Assignment 2C

Angular Files:

home.component.ts:

```
import { Component } from '@angular/core';
import { CookieService } from 'ngx-cookie-service';
import { Router } from '@angular/router';

@Component({
  selector: 'app-home',
  templateUrl: './home.component.html',
  styleUrls: ['./home.component.css'],
})
export class HomeComponent {
  message: string;

  constructor(private cookieService: CookieService, private router: Router) {
    let user: string = localStorage.getItem('user') ?? '{}';
    const userDetails = JSON.parse(user);
    this.message = userDetails.username
      ? 'Hello ' + userDetails.username
      : 'LOGIN';
    this.validate();
  }

  validate() {
    // verify
    const jwt = this.cookieService.get('jwt');
    console.log(jwt);
    const myHeaders = new Headers({
      'Content-Type': 'application/json',
      Cookie: `jwt=${jwt}`,
    });

    var requestOptions: Object = {
      method: 'GET',
      headers: myHeaders,
      redirect: 'follow',
      credentials: 'include',
    };

    fetch('http://localhost:2324', requestOptions)
      .then((response) => {
        if (!response.ok) {
          throw new Error(String(response.status));
        }
        return response.json();
      })
      .then((result) => {
        console.log(result);
        if (result.message == 'Token is not valid') {
          // redirect to login
          localStorage.removeItem('user');
          this.router.navigate(['/login']);
        } else if (result.message == 'Token is not provided') {
          // redirect to login
          localStorage.removeItem('user');
          this.router.navigate(['/login']);
        }
        if (this.message == 'LOGIN') {
          this.router.navigate(['/login']);
        }
      })
      .catch((error) => console.log('error', error));
  }

  logout() {
    localStorage.removeItem('user');
    this.router.navigate(['/login']);
  }
}
```

login.component.ts:

```
import { Component } from '@angular/core';
import { Router } from '@angular/router';

@Component({
  selector: 'app-login',
  templateUrl: './login.component.html',
  styleUrls: ['./login.component.css'],
})
export class LoginComponent {
  usernameError: string = '';
  passwordError: string = '';

  constructor(private router: Router) {}

  async getData(username: any, password: any) {
    this.usernameError = username ? '' : 'Username is required';
    this.passwordError = password ? '' : 'Password is required';

    if (username && password) {
      var myHeaders = new Headers();
      myHeaders.append('Content-Type', 'application/json');

      var raw = JSON.stringify({
        username: username,
        password: password,
      });

      var requestOptions: Object = {
        method: 'POST',
        headers: myHeaders,
        body: raw,
        redirect: 'follow',
      };

      fetch('http://localhost:2324/login', requestOptions)
        .then((response) => response.json())
        .then((result) => {
          if (result.token) {
            document.cookie = 'jwt=' + result.token;
            localStorage.setItem('user', JSON.stringify(result.user));
            this.router.navigate(['']);
          } else {
            this.usernameError =
              result.message == 'Username Not Found' ? result.message : '';
            this.passwordError =
              result.message == 'Incorrect Password' ? result.message : '';
          }
        })
        .catch((error) => console.log('error', error));
    }
  }
}
```

signup.component.ts:

```
import { Component } from '@angular/core';
import { Router } from '@angular/router';

@Component({
  selector: 'app-signup',
  templateUrl: './signup.component.html',
  styleUrls: ['./signup.component.css'],
})
export class SignupComponent {
  usernameError: string = '';
  passwordError: string = '';

  constructor(private router: Router) {}

  async getData(username: any, password: any) {
    this.usernameError = username ? '' : 'Username is required';
    this.passwordError = password ? '' : 'Password is required';

    if (username && password) {
      var myHeaders = new Headers();
      myHeaders.append('Content-Type', 'application/json');
      var raw = JSON.stringify({
        username: username,
        password: password,
      });

      var requestOptions: Object = {
        method: 'POST',
        headers: myHeaders,
        body: raw,
        redirect: 'follow',
      };

      fetch('http://localhost:2324/signup', requestOptions)
        .then((response) => {
          if (response.ok) {
            return response.json();
          } else {
            this.usernameError = 'Username Already Exists';
            throw Error('Not OK');
          }
        })
        .then((result) => {
          if (result.token) {
            document.cookie = 'jwt=' + result.token;
            localStorage.setItem('user', JSON.stringify(result.user));
            this.router.navigate(['']);
          }
        })
        .catch((error) => console.log('error', error));
    }
  }
}
```

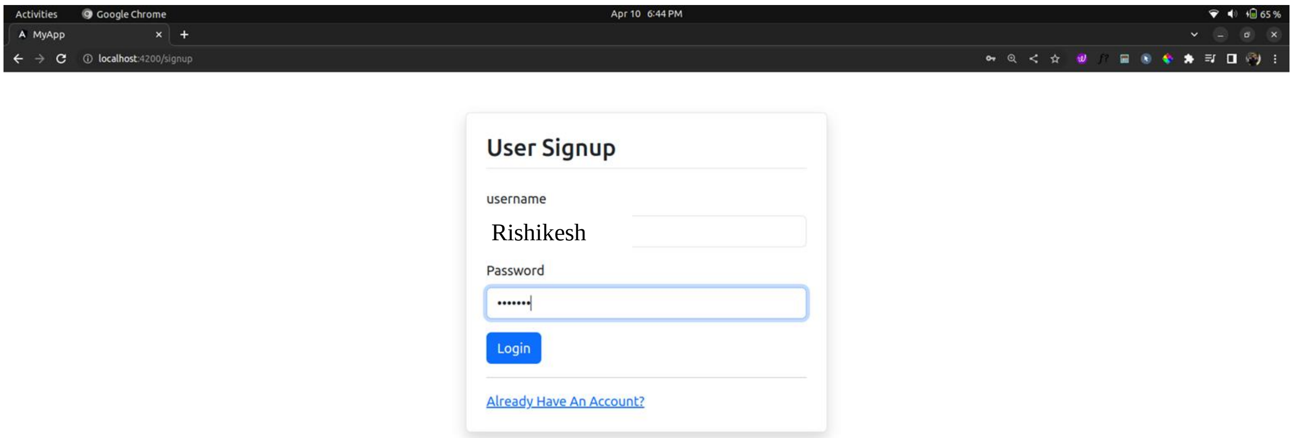
Server Running Database:

index.js:

```
import express from "express"
import User from "./User.js"
import cookieParser from "cookie-parser";
import jwt from "jsonwebtoken";
import connection from "./mongodb.js";
import cors from 'cors';

import dotenv from "dotenv"
dotenv.config();

const app = express();

//middleware
app.use(express.json());
app.use(cookieParser());
app.use(function (req, res, next) {
   // Website you wish to allow to connect
   res.setHeader('Access-Control-Allow-Origin', '*');
   // Request methods you wish to allow
   res.setHeader('Access-Control-Allow-Methods', 'GET, POST, OPTIONS, PUT, PATCH,
DELETE');
   // Request headers you wish to allow
   res.setHeader('Access-Control-Allow-Headers', 'X-Requested-With,content-type');
   // Set to true if you need the website to include cookies in the requests sent
   // to the API (e.g. in case you use sessions)
   res.setHeader('Access-Control-Allow-Credentials', true);
   // Pass to next layer of middleware
   next();
});

app.use(cors({
   origin: 'http://localhost:4200',
   credentials: true
}));


const authMiddleware = (req, res, next) => {
   const jwToken = req.cookies.jwt
   if (jwToken) {
      jwt.verify(jwToken, "This is My Secret", (error, decode) => {
         if (error) {
            res.json({ message: "Token is not valid" })
         } else {
            req._id = decode._id;
            next()
         }
      })
   } else {
      res.json({ message: "Token is not provided" })
   }
}

// functions
async function createToken(_id, res) {
   const token = await jwt.sign({ _id }, "This is My Secret", {
      expiresIn: "2d"
   })
   res.cookie("jwt", token, {
      maxAge: 1000 * 60 * 60 * 24 * 2,
      httpOnly: true
   });
   return token;
}

//routes
app.post("/login", async (req, res) => {
    try {
        const user = await User.login(req.body);
        if (user == 1) {
            res.clearCookie("jwt");
            return res.send({ message: "Username Not Found" })

        } else if (user == 2) {
            res.clearCookie("jwt");
            return res.send({ message: "Incorrect Password" })
        }

        const token = await createToken(user._id, res)
        return res.send({ user, token });
    } catch (error) {
        res.status(404).send({ error })
    }
})

app.post("/signup", async (req, res) => {
    const { username, password } = req.body;
    if (username && password) {
        try {
            const user = await User.create({ username, password });
            const token = await createToken(user._id, res)
            return res.send({ user, token });
        } catch (error) {
            return res.status(404).send({ error })
        }
    } else {
        return res.status(404).json({ message: "`username` and `password` are required" })
    }
})

app.get("/", authMiddleware, async (req, res) => {
    try {
        const user = await User.findOne({ _id: req._id })
        res.status(200).json({ username: user.username })
    } catch (error) {
        res.status(494).json({ error })
    }
})

//server connection
connection.then(() => {
    app.listen(2324, () => {
        console.log("server started: http://localhost:2324")
    })
}).catch(err => {
    console.log(err)
})
```

Output-

Web Output:



User Signup

username

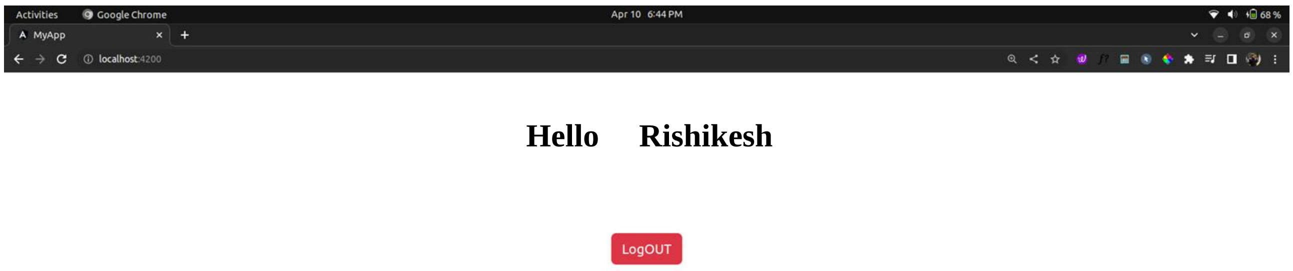
Rishikesh

Password

•••••••

Login

Already Have An Account?



Hello Rishikesh

LogOUT