SmallQuestion16

File System in Node.js: Create a new text file and perform read, write, and append operations.

Create a new file:

App.js:

```
var fs=require('fs');
fs.writeFile("write.txt","console.log('done')",function(err){
    console.log("helloo world");
})
```

Write.txt:

```
console.log('done')
```

By performing the writefile Operation a file with name “write.txt” is created with text “console.log(‘done’)”.

Output:

```
PS C:\Users\Janani\Desktop\AWT-lab\node_practice> node app.js
helloo world
```

Read a read a file:

```
var fs=require('fs');

fs.readFile("write.txt","utf-8",function(err,data){
    console.log(data);
})
```

Output:

```
PS C:\Users\Janani\Desktop\AWT-lab\node_practice> node app.js
console.log('done')
```

Append to a file:

```
var fs=require('fs');
fs.appendFile("write.txt","console.log('hello')",function(err){
    console.log("added successfully")
})
```

Write.txt:

```
console.log('done')console.log('hello')
```

output:

```
PS C:\Users\Janani\Desktop\AWT-lab\node_practice> node app.js
added successfully
```

17.File System in Node js : Perform rename a file and copy a file to another file, delete file operations.

Rename a file:

```
var fs=require('fs');
fs.writeFile("file.txt","console.log('hi')",function(err){
        console.log("file created");
})
fs.rename("file.txt","renamedfile.txt",function(err){
    console.log("file renamed");
})
```

Output:

```
PS C:\Users\Janani\Desktop\AWT-lab\node_practice> node app.js
file renamed
file created
```

Copy a file to another file:

```
var fs=require('fs');
fs.writeFile("first.txt","hello",function(err){
        console.log("first file created");
})
fs.writeFile("second.txt","hello second",function(err){
    console.log("second file created");
})
fs.copyFile("first.txt","second.txt",function(err){
    console.log("copied")
})

First.txt:
Hello

Second.txt:
Hello second
```

Output:

```
PS C:\Users\Janani\Desktop\AWT-lab\node_practice> node app.js
first file created
second file created
copied
```

Second.txt:

hello

Delete a file:

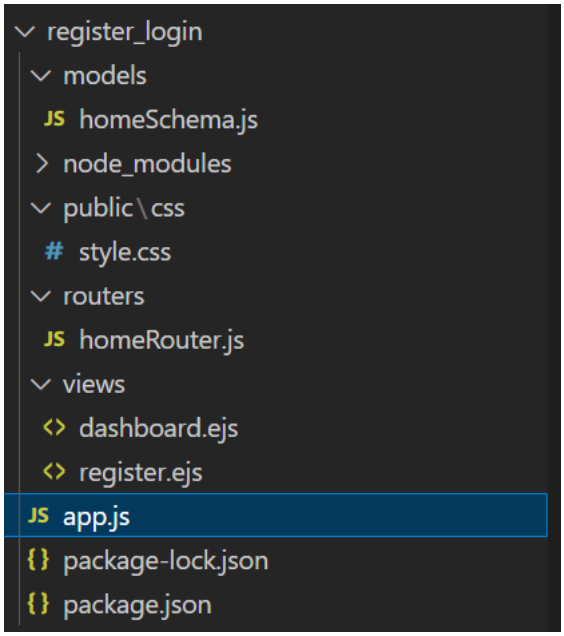
```
var fs=require('fs');
fs.writeFile("file.txt","console.log('hi')",function(err){
        console.log("second file created");
})
fs.unlink("file.txt",function(err){
    console.log("file deleted")
})
```

Output:

```
PS C:\Users\Janani\Desktop\AWT-lab\node_practice> node app.js
file deleted
second file created
```

Big Project:7

Develop a simple authentication(login, register)app(both frontend and backend).



app.js:

```
const express=require("express");
const mongoose=require("mongoose");
const bodyParser=require("body-parser");
const homeRouter=require("./routers/homeRouter");
const port=process.env.port|| 8080;
const app=express();
//db con
mongoose.connect('mongodb://localhost:27017/studentdata',{useNewUrlParser:true
})
const db = mongoose.connection;
db.on("error",()=>{console.log("error in connection");})
db.once('open',()=>{console.log("connected");})
app.set('view engine','ejs')
app.use(express.static('public'))
app.use(bodyParser.urlencoded({ extended: false }))
// parse application/json
app.use(bodyParser.json())

app.use("/",homeRouter);
app.listen(8080)
```

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>
    <link rel="stylesheet" href="/css/style.css" />
  </head>
  <body>
    <h1 style="color: rgb(34, 3, 3)"><%= title %></h1>
    <h1 style="color: rgb(65, 211, 7)"><%= password %></h1>
    <h1 style="color: rgb(255, 0, 200)"><%= email %></h1>

    <div class="main">
      <input type="checkbox" id="chk" aria-hidden="true" />
      <!-- signup -->
      <div class="signup">
        <form action="/register" method="post">
          <label for="chk" aria-hidden="true">Sign up</label>
          <input type="text" name="name" placeholder="User name" />
          <input type="number" name="number" placeholder="PHone number" />
          <input type="email" name="email" placeholder="Email" />
          <input type="password" name="password" placeholder="Password" />
          <input
            type="password"
            name="cpassword"
            placeholder="Confirm Password"
          />
          <button type="submit">Sign up</button>
        </form>
      </div>
      <!-- login -->
      <div class="login">
        <form method="post" action="/login">
          <label for="chk" aria-hidden="true">Login</label>
          <input type="email" name="email" placeholder="Email" required="" />
          <input
            type="password"
            name="password"
            placeholder="Password"
            required=""
          />
          <button>Login</button>
        </form>
      </div>
    </div>
```

```
  </body>
</html>
```

Dashboard.ejs:

```
<%= name %>
```

Style.css:

```
style.css
body{
    margin: 0;
    padding: 0;
    display: flex;
    justify-content: center;
    align-items: center;
    min-height: 100vh;
    font-family: 'Jost', sans-serif;
    background: linear-gradient(to top, #03a0e4,  #ac048af5);
}
h1{
position: absolute;
right: 14%;
top: 5%;
background-color: #fff;
padding: 10px 40px;
}
.main{
    width: 350px;
    height: 600px;
    overflow: hidden;
    border-radius: 10px;
    box-shadow: 5px 20px 50px rgba(0, 0, 0, 0.623);
}
#chk{
    display: none;
}
.signup{
    position: relative;
    width:100%;
    height: 100%;
}
label{
    color: #fff;
    font-size: 2.3em;
    justify-content: center;
    display: flex;
    margin: 60px;
```

```
    font-weight: bold;
    cursor: pointer;
    transition: .5s ease-in-out;
}
input{
    width: 60%;
    height: 20px;
    background: #e0dede;
    justify-content: center;
    display: flex;
    margin: 20px auto;
    padding: 10px;
    border: none;
    outline: none;
    border-radius: 5px;
}
button{
    width: 60%;
    height: 40px;
    margin: 10px auto;
    justify-content: center;
    display: block;
    color: #fff;
    background: #573b8a;
    font-size: 1em;
    font-weight: bold;
    margin-top: 20px;
    outline: none;
    border: none;
    border-radius: 5px;
    transition: .2s ease-in;
    cursor: pointer;
}
button:hover{
    background: #6d44b8;
}
.login{
    height: 460px;
    background: #eee;
    border-radius: 60% / 10%;
    transform: translateY(-180px);
    transition: .8s ease-in-out;
}
.login label{
    color: #573b8a;
    transform: scale(.6);
}
```

```
#chk:checked ~ .login{
    transform: translateY(-500px);
}
#chk:checked ~ .login label{
    transform: scale(1);
}
#chk:checked ~ .signup label{
    transform: scale(.6);
}
```

homeRouter.js:

```
const express = require('express');
const Router = express.Router();
const homeSchema = require('../models/homeSchema');

Router.get('/', (req, res) => {
    res.render('register', { title: 'Fill Form', password: '', email: '' });
});

Router.post('/register', async (req, res) => {
    try {
        const { name, number, email, password, cpassword } = req.body;

        if (password === cpassword) {
            // Check if the email already exists
            const existingUser = await homeSchema.findOne({ email });

            if (existingUser) {
                return res.render('register', {
                    title: '',
                    password: '',
                    email: 'Email is already in use. Please choose a different
one.',
                });
            }

            const userData = new homeSchema({
                name,
                number,
                email,
                password,
            });

            await userData.save();
            res.render('register', { title: 'Done', password: '', email: ''
});
```

```
        } else {
            res.render('register', { title: '', password: 'Password not
Matching', email: '' });
        }
    } catch (error) {
        console.error(error);
        res.render('register', { title: 'Error in Code', password: '', email:
'' });
    }
});

// sign in

Router.post('/login', async (req, res) => {
    const { email, password } = req.body;

    try {
        const result = await homeSchema.findOne({ email });

        if (result && result.password === password) {
            res.render('dashboard', { name: result.name });
        } else {
            console.log('Invalid email or password');
            res.render('login', { title: '', password: 'Invalid email or
password', email: '' });
        }
    } catch (error) {
        console.error(error);
        res.render('login', { title: 'Error in Code', password: '', email: ''
});
    }
});

module.exports = Router;
```

homeSchema:

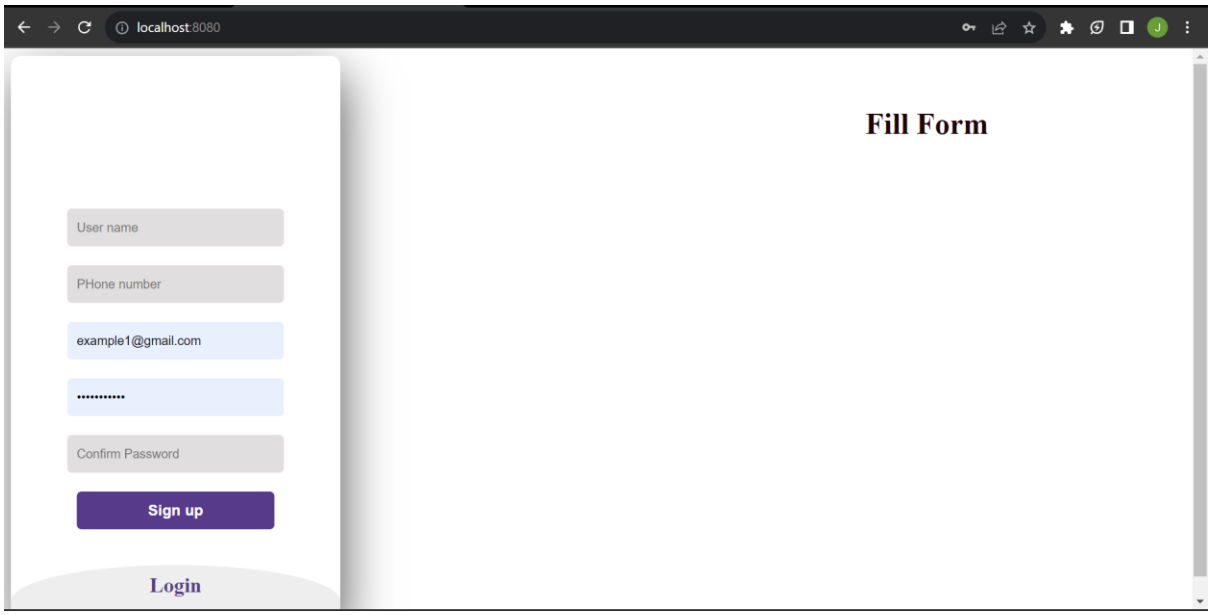
```
const mongoose = require('mongoose');

const schema = mongoose.Schema;
const userSchema = new schema({
    name:{
        type:String,
        required:true
    },
    number:{
```

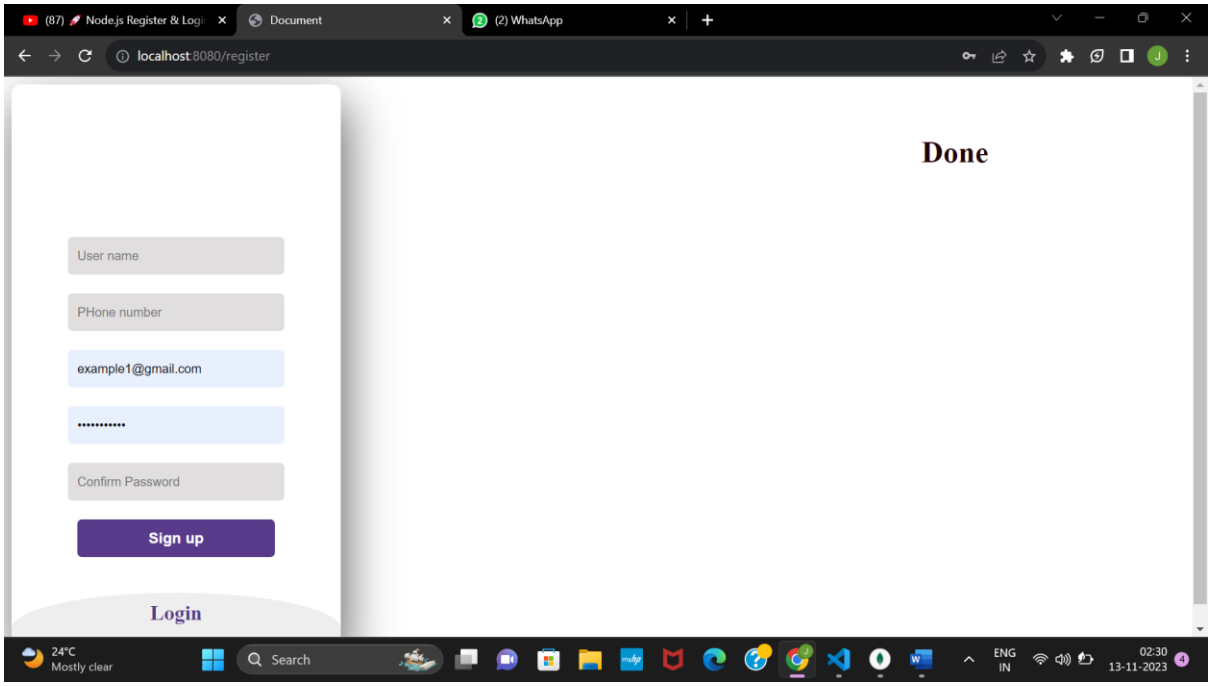
```
        type:Number,
        required:true
    },
    email:{
        type:String,
        unique:true,
        required:true
    },
    password:{
        type:String,
        required:true
    },


})

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

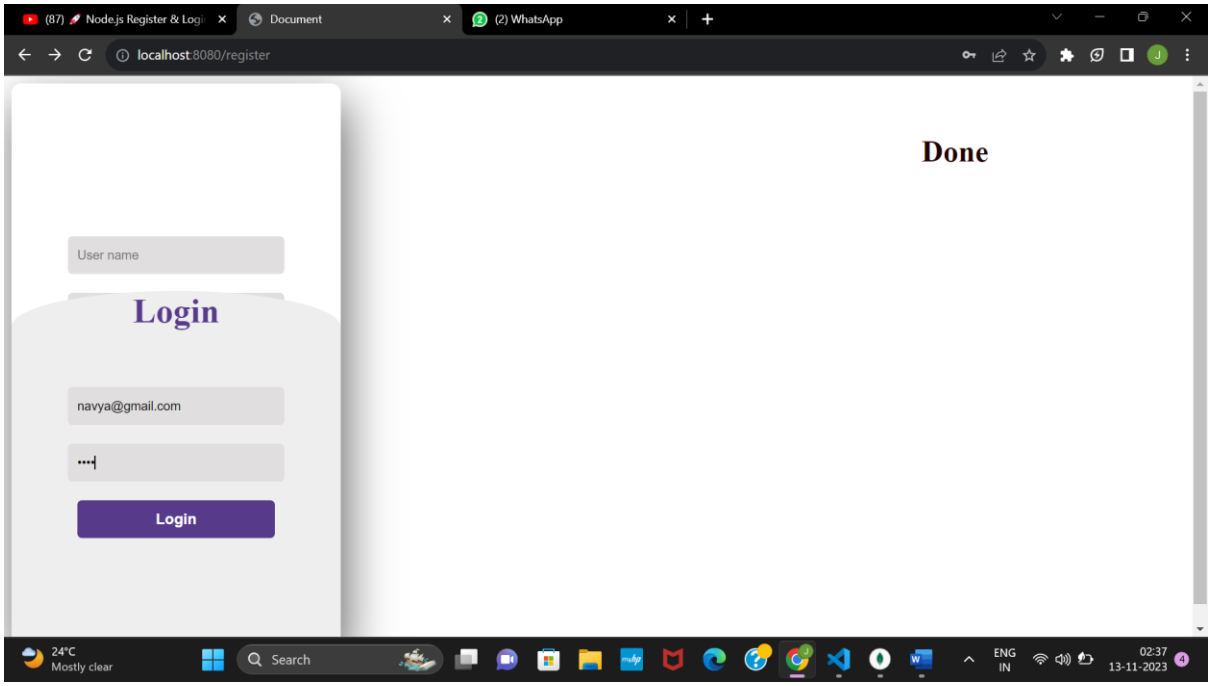
Run by using the command:

npm run dev

Output:




(87) Node.js Register & Logi
Document
(2) WhatsApp
localhost:8080/register
User name
PHone number
example1@gmail.com
Confirm Password
Sign up
Login
Done
24°C
Mostly clear
Search
ENG
IN
02:30
13-11-2023


(87) Node.js Register & Logi
Document
(2) WhatsApp
localhost:8080/register
User name
Login
navya@gmail.com
Login
Done
24°C
Mostly clear
Search
ENG
IN
02:37
13-11-2023


(87) Node.js Register & Logi
localhost:8080/login
(2) WhatsApp
localhost:8080/login

navya


24°C
Mostly clear
Search
ENG
IN
02:37
13-11-2023