CS-546 Lab 9

A Simple User Login System

Using the concepts learned on authentication, middleware, etc, you will implement a basic user system with only 3 routes.

This lab may require a good deal of reading, but relatively little code to be written out.

You have the option of using Passport for this lab. If you choose to use the Passport package to implement your login system, you must first read up on Passport (Links to an external site.)Links to an external site. and what it accomplishes, then read up on the Passport Local Strategy (Links to an external site.)Links to an external site. in order to complete this lab; you can use other strategies as well, should your research take you in another direction. Passport is a very simple, easy to drop in middleware that allows you to quickly add application authentication. A strategy is a way of checking whether or not a request is authenticated.

You also have the option of **not** using Passport, and implementing your own simple user login system following the description from the lecture slides and lecture code example.

Your Routes

GET /

The root route of the application will do one of two things:

1. If the user is authenticated, it will redirect to `/private`
2. If the user is not authenticated, it will render a view with a login screen for a username and password. If there are any errors from a previous login, it will display the errors as well.

An authenticated user should not ever see the login screen.

POST /login

This route is simple: posting to this route will attempt to log a user in with the credentials they provide in the login form.

If you are using passport, you will check the provided username and password by telling passport to use a strategy that calls your data modules to get a user by username and password. Otherwise, you will set your cookies manually if the username and password is successful.

If the user cannot be authenticated, they will be redirected to the `/`route and an error message must be displayed.
If the user has been authenticated, they will be redirected to `/private` and details will be shown about the user.

GET /private

This route will be simple, as well. You will protect the `/private`route with your authentication middleware / Passport to only allow validated users to login.

When logged in, make sure that your user login info or Passport provides you access to the `user` property on your `request` data in express. Using the `req.user` property, you will make a simple view that displays all details **except** the password for the currently logged in user.

Users

You will use the following information to compose your users. **For the sake of this assignment and focusing on authentiction, you will store them in memory and not in a database**; the data module methods you create to access and search for your users must, however, return promises.

For example, you may store, in your `users.js` file, something like this:

```
const users = [
 { _id: "1245325124124", username: "masterdetective123", hashedPassword: "THE HASH", firstName: "Sherlock", lastName: "holmes" }, // etc, dont forget the other data
 { _id: "723445325124124", username: "lemon", hashedPassword: "THE HASH", firstName: "Elizabeth", lastName: "Lemon" }, // etc, dont forget the other data
]
```

Remember, all passwords must be hashed at all times using an algorithm such as bcrypt

You do **not** need to create a signup form for users! Simply add these users, with any associated data you may need, with **hashed**passwords to an array in memory.

For the sake of simplicity of the assignment, I have supplied you with bcrypted passwords through 16 salt rounds. You may hardcode the hashes, but not the actual passwords, in your data modules. The passwords listed below are the passwords you will input into the login form that need to work.

User 1: Sherlock Holmes

username: masterdetective123

First Name: Sherlock

Last Name: Holmes

Profession: Detective

Bio: Sherlock Holmes (/ˈʃɜːrlɒk ˈhoʊmz/) is a fictional private detective created by British author Sir Arthur Conan Doyle. Known as a "consulting detective" in the stories, Holmes is known for a proficiency with observation, forensic science, and logical reasoning that borders on the fantastic, which he employs when investigating cases for a wide variety of clients, including Scotland Yard.

Password: elementarymydearwatson

Hashed Password: $2a$16$7JKSiEmoP3GNDSalogqgPu0sUbwder7CAN/5wnvCWe6xCKAKwlTD.

User 2: Liz Lemon

username: lemon

First Name: Elizabeth

Last Name: Lemon

Profession: Writer

Bio: Elizabeth Miervaldis "Liz" Lemon is the main character of the American television series 30 Rock. She created and writes for the fictional comedy-sketch show The Girlie Show or TGS with Tracy Jordan.

Password: damnyoujackdonaghy

Hashed Password: $2a$16$SsR2TGPD24nfBpyRlBzINeGU61AH0Yo/CbgfOlU1ajpjnPuiQaiDm

User 3: Harry Potter

username: theboywholived

First Name: Harry

Last Name: Potter

Profession: Student

Bio: Harry Potter is a series of fantasy novels written by British author J. K. Rowling. The novels chronicle the life of a young wizard, Harry Potter, and his friends Hermione Granger and Ron Weasley, all of whom are students at Hogwarts School of Witchcraft and Wizardry . The main story arc concerns Harry's struggle against Lord Voldemort, a dark wizard who intends to become immortal, overthrow the wizard governing body known as the Ministry of Magic, and subjugate all wizards and Muggles.

Password: quidditch

Hashed Password: $2a$16$4o0WWtrq.ZefEmEbijNCGukCezqWTqz1VWlPm/xnaLM8d3WlS5pnK

Requirements

1. All general requirements from previous labs apply.