Login

Tuesday, June 13, 2023 9:53 PM

Step 1: create the html

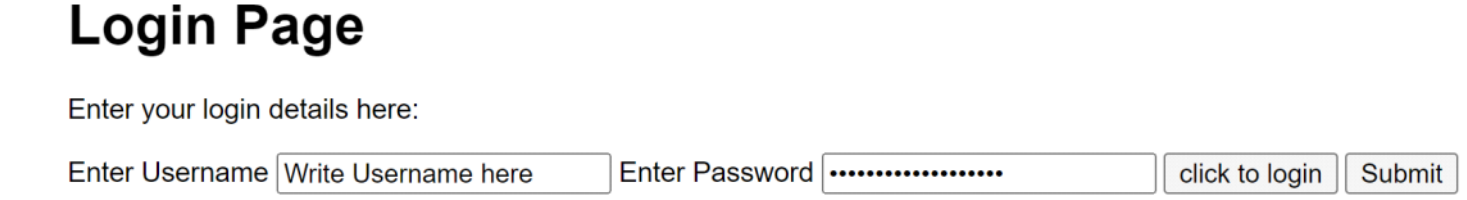
```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>login Page</title>
    <link rel="stylesheet" href="/stylesheets/style.css">
</head>

<body>
    <h1>Login Page</h1>
    <p>Enter your login details here:</p>
    <form action="myPage.html" method="get">
        <label for="user">Enter Username</label>
        <input type="text" id="user" value="Write Username here"/>

        <label for="pass">Enter Password</label>
        <input type="password" id="pass" value="Write Password here"

        <button type="button">click to login</button>
    </form>
</body>
</html>
```

Login html created.



Step 2: Create the JSON of the input details and send them to the server using AJAX

In myJavascript.js

```
function myloginFunc() {
    let loginData = {
        username: document.getElementById('user').value,
        password: document.getElementById('pass').value
    };

    let myxhttp = new XMLHttpRequest();

    myxhttp.open('POST', '/login');
    myxhttp.setRequestHeader('content-type', 'application/json');
    myxhttp.send(JSON.stringify(loginData));
}
```

Step 3:
Lets set the server side as well so that it can handle the post to /login path

In index.js

```
router.post('/login', function(req, res, next){
  console.log(JSON.stringify(req.body));
  res.end();
});

module.exports = router;
```

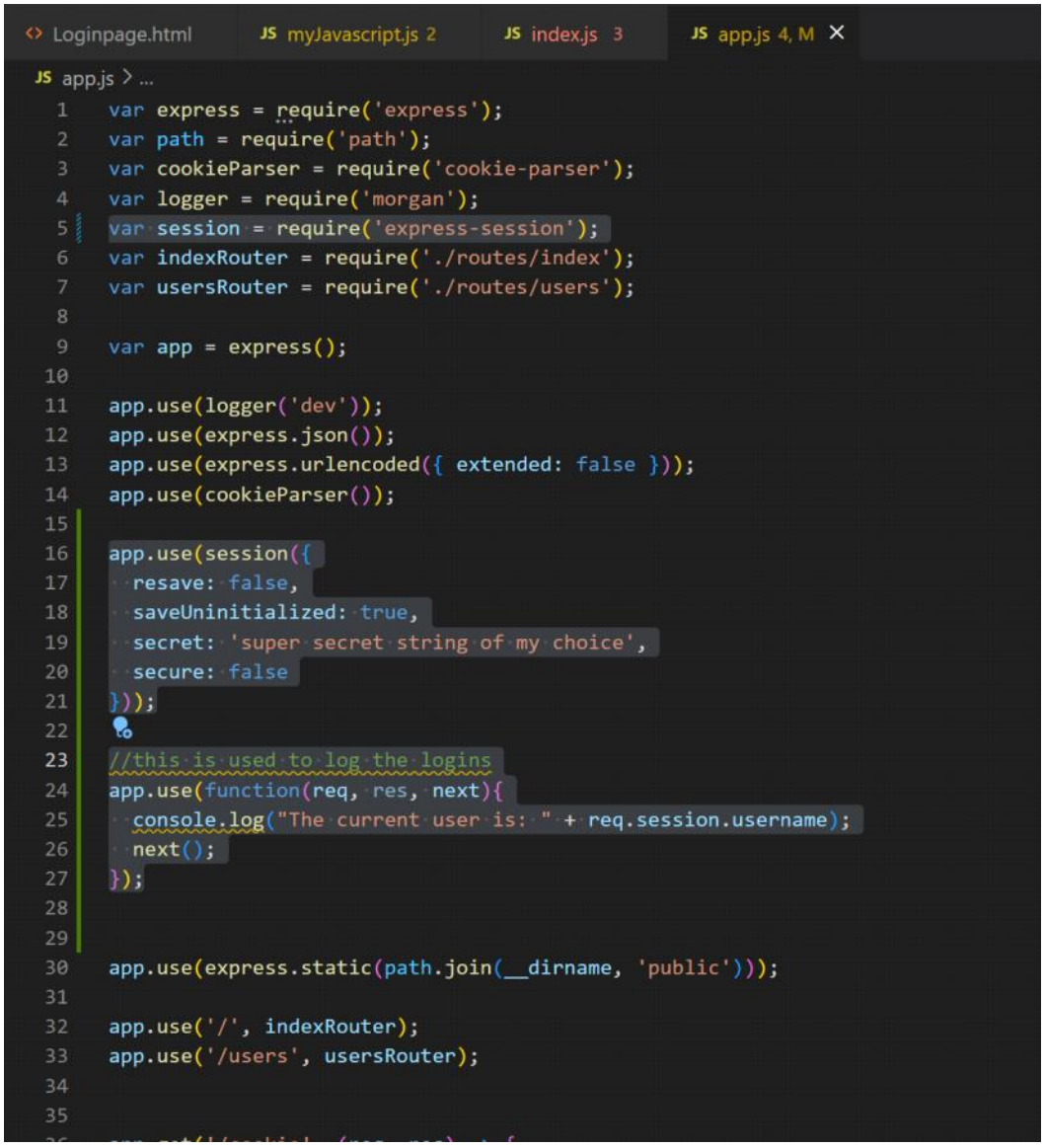
Pushing button sends response to the server through AJAX client:

```
[nodemon] starting `node ./bin/www`
{"username":"Turjoasd","password":"asdasd"}
POST /login 200 845.759 ms - -
{"username":"Turjoasd","password":"asdasd"}
POST /login 200 3.112 ms - -
^C
```

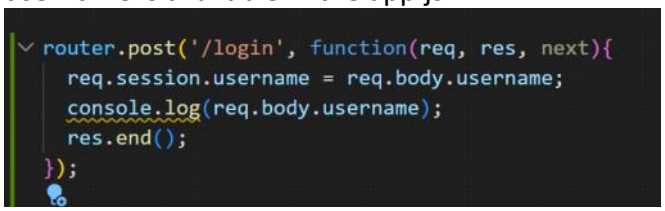
Step 4: Now we need to store them so let's create a server session

```
root ➜/workspaces/23S1_WDC_prac7-Turjo009 (main) $ npm install --save express-session
(                ) :. idealTree:23S1_WDC_prac7-Turjo009: sill idealTree buildDeps
```

In the app.js setup the server session as well. NOTE: keep the name of the variable as "sessions"



A slight modification in the server index.js is made so that the username is available in the app.js:



DONE:

NOW LETS CHECK USERS FROM STORED DATA

In index.js:
Create an object user, and set and if and else statement to

verify if the input matches the data of the user.

```
39
40  let users = {
41    Ami: {password: 'mypass1'},
42    Gandu: {password: 'mypass2'}
43  };
44
45  router.post('/login', function(req, res, next){
46
47    if(req.body.username in users){
48      if(req.body.password === users[req.body.username].password){
49        req.session.username = req.body.username;
50        console.log("you are loggin in, Congrats");
51        res.end();
52      }
53      else{
54        res.sendStatus(401);
55        console.log("Wrong password, sad");
56      }
57
58    }
59    else{
60      res.sendStatus(401);
61      console.log("User Not Found, sad");
62    }
63  });
64
65  module.exports = router;
```

DONE

Suppose you want to make a post only when a user is logged in:

```
35  router.post('/newpost', function(req,res,next){
36
37    if (!('username' in req.session)){
38      res.sendStatus(403);
39      return;
40    }
41
42    // Receive data from the client; we'll look more at post requests later this semester
43    let post = req.body;
44    /*
ROBLEMS 16  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS 1
```

So if a username attribute is created for the session object then the server will allow to post new content.