfItems = result.data;

        ArrayOfItems.forEach(ArrayOfItem=>{
```

```

        var obj = ArrayOfItem;
        for(var key in obj)

        {
            data+=`<tr>
                <td>${obj["student_name"]}</td>

                <td>${obj["student_email"]}</td>
                <td>${obj["student_gender"]}</td>
                <td>${obj["student_phone"]}</td>

                <td>
                    <button type="button" id="editBtn" class="btn
btn-primary" data-id="${obj["_id"]}">Edit</button>

                    <button type="button" id="removeBtn"
class="btn btn-danger" data-id="${obj["_id"]}">Remove</button>
                </td>

            </tr>`;
            break;
        }

    });
    $("#tbody").html(data);
}

else
{
    alert(result.message);
}
}
});

$(document).on("click","#removeBtn",(e)=>{
    e.preventDefault();

    const id = $(e.target).attr("data-id");
    var tr = $(e.target.parentElement.parentElement);

    if(confirm("Are you sure?"))
    {
        $.ajax({
            url:`/api/student/${id}`,
            type:"delete",

```

```
dataType:"json",
processData:false,
contentType:false,

success:function(result)
{
    alert(result.message);

    if(result.status)
    {
        //      $(element).closest("tr").fadeOut();
        $(tr).fadeOut();
    }
},
```

```

        error:function(err)
        {

            alert("Something went wrong...");

        }
    });

}

});

$(document).on("click","#editBtn",(e)=>{
    e.preventDefault();

    const id = $(e.target).attr("data-id");
    student_id = id;
    var tr = $(e.target.parentElement.parentElement);

    $.ajax({
        url:`/api/student/get/${id}`,
        type:"get",

        processData:false,
        contentType:'application/json',
        dataType:"json",

        success:function(result)
        {
            // alert(result.message);

            if(result.status)
            {
                $("#edit-form").html(result.data);

                $("#student_edit_modal").modal("show");
            }
        }
    });

});

$("#add-student").on("submit",(e)=>{

    e.preventDefault();

```

```
var formData = new FormData(e.target);
var object = {};
formData.forEach((value,key)=>object[key]=value);

console.log(JSON.stringify(object));
console.log($(e.target).serialize());
$.ajax({

    url:`/api/student/create`,
    type:"post",
    data:JSON.stringify(object),

    contentType:'application/json',
    dataType:"json",
    success:function(result)
```

```

        {
            alert(result.message);

            if(result.status)
            {
                $("#student_modal").modal("hide");

                window.location.reload();
            }
        }
    });
});

$(document).on("submit","#edit-form",(e)=>{
    e.preventDefault();
    var formData = new FormData(e.target);

    var object = {};
    if(student_id)
    {

        object["student_id"] = student_id;
    }
    formData.forEach((value,key)=>object[key]=value);

    if(student_id)
    {

        $.ajax({
            url:`/api/student/${object.student_id}`,
            type:"patch",

            data:JSON.stringify(object),
            processData:false,
            contentType:'application/json',

            dataType:"json",
            success:function(result)
            {

                alert(result.message);
                if(result.status)
                {

                    $("#student_edit_modal").modal("hide");
                    window.location.reload();
                }
            }
        });
    }
});

```



```
        }  
    },  
    error:function(err)  
    {  
        console.log('An unknown error is occurred...');  
    }  
});  
  
}  
else  
{  
    alert("Can't update the record")  
}  
});
```

```
    });  
  
</script>  
</html>
```

Users/Create.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>Create user</title>  
  <link rel="stylesheet" href="/bootstrap/bootstrap.min.css">  
  
</head>  
<body>  
  
  <div class="container">  
    <div class="header">  
  
      <h3 class="text-center text-dark">Register</h3>  
    </div>  
    <form id="register-user">  
      <div class="form-group">  
  
        <label for="firstname">First name</label>  
  
        <input type="text" name="firstname" id="firstname"  
class="form-control">  
  
      </div>  
  
      <div class="form-group">  
  
        <label for="lastname">Last name</label>  
        <input type="text" name="lastname" id="lastname" class="form-  
control">  
  
      </div>  
  
      <div class="form-group">
```

```
control">  
    <label for="email">Email</label>  
    <input type="email" name="email" id="email" class="form-
```

```
</div>
```

```
<div class="form-group">
```

```
    <label for="password">Password</label>
```

```
    <input type="password" name="password" id="password"  
class="form-control">
```

```
</div>
```

```
<div class="form-group">
```

```

        <label for="cpassword">Confirm Password</label>

        <input type="cpassword" name="cpassword" id="cpassword"
class="form-control">
    </div>

    <div class="form-group">

        <button type="submit" class="btn btn-primary">Submit</button> <a
href="login">Already have an account?</a>

    </div>
</form>
</div>

```

```

</body>
<script>

```

```

    document.getElementById("register-
user").addEventListener("submit",registerUser);

```

```

function returnValue(fieldId)
{

    return document.getElementById(fieldId).value;
}

```

```

async function registerUser(event)
{
    event.preventDefault();

    let firstname = returnValue("firstname"); let
lastname = returnValue("lastname"); let email =
returnValue("email");

    let password = returnValue("password"); let
cpassword = returnValue("cpassword");

    const result = await fetch("/api/user/create",{
        method:"POST",

        headers:{
            'Content-type':'application/json'
        },

        body:JSON.stringify({

```

```
        firstname,lastname,email,password,cpassword
    })
    }).then(res=>res.json()).catch(err=>{console.log('Error is there')});

    alert(result.message);

    if(result.status)
    {
        window.location.href = "login";
    }
}
```

```
</script>
```

```
</html>
```

Users/index.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>Login</title>
```

```
  <link rel="stylesheet" href="/bootstrap/bootstrap.min.css"> <style>
```

```
  </style>
```

```
</head>
```

```
<body>
```

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

```
    <div class="header">
```

```
      <h3 class="text-center text-dark">Login</h3>
```

```
    </div>
```

```
    <form id="login-form">
```

```
      <div class="form-group">
```

```
        <label for="email">Email</label>
```

```
        <input type="email" name="email" class="form-control"
placeholder="Email" id="email">
```

```
      </div>
```

```
      <div class="form-group">
```

```
        <label for="password">Password</label>
```

```
        <input type="password" name="password">
```

```
placeholder="Password" id="password" class="form-control">
    </div>

    <div class="form-group mt-3">

        <input type="submit" value="Login" class="btn btn-primary"> <a
        href="create">Don't have an account?</a>
    </div>

    </form>
</div>
</body>

<script type="text/javascript">
    document.getElementById("login-
form").addEventListener("submit",loginForm);

    async function loginForm(event)
    {
```

```
event.preventDefault();

const email = document.getElementById("email").value; const
password = document.getElementById("password").value;

const result = await fetch("/api/user/login",{
  method:"POST",
  headers:{

    'Content-Type':'application/json',
  },
  body:JSON.stringify({

    email,password
  })
}).then(res=>{

  return res.json();
}).catch(err=>{
  console.log("Error: "+err);

});

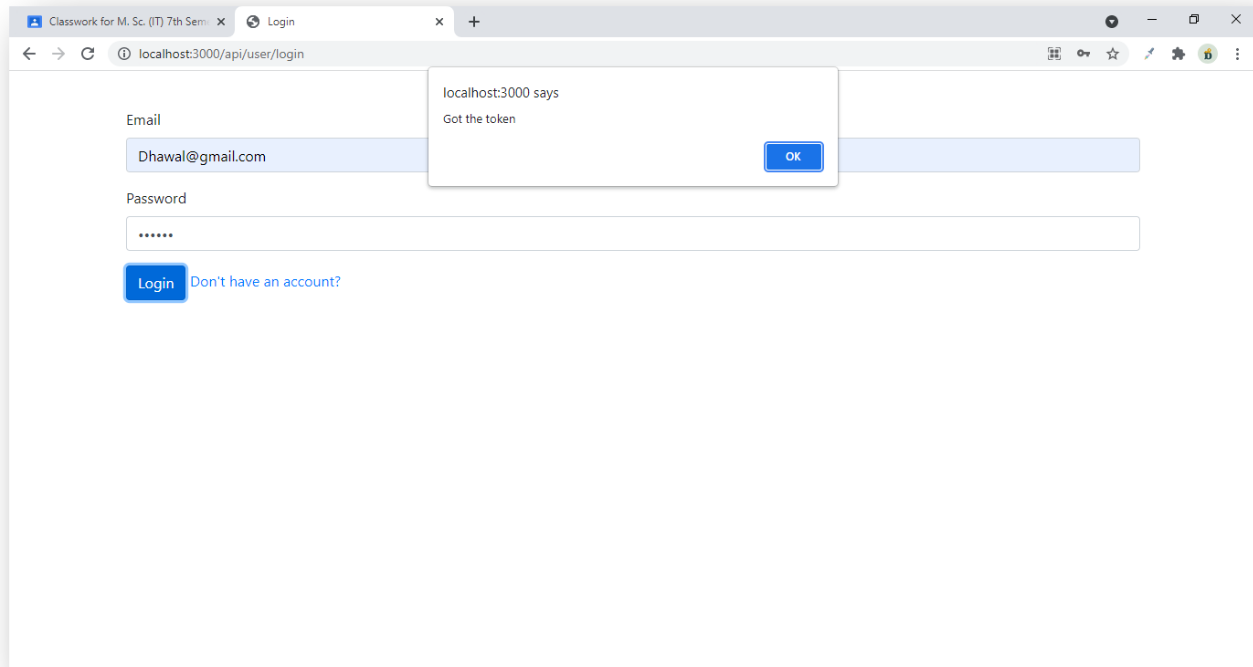
alert(result.message);

if(result.status)
{
  window.location.href = "/api/student/index";

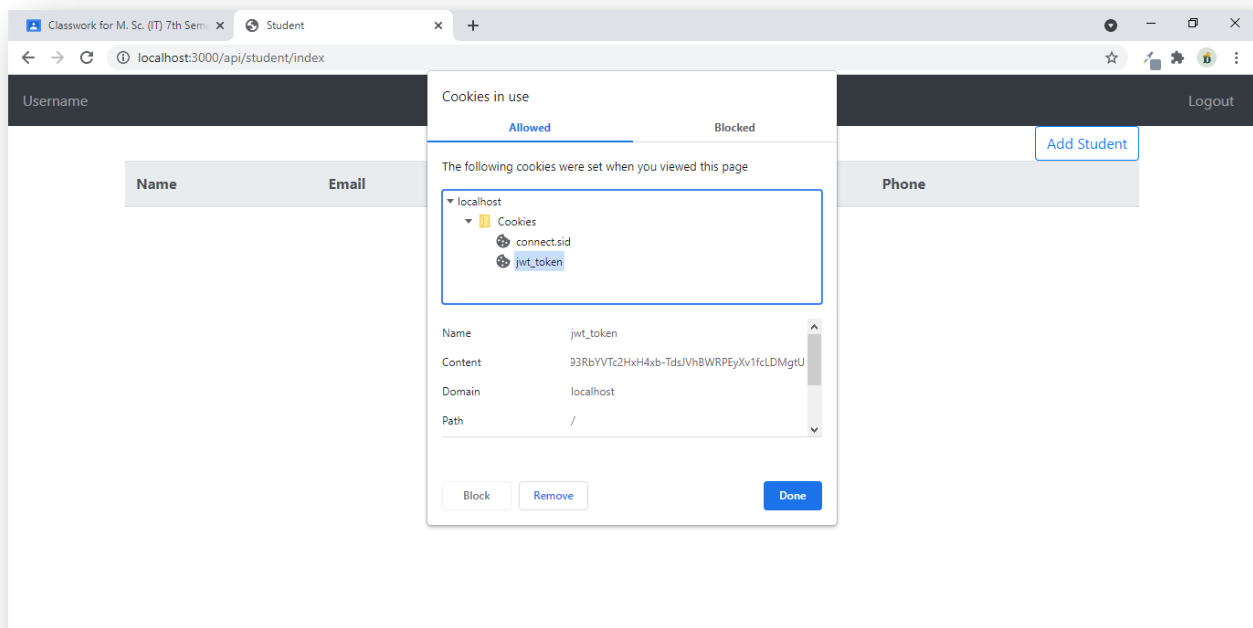
}
}

</script>
</html>
```


Login With JWT



I've stored jwt token in cookie



Adding student

Username Logout

Add Student

Name
Dhawal Parmar

Email
Dhawal@gmail.com

Age
20

Gender ☒ Male ☐ Female

Phone
9979562541

Submit Cancel

localhost:3000 says
Record has been inserted...

OK

Add Student

Name
Dhawal Parmar

Email
Dhawal@gmail.com

Age
20

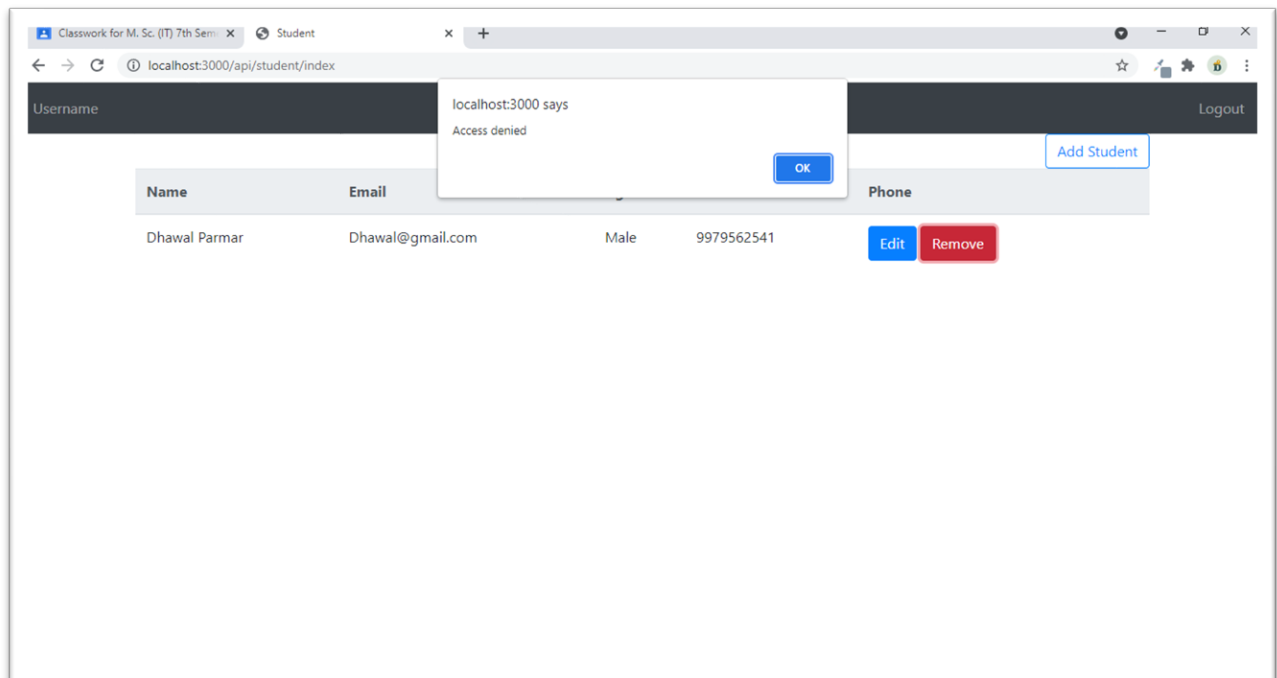
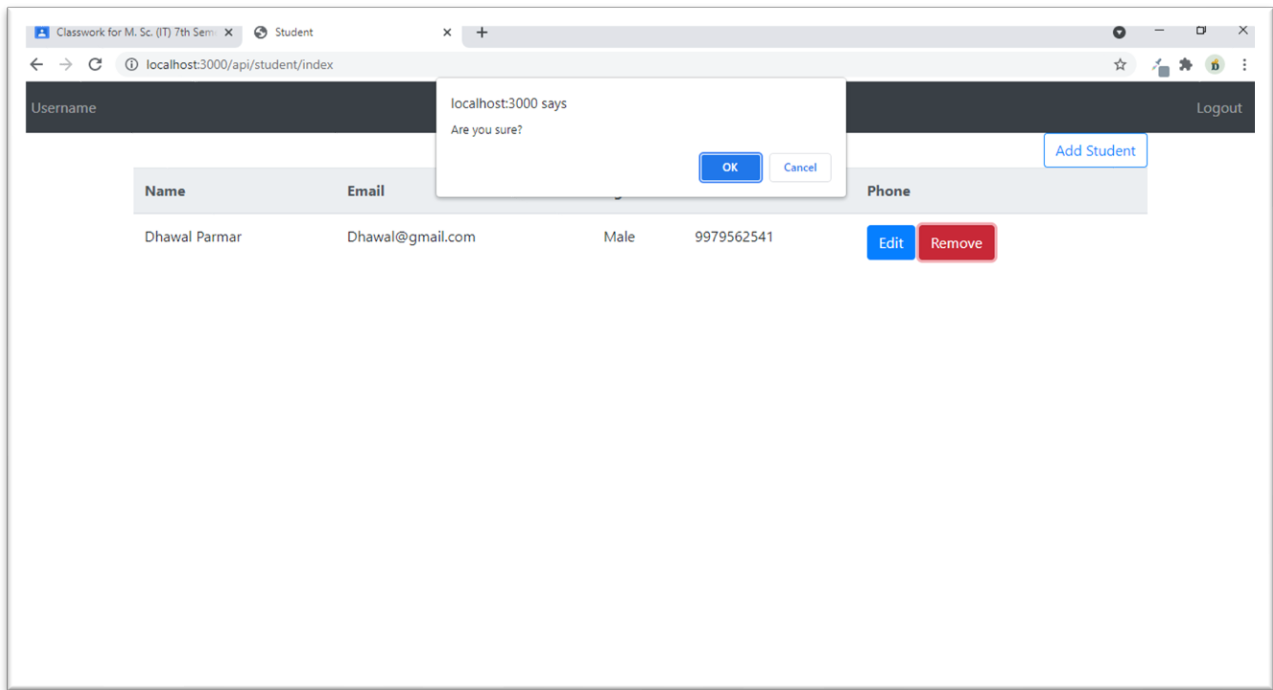
Gender ☒ Male ☐ Female

Phone
9979562541

Submit Cancel

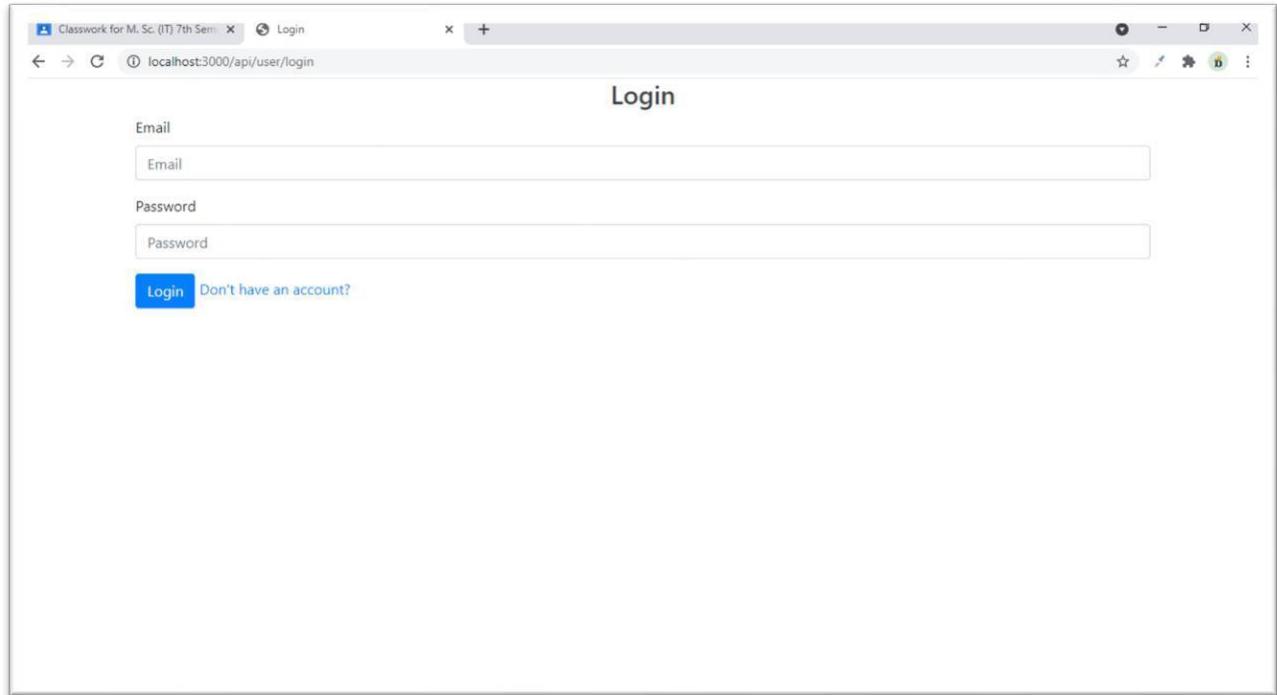
Let me remove JWT_COOKIE and then perform
Action Now let me remove the record

Removing...



Even if you click on edit,add student or remove it will print the message “Access denied”

If you refresh the page it will redirect to login page



Classwork for M. Sc. (IT) 7th Sem x Login x +

localhost:3000/api/user/login

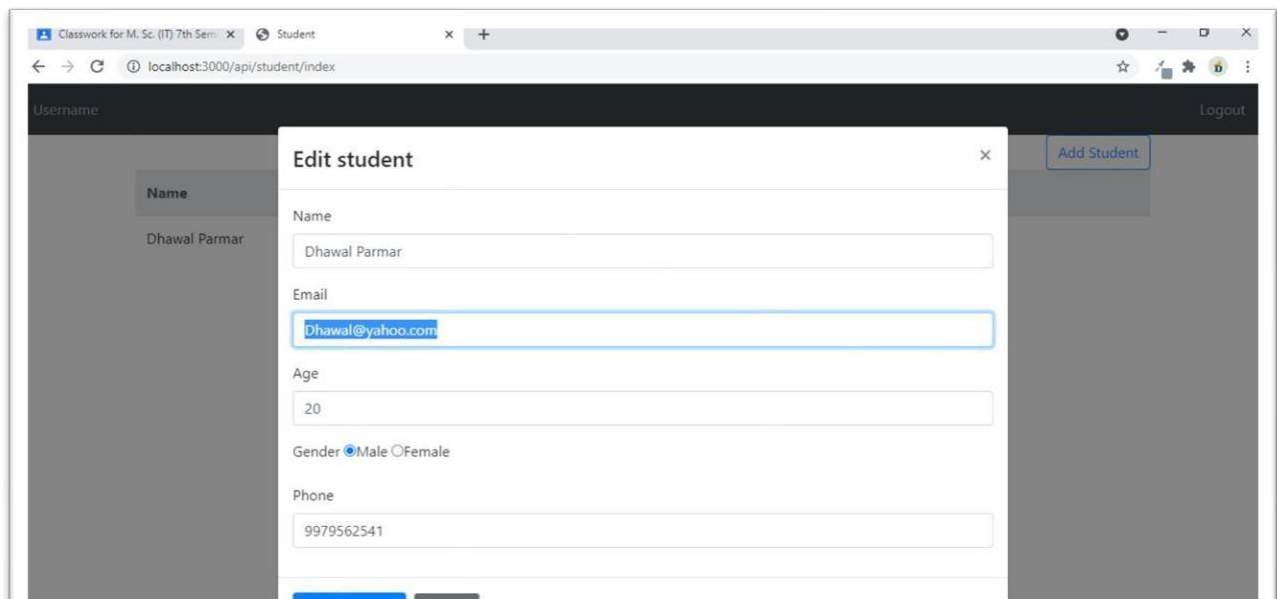
Login

Email

Password

Login Don't have an account?

Again logged In and performing crud operation



Classwork for M. Sc. (IT) 7th Sem x Student x +

localhost:3000/api/student/index

Username Logout

Name
Dhawal Parmar

Edit student

Name

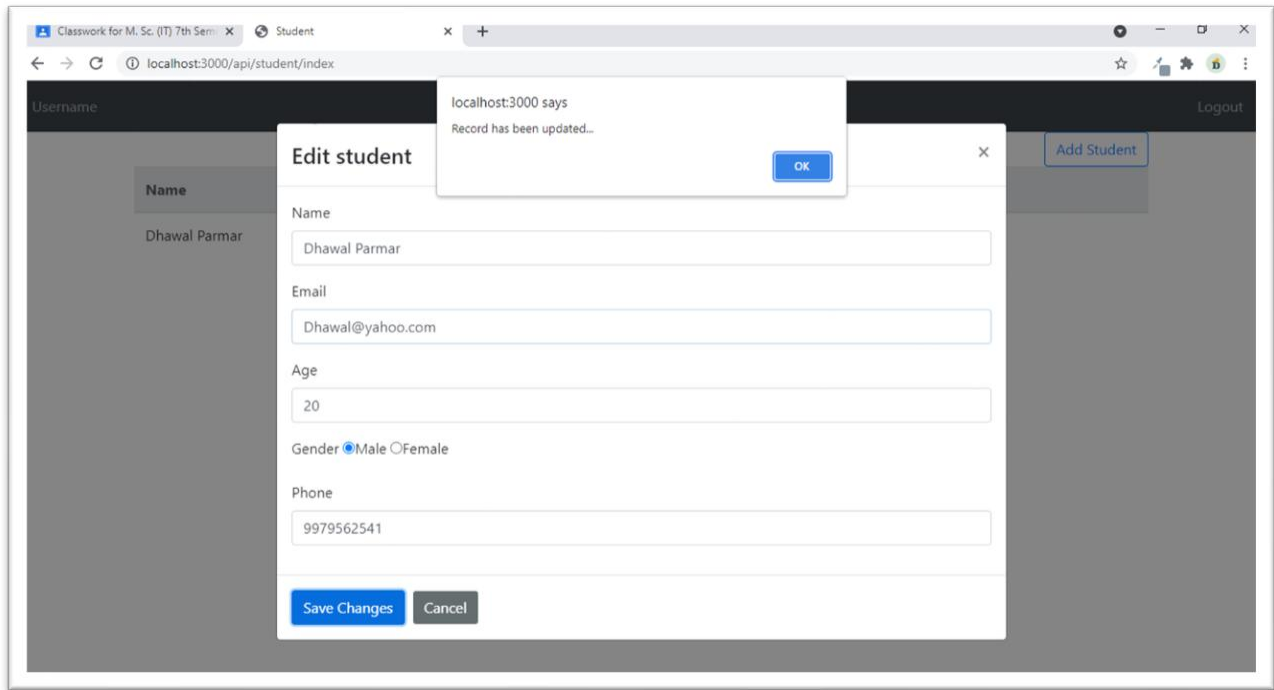
Email

Age

Gender ☒ Male ☐ Female

Phone

Add Student



Removing...

