

A Simple User Login System

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

This lab requires a good deal of planning, but relatively little code to be written out. You will implement your own user login system, without using node modules like Passport.

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. The form used to submit to the server must have an id of login-form. The input for the username must have a name of username; the input for the password must have a name of password.

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 the user provides a successful username / password combination, you will set a cookie named AuthCookie. This cookie must be named AuthCookie or your assignment will receive a major point deduction. After logging in, you will redirect the user to the /private route.

If the user does not provide a valid login, you will render the login screen once again, and this time show an error message to the user explaining that they did not provide a valid username / password.

GET /private

This route will be simple, as well. You will protect the /private route with your authentication middleware to only allow valid, logged in users to see this page. If a user is not logged in, you will return an HTML page saying that the user is not logged in, and the page must issue a status code of 403.

If the user is logged in, you will make a simple view that displays all details except the password for the currently logged in user.

GET /logout

This route will expire the AuthCookie and inform the user that they have been logged out. It will provide a URL to the / route.

Users

You will use the following information to compose your users. For the sake of this assignment and focusing on authentication, 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/5wnvCWe6xCKAKw1TD.

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\$SsR2TGP24nfBpyR1BzINeGU61AH0Yo/Cbgf01U1ajpjnPuiQaiDm

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

General Note

For the sake of grading properly, when you start your server, please have the process send the following message after the app was listening on the proper port:

```
if (process && process.send) process.send({done: true}); // ADD THIS LINE
```

Like so:

```
app.listen(3000, function() {  
  console.log("Your server is now listening on port 3000! Navigate to http://localhost:3000 to access it");  
  
  if (process && process.send) process.send({done: true}); // ADD THIS LINE  
});
```

We will be unable to grade your assignments properly without this line.

Requirements

1. All general requirements from previous labs apply.