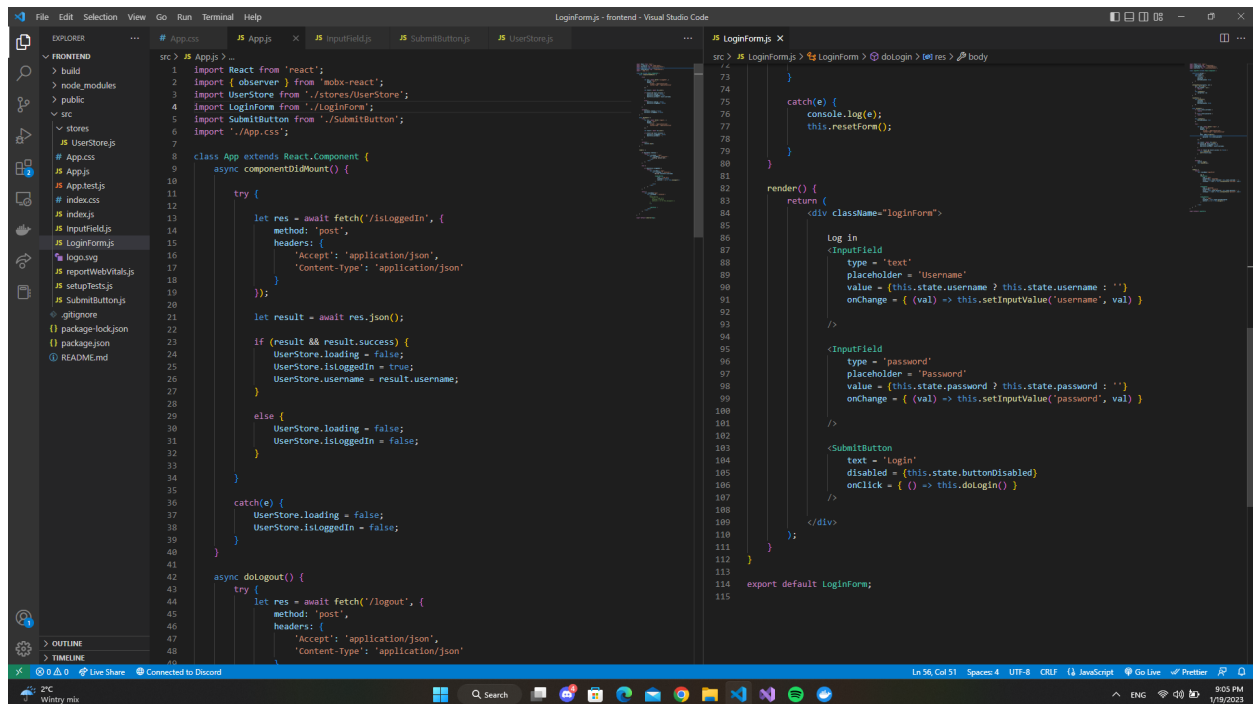


Anar Nuriyev
ID 39472
Security Of Web Applications L1
Lab 11

Project name: Web Login Panel

Used programming languages: React + Express + Node.js + MySQL

The Frontend part is already rendered in React before dockerizing my project so I can only show screenshots:



```
src > # App.js > ...
1 import React from 'react';
2 import { observer } from 'mobx-react';
3 import UserStore from './stores/userStore';
4 import LoginForm from './LoginForm';
5 import SubmitButton from './SubmitButton';
6 import './App.css';
7
8 class App extends React.Component {
9   async componentDidMount() {
10     try {
11       let res = await fetch('/isLoggedIn', {
12         method: 'post',
13         headers: {
14           'Accept': 'application/json',
15           'Content-Type': 'application/json'
16         }
17       });
18       let result = await res.json();
19       if (result && result.success) {
20         UserStore.loading = false;
21         UserStore.isLoggedIn = true;
22         UserStore.username = result.username;
23       }
24       else {
25         UserStore.loading = false;
26         UserStore.isLoggedIn = false;
27       }
28     } catch (e) {
29       UserStore.loading = false;
30       UserStore.isLoggedIn = false;
31     }
32   }
33
34   async doLogout() {
35     try {
36       let res = await fetch('/logout', {
37         method: 'post',
38         headers: {
39           'Accept': 'application/json',
40           'Content-Type': 'application/json'
41         }
42       });
43     } catch (e) {
44       console.log(e);
45     }
46   }
47
48   render() {
49     return (
50       <div className="loginForm">
51         <div>
52           <input type="text" value={this.state.username} onChange={this.handleChange('username')} />
53           <input type="password" value={this.state.password} onChange={this.handleChange('password')} />
54           <SubmitButton text="login" disabled={this.state.buttonDisabled} onClick={this.handleClick} />
55         </div>
56       </div>
57     );
58   }
59 }
60
61 export default App;
```

```
src > # App.js > ...
1 // padding: 12px;
2 font-size: 14px;
3 background: #f8f8f8;
4 }
5
6 .submitButton {
7   padding-top: 16px;
8 }
9
10 .btn {
11   width: 100%;
12   min-width: 200px;
13   color: #555555;
14   padding: 12px;
15   font-size: 14px;
16   font-weight: bold;
17   border: solid 2px #1189de;
18   border-radius: 4px;
19   background: #fff;
20   cursor: pointer;
21 }
22
23 export default new UserStore();
```

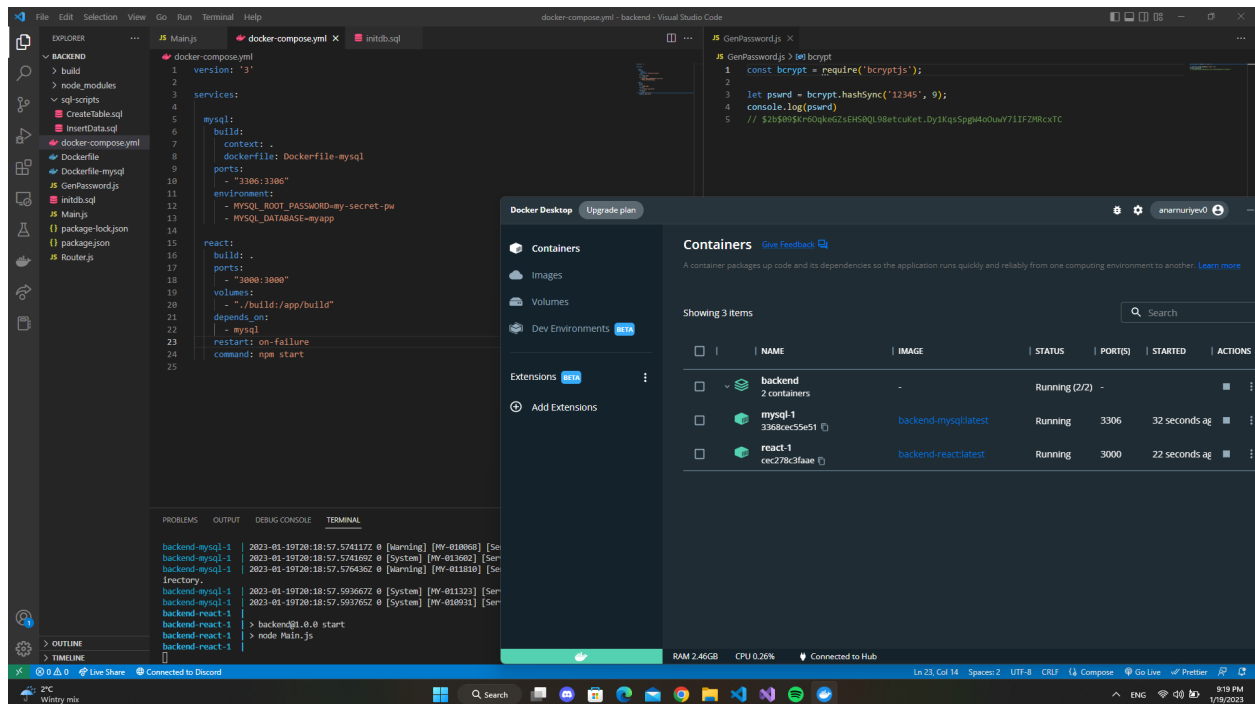
```
src > # SubmitButton.js > SubmitButton
1 import React from 'react';
2
3 class SubmitButton extends React.Component {
4
5   render() {
6     return (
7       <div className="submitButton">
8         <button
9           className = 'btn'
10           disabled = {this.props.disabled}
11           onClick = { () => this.props.onClick() }
12         >
13           {this.props.text}
14         </button>
15       </div>
16     );
17   }
18 }
19
20 export default SubmitButton;
```

Here is the backend part:

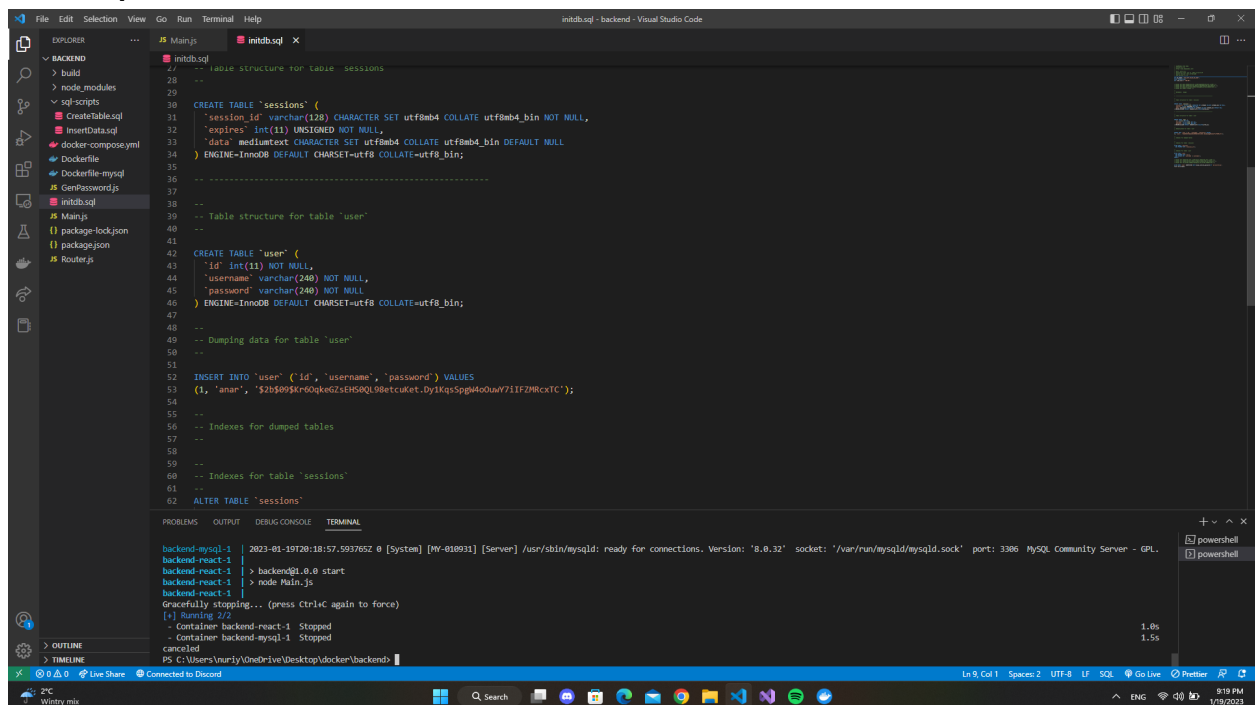
```
src > # Main.js > @ database
1 const express = require('express');
2 const app = express();
3 const path = require('path');
4 const mysql = require('mysql');
5 const session = require('express-session');
6 const MySQLStore = require('express-mysql-session')(session);
7 const Router = require('./Router');
8
9 app.use(express.static(path.join(__dirname, 'build')));
10 app.use(express.json());
11
12 // Database
13 const db = mysql.createConnection({
14   host: 'mysql',
15   user: 'root',
16   password: 'my-secret-pw',
17   database: 'myapp'
18 });
19
20 db.connect(function(err) {
21   if (err) {
22     console.log('DB error');
23     throw err;
24     return false;
25   }
26 });
27
28 const sessionStore = new MySQLStore({
29   expiration: (1825 * 86400 * 1000), // 5 years session
30   endConnectionOnClose: false
31 }, db);
32
33 app.use(session({
34   key: 'sessionid',
35   secret: 'sessionssekret',
36   store: sessionStore,
37   resave: false,
38   saveUninitialized: false,
39   cookie: {
40     maxAge: (1825 * 86400 * 1000),
41     httpOnly: false
42   }
43 }));
44
45 new Router(app, db);
46
47 app.get('/', function(req, res) {
48   res.sendFile(path.join(__dirname, 'build', 'index.html'));
```

```
src > # Router.js > @ bcrypt
1 const bcrypt = require('bcryptjs');
2
3 class Router {
4
5   constructor(app, db) {
6     this.login(app, db);
7     this.logout(app, db);
8     this.isLoggedIn(app, db);
9   }
10
11   login(app, db) {
12
13     app.post('/login', (req, res) => {
14
15       let username = req.body.username;
16       let password = req.body.password;
17
18       username = username.toLowerCase();
19
20       if (username.length > 12 || password.length > 12) {
21         res.json({
22           success: false,
23           msg: 'An errors occurred, please try again'
24         });
25         return;
26       }
27
28       let cols = [username];
29       db.query('SELECT * FROM user WHERE username = ? LIMIT 1', cols, (err, data,
30
31         if (err) {
32           res.json({
33             success: false,
34             msg: 'An errors occurred, please try again'
35           });
36           return;
37         }
38
39         // found 1 user with this username
40         if (data && data.length > 0) {
41           bcrypt.compare(password, data[0].password, (bcryptErr, verified) =>
42
43             if (verified) {
44               req.session.userId = data[0].id;
45
46               res.json({
47                 success: true,
```

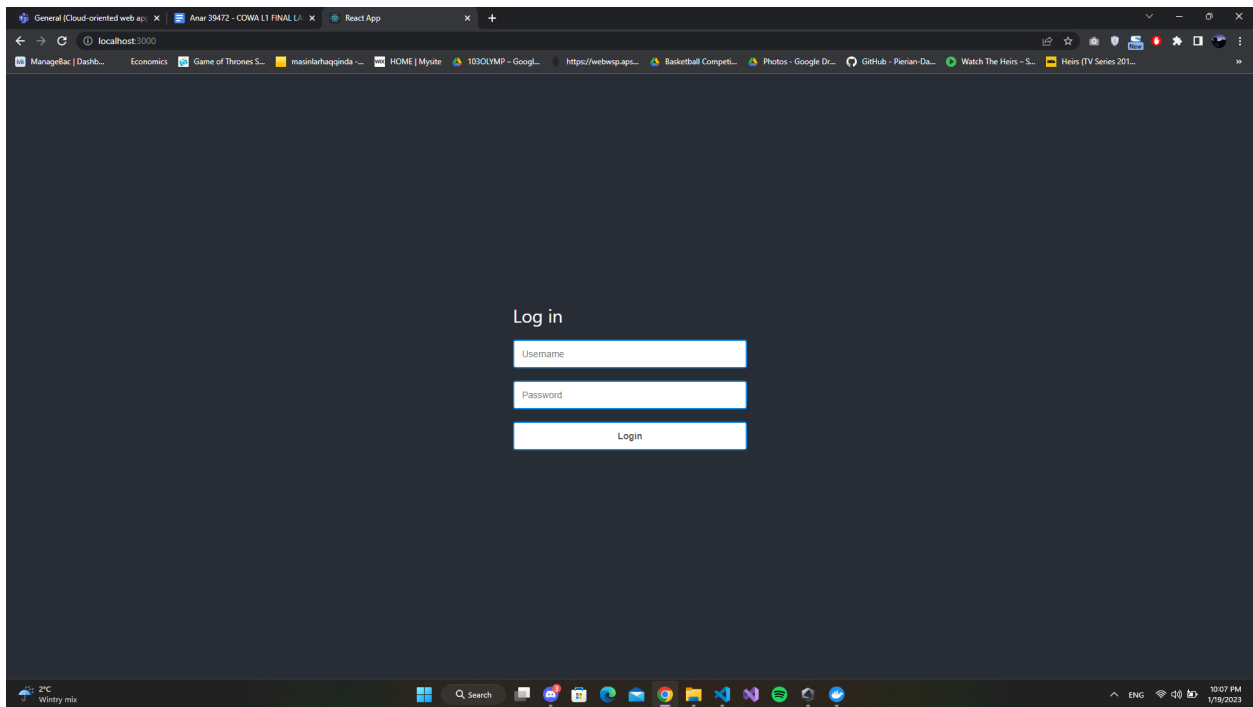
Code: docker-compose up



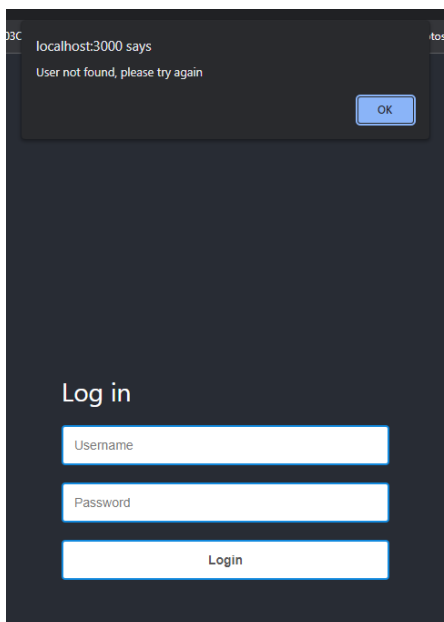
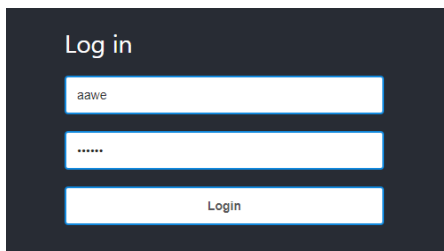
For the SQL part, I manually inserted my SQL tables and contents into my docker MYSQL. Thanks to the phpmyadmin! Because I exported my tables and data as an initdb.sql file.



FRONTEND:



If you try to log in with the wrong username or password it will automatically reset all the inputboxes.

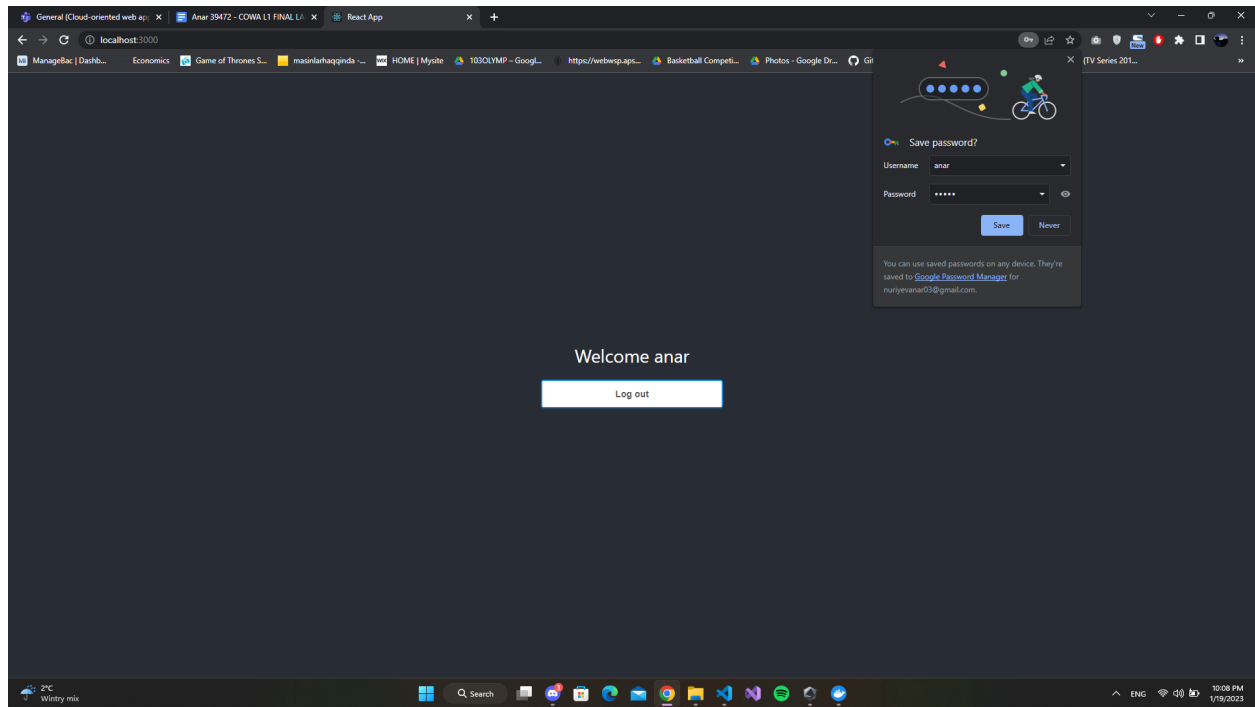


Username: anar

Password: 12345

(if you check the user tables you can see that all passwords are encrypted)

Firstly I used bcrypt. But then I realised that bcrypt only works on Windows environment. Because some library parts are written in C++. Then I updated my bcrypt library to the bcryptJS.



SESSION is live!!! So if you restart the MySQL (container or whole container) it always stays on this page if you already signed in. All the sessions will be saved in the client's chrome data (cookies) and the server's MySQL data. So data is secured in two ways.