CSCI 4131 – Internet Programming (Version 2, 4/16)
Homework Assignment 6

Due Friday 4/19

Late Submissions (WITHOUT PENALTY) accepted until 4/22

Description

Assignment 5 introduced web development using Node.js. This assignment will build upon what you have developed with assignment 5. The objective of this assignment is to develop a basic website using Express. Express is an application framework that simplifies the development of node.js server-side applications, and it is the most widely used Node.js application framework for doing so. Typical features of Express are:

- routing: a simple way to map URLs and http verbs to code paths on the Node.js server
- easy methods for parsing http requests and building http responses:

The following are ***some of the resources*** *you should use to familiarize yourself with Express:*

- Essential
  - Installing Express
  - Hello world example of Express
  - Basic routing in Express
  - Serving static files in Express
- Additional References
  - Express website
  - FAQ
  - Routing in Express
  - API Reference

This assignment will also introduce you to SQL and the MySQL database (so upon successful completion, you will have developed a FULL-STACK application).

The following are resources you should review to get familiar with SQL, MYSQL, and MYSQL/Node.js

➢ https://www.w3schools.com/sql/
➢ https://www.w3schools.com/sql/sql_ref_mysql.asp
➢ https://www.w3schools.com/nodejs/nodejs_mysql.asp

Preparation and Provided Files

I. The first step will be to get Node.js and MySQL running on CSE lab machines. This can be accomplished as follows:

1. Log into a CSE lab machine remotely (by SSH, VOLE, or in person).
2. Most of the CSE lab machines run version 12.20.0 (or similar version) of Node.js.
3. Type the following command to check the availability and the version of Node.js on the machine:

   ```
   node -v
   ```

   And this will display the current installed version.

4. To use the MYSQL database, you will need a database user id and password. Your MYSQL database user id (alphanumeric) and password (numeric) can be found on the class Canvas site in your Grades menu. Check the column named: **DATABASE INFORMATION**. You will find your ***alphanumeric*** user id (alpha-numeric name) and ***numeric*** password in the comments section.

5. At the terminal, type the following command to login to MySQL and check whether it's active

*mysql -u**your_database_user** -hcse-mysql-classes-01.cse.umn.edu -P3306 -p **your_database_user***

   NOTE: There is a space between the lowercase –p, and your_database_user id, and your database user id is used in 2 places

   Replace ***your_database_user*** with the database id provided to you before hitting enter. ***your_database_user*** can be accessed on canvas "Database Information".

   When prompted for a password, **enter the NUMERIC password** provided to you.

6. After successful login, you should see the prompt:

   mysql>

II. The second step is to create a Node.js (Express) project for this assignment. This can be accomplished as follows:

1. Open the terminal on a CSE lab machine (in person, via Vole, or SSH)

2. Create a directory named <x500id_hw06> by typing the following command:

   ```
   mkdir yourx500id_hw06
   ```

3. Go inside the directory by typing the command: `cd  yourx500id_hw06`

4. Having a file named *package.json* in Node.js project makes it easy to manage module dependencies and makes the build process easier. To create *package.json* file, type the following command: `npm init`

5. The *npm init* command will prompt you to enter the information. Use the guidelines on the next page to enter the information (The things that you need to enter are in **bold** font. Some fields can be left blank.):

```
name: (yourx500id_hw06) yourx500id_hw06

version: (1.0.0) <Leave blank>

description: Assignment 6

entry point: (index.js) <Leave blank> (You will create an index.js file
for your use)

test command: <Leave blank>

git repository: <Leave blank>

keywords: <Leave blank>

author: yourx500id

license: (ISC) <Leave blank>
```

6. After filling in the above information, you will be prompted to answer the question: "`Is this ok? (yes)`". Type **yes** and hit enter.

7. The listing of all the available files in the directory (obtained by typing `ls -al followed by the <enter> key`) should display the following entry:

```
-rw------- 1 yourid CSEL-student 209 Apr 6 17:33 package.json
```

8. Install Express by typing the following command:

```
npm install --save express
```

9. Install mysql for javascript by typing the following command:

```
npm install --save mysql
```

10. You can use any npm module that you deem fit for this assignment. The npm modules that will be useful for this assignment and should be installed are:
    - pug(*npm install --save pug*)
    - body-parser (*npm install --save body-parser*)
    - express-session (*npm install --save express-session*)
    - bcrypt (*npm install --save bcrypt*)

11. You are free to decide your own project structure for this assignment.

III. <u>Database setup:</u>

1. The following files have been provided to you for this assignment:
   - create_accounts_table.js
   - insert_into_accounts_table.js
   - create_events_table.js

2. Download the files listed above and move them to `yourx500id_hw06` directory.
3. Edit each of the files and insert your database id and numeric password, which you can find on Canvas in your grades in the comments portion of column named: **DATABASE Information**
   Set the permissions on the files and directories to `rwxr-xr-x` (i.e., **`chmod 755 filename`**)
   Note, *for steps 4,5, and 6 below you may have to put each of the files in a separate directory and do an* ***npm init****, and* ***npm install --save mysql*** *before running node.js*

4. At the terminal, type the following command to create the MySQL table: tbl_accounts

   ```
   node create_accounts_table.js
   ```

   This table will be used to store your encrypted login credentials.

5. At the terminal, type the following command to insert values for acc_name, acc_login, acc_password into tbl_accounts table:

   ```
   node insert_into_accounts_table.js
   ```

   You will use the values you choose (we have provided default values) for acc_login and acc_password to login to the website. Keep the values in a safe place and do not share them with anyone.

6. At the terminal, type the following command to create the MySQL table: tbl_events

   ```
   node create_events_table.js
   ```

   This table will be used to store the details of the events.

7. Add the provided index.js file or create your own index.js file. If using the provided index.js you can start the server by running: ***`node index.js`***

   Then enter: http://localhost:9007/ `in` your browser's URL bar, and you should be able to see the welcome page. If your index.js file is using a different port make sure to change 9007 to whichever port you have selected.

You are now ready to start designing and implementing the website functionality for the assignment!!!!

3 Functionality

Your website will have 4 pages:

- A **Welcome** Page (provided)
- A **Login** page.
- A **Schedule** Page (You should be able to use (and update) your webpage from Homework 5)
- An **Add Event** page (You should be able to use (and update) your webpage from Homework 5).

All html pages will need to follow the Pug template either by being converted using a converter or written that way.

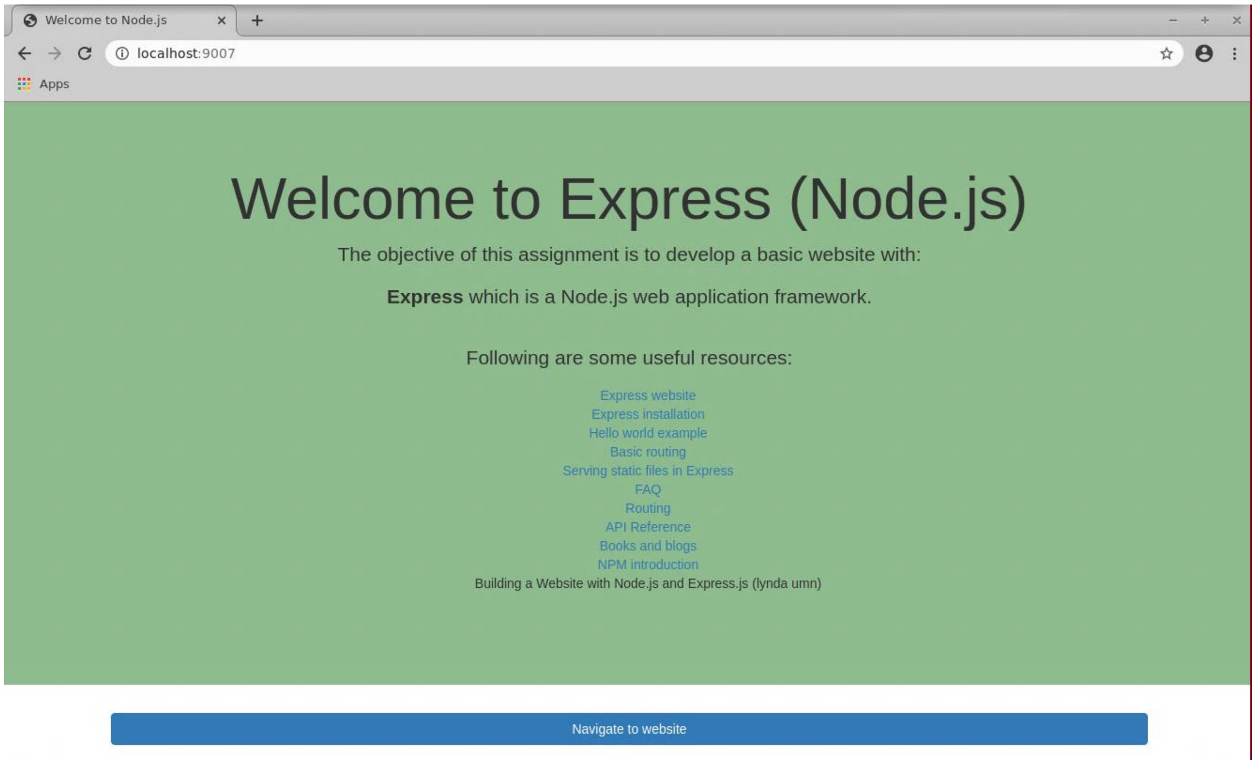
A logout button is required to end the current session once a user has logged in and should be present and functional on:

- The **Schedule** page
- The **Add Event** page

NOTE: For this assignment, you will need to develop the entire website including frontend (HTML pages, CSS, JavaScript) and backend (Express server) + MySQL database.

The pages below specify the functionality we have provided, and the functionality you must develop.

Welcome Page



Welcome to Express (Node.js)

The objective of this assignment is to develop a basic website with:

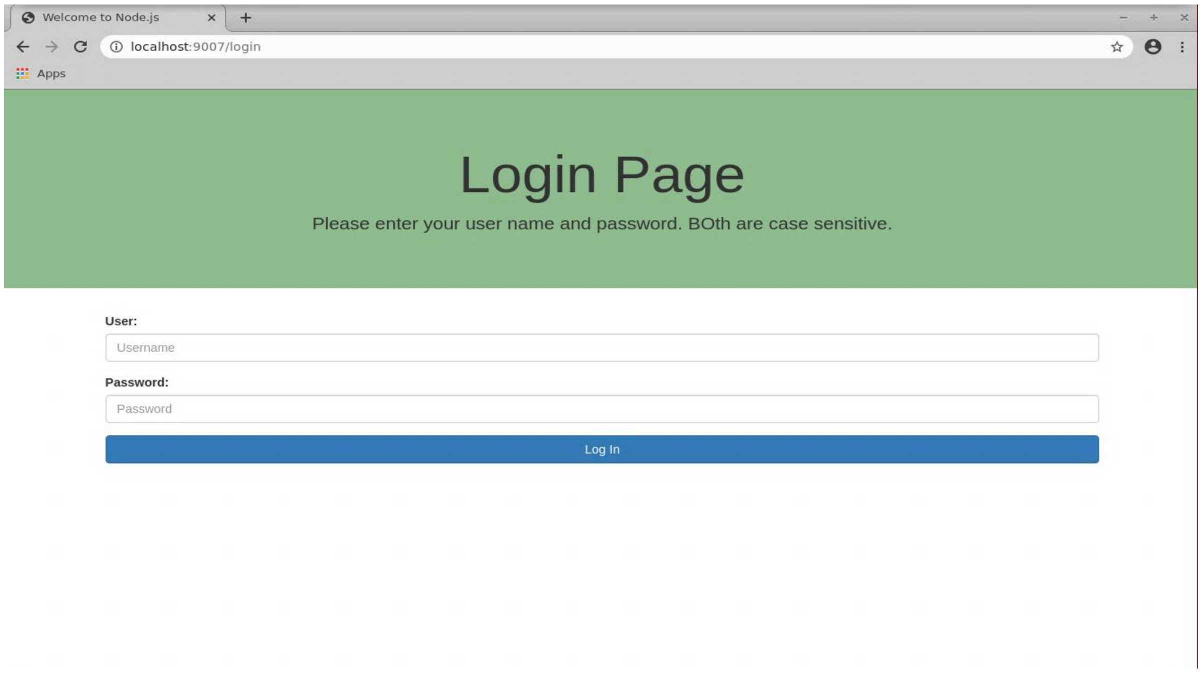
Express which is a Node.js web application framework.

Following are some useful resources:

Express website
Express installation
Hello world example
Basic routing
Serving static files in Express
FAQ
Routing
API Reference
Books and blogs
NPM introduction
Building a Website with Node.js and Express.js (lynda umn)

- The Welcome page is already provided to you and is displayed when the default route "/" is called. (for example, by typing http://localhost:portnumber into your browser's address bar)
- When you click on the Navigate to website button, the `/login` route in your Node.js/Express server (found in the file: `index.js` we have provided) will be called. You need to develop all the remaining functionality.

Login page



Login Page

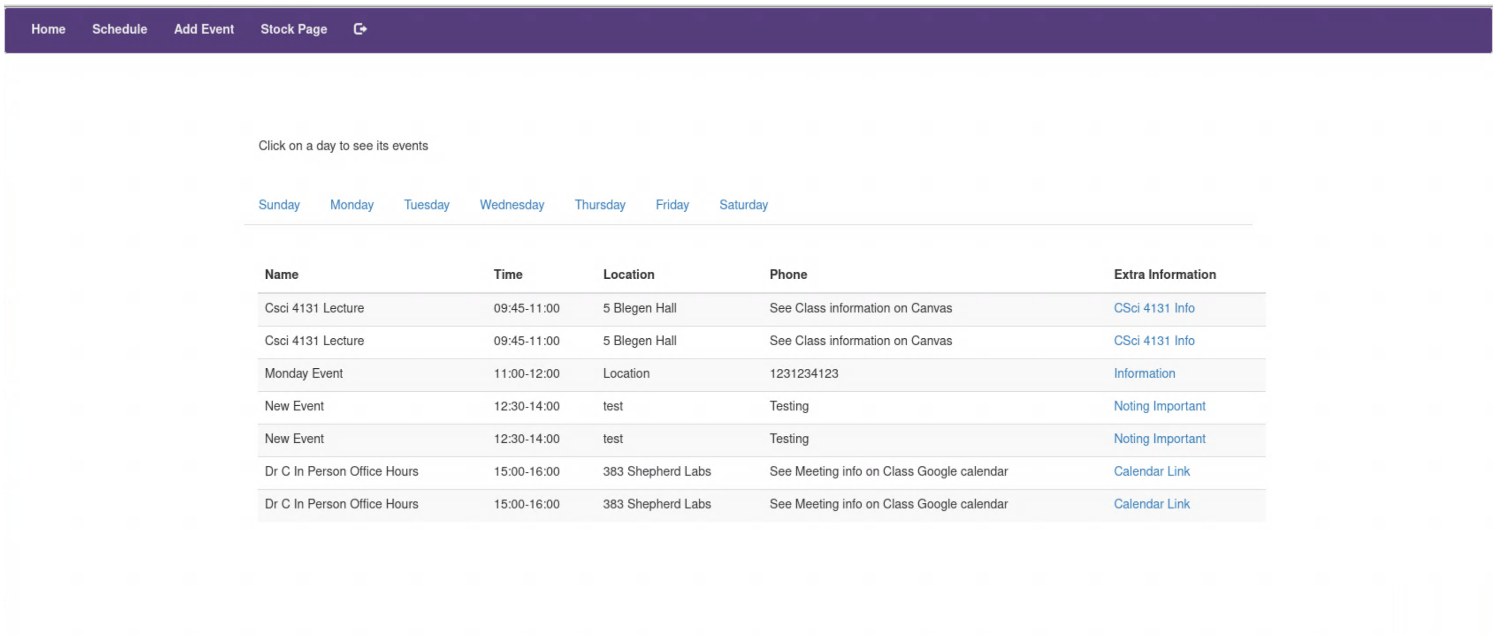
Please enter your user name and password. BOth are case sensitive.

- The Login page should be named **login.html**.
- If the page is accessed by a user that has already logged in, the user should be routed to the "Schedule" page.
- The Login page should have a form with two fields: "User", and "Password"
- Both of these fields are mandatory (required).
- When the submit button is clicked, an Fetch request carrying the value entered for "User" and "Password" should be sent to the server for user credentials validation before allowing further access to the website.
- The server will validate the values obtained from the form against the *acc_login*, and *acc_password* fields stored in *tbl_accounts*. The Server should compare the bcrypt password-hash of the password string it obtains from the form with the one stored in the database table `tbl_accounts`.
    - ***Note:*** *your implementation must use the bcrypt node module.*

- Upon successful validation, server should
    - Create a user session (*Note: this requires an express-session module*).
    - Send a response back to the client indicating successful validation.
- If the validation fails, server should:
    - Send a response back to the client indicating validation failure. Then your webpage needs to display an error information.

- If a successful response is received from the server, the user should be routed to the "Schedule" page, otherwise an appropriate error message should be displayed to the user (Check screenshots towards the end of this assignment)

Schedule Page



Click on a day to see its events

Sunday Monday Tuesday Wednesday Thursday Friday Saturday

Name	Time	Location	Phone	Extra Information
Csci 4131 Lecture	09:45-11:00	5 Blegen Hall	See Class information on Canvas	CSci 4131 Info
Csci 4131 Lecture	09:45-11:00	5 Blegen Hall	See Class information on Canvas	CSci 4131 Info
Monday Event	11:00-12:00	Location	1231234123	Information
New Event	12:30-14:00	test	Testing	Noting Important
New Event	12:30-14:00	test	Testing	Noting Important
Dr C In Person Office Hours	15:00-16:00	383 Shepherd Labs	See Meeting info on Class Google calendar	Calendar Link
Dr C In Person Office Hours	15:00-16:00	383 Shepherd Labs	See Meeting info on Class Google calendar	Calendar Link

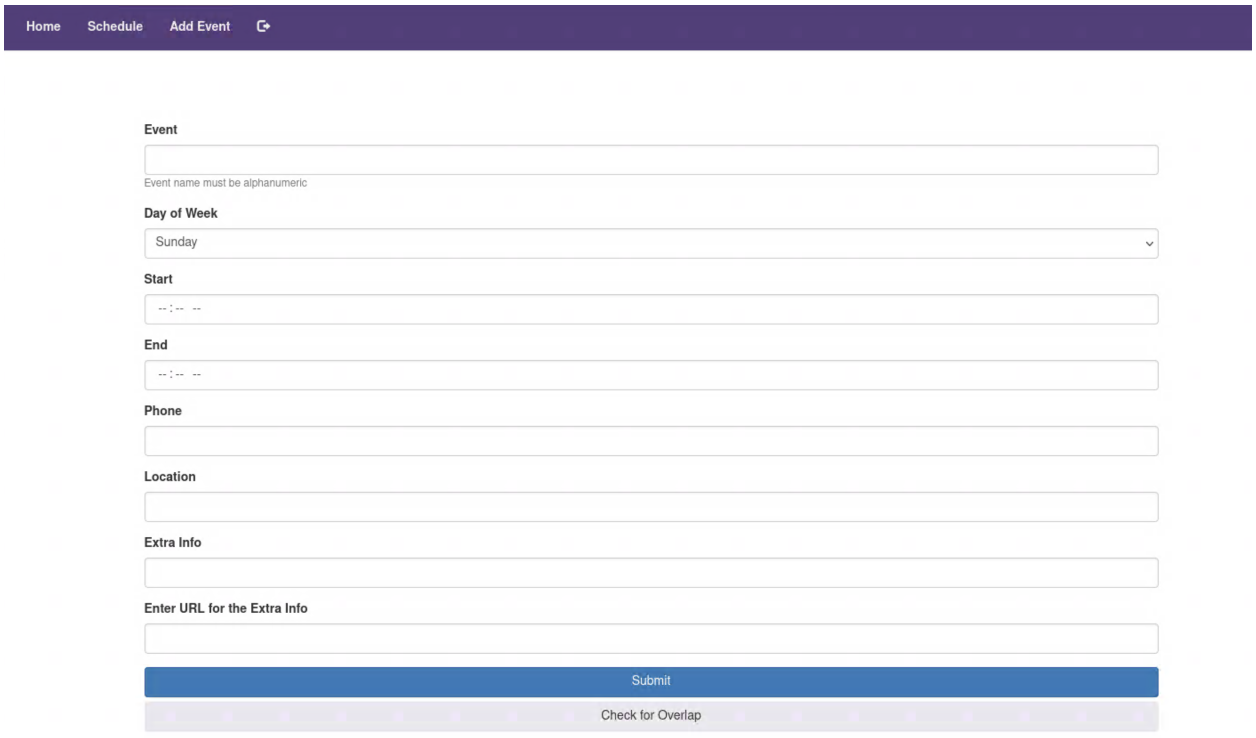
This page should be very similar to the one developed in homework assignment 5. The screenshot displayed above shows the schedule after clicking “Monday”. It should be named **schedule.html**.

If this page is accessed **without** a valid login, the user should be routed to the “**Login**” page.

For this assignment, you will have to change the server functionality to retrieve the schedule information from the MySQL database (*tbl_events*) with a Fetch request and return it in a JSON object, instead of reading the schedule information from a local JSON file and returning it as was done in HW 5.

Note: you need to make sure the events in the Schedule Page are sorted in ascending order (by their start time, as shown on the screenshots).

Add Event page



Event

Event name must be alphanumeric

Day of Week: Sunday

Start: --:-- --

End: --:-- --

Phone

Location

Extra Info

Enter URL for the Extra Info

- You can start from the form provided in Homework 5 for the 'Add Event' page(**addEvent.html**).
- If this page is accessed **without** a valid login, the user should be routed to the **"Login"** page.
- The page should have a navigation bar with a logout button.
- Upon clicking submit, the form data should be posted to the server with a Fetch request.
- The server should insert the data received from the client into the following table: *tbl_events* instead of the file schedule.json used in HW 5 (*Hint: you will need use functionality provided by the mysql module*)
- The mapping between form fields and table columns could be (NOTE: you are free to modify this mapping as you deem necessary to comply with the functional requirements of this assignment):
    - day: day
    - event: event
    - start: start
    - end: end
    - location: location
    - phone: phone
    - info: info
    - url: URL
- Upon successful insertion, the server should return a redirect response to the browser to display the "Schedule" page.

Logout button

Upon clicking the logout button on the menu bar (pictured below), the session should be destroyed and the server should send a redirect message to the browser to display the **Login** page. ***The button does not need to look the same as pictured below*** (for example, you can create and use a button with the word Logout displayed on it) but should be evident it is a logout button.

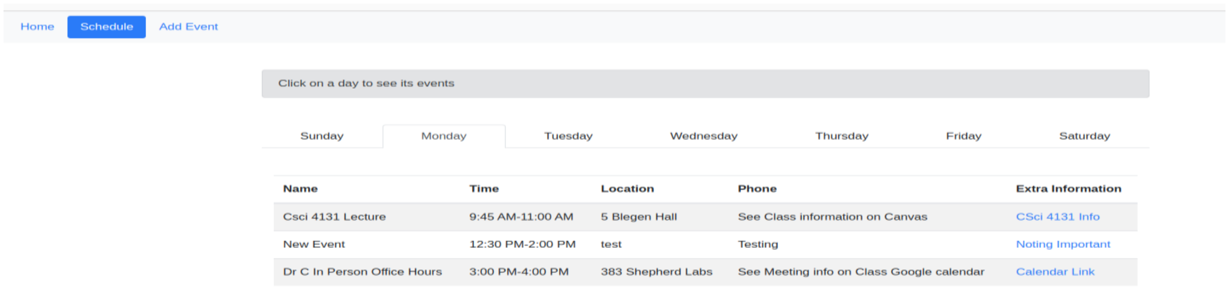
Pug

All of your html code should be submitted as pug instead for this assignment. The use of a converter is permitted and encouraged for html already written. HTML to PUG Converter Online tool is an option to use but any site that converts it correctly is allowed.

To take full advantage of Pug it is advised to write it by hand without the converter, such as for the “Login” page.

Bonus: Quick Setup

Go back to your HW5, go to the schedule page, and click on one of the days with events on the schedule:

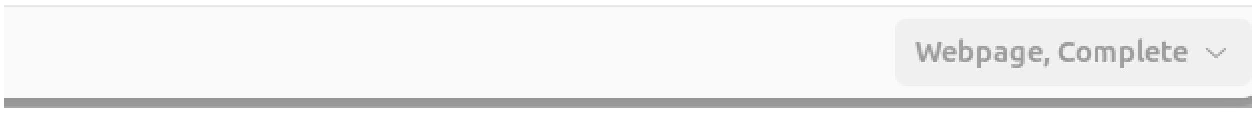


Click on a day to see its events

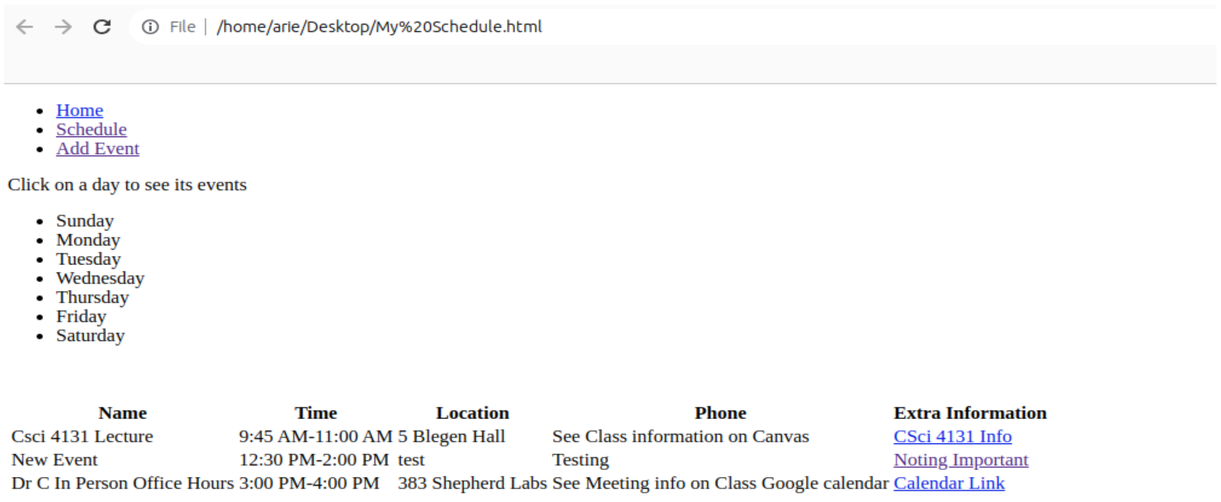
Sunday Monday Tuesday Wednesday Thursday Friday Saturday

Name	Time	Location	Phone	Extra Information
Csci 4131 Lecture	9:45 AM-11:00 AM	5 Blegen Hall	See Class information on Canvas	CSci 4131 Info
New Event	12:30 PM-2:00 PM	test	Testing	Noting Important
Dr C In Person Office Hours	3:00 PM-4:00 PM	383 Shepherd Labs	See Meeting info on Class Google calendar	Calendar Link

Right-click on the page, and select “Save as” (Chrome) or “Save page as” (Firefox). If you are using Chrome, specify the page as “Webpage, Complete”.



You will get a html page like this:



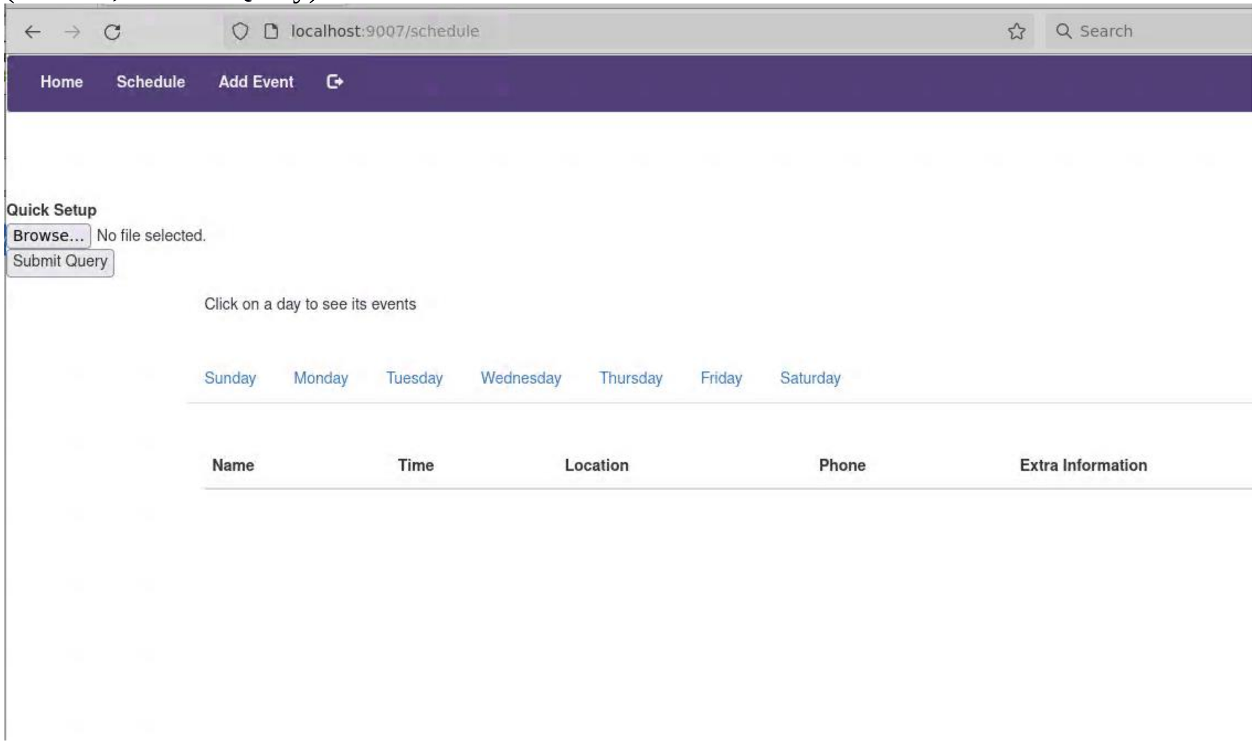
- Home
- Schedule
- Add Event

Click on a day to see its events

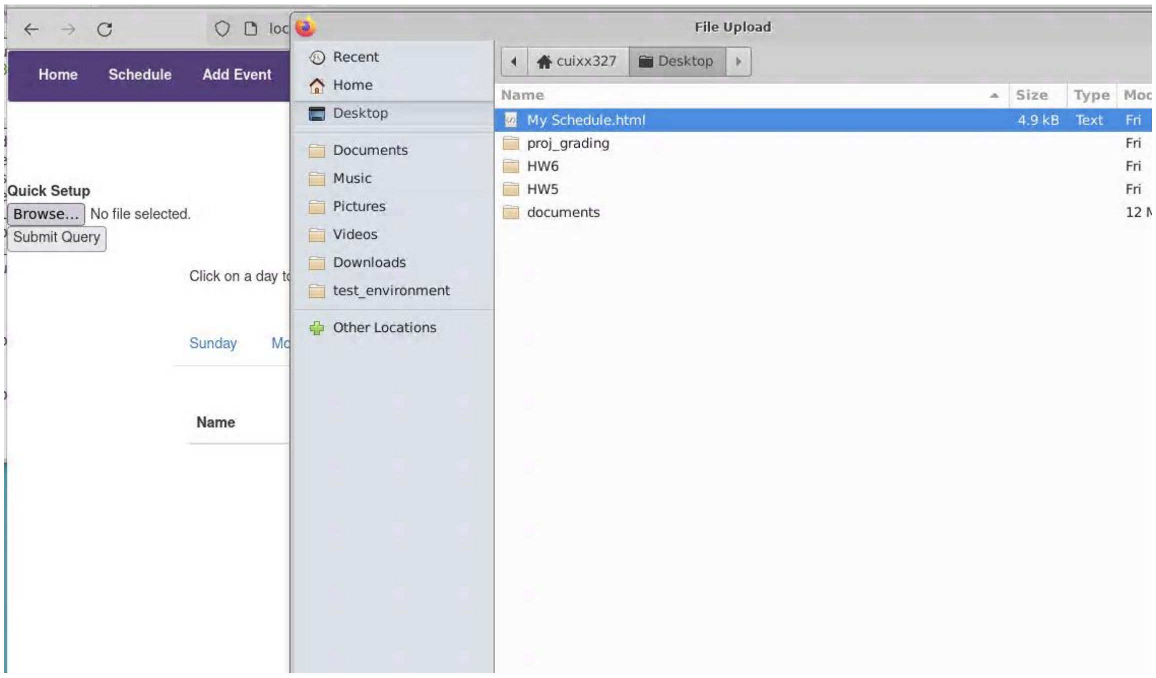
- Sunday
- Monday
- Tuesday
- Wednesday
- Thursday
- Friday
- Saturday

Name	Time	Location	Phone	Extra Information
Csci 4131 Lecture	9:45 AM-11:00 AM	5 Blegen Hall	See Class information on Canvas	CSci 4131 Info
New Event	12:30 PM-2:00 PM	test	Testing	Noting Important
Dr C In Person Office Hours	3:00 PM-4:00 PM	383 Shepherd Labs	See Meeting info on Class Google calendar	Calendar Link

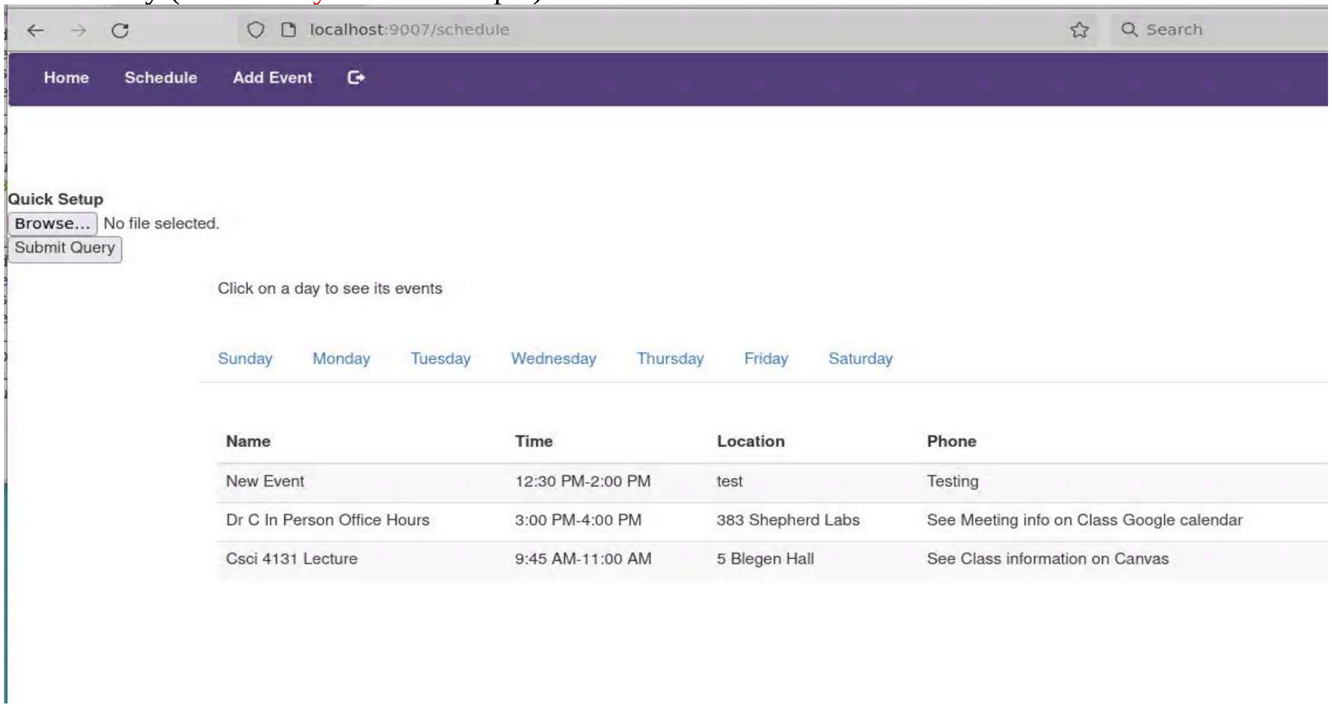
On your HW6, schedule page add buttons similar to the buttons shown in the picture below (Browse, Submit Query)



First, click on one of the days (for example, Wednesday), then, click on Browse and select the html you got from HW5 (on Wednesday):



After the file is selected (MySchedule.html in this example) click submit the query and the html file you selected should be submitted to the server, and your server should add these events to the selected day (Wednesday in this example).



Name	Time	Location	Phone
New Event	12:30 PM-2:00 PM	test	Testing
Dr C In Person Office Hours	3:00 PM-4:00 PM	383 Shepherd Labs	See Meeting info on Class Google calendar
Csci 4131 Lecture	9:45 AM-11:00 AM	5 Blegen Hall	See Class information on Canvas

Hints:
1. The node.js module formidable can be a helpful tool when uploading files.
2. You do not need to use Fetch for this bonus.

Submission Instructions

PLEASE ENSURE TO TEST YOUR CODE ON CSE LAB MACHINES.

You will need to submit all the files used to develop the website. This includes all the HTML, CSS, JavaScript, package.json, index.js, and any other files.

Towards this end, make a copy of your working directory: `yourx500id_hw06`. Rename the copied folder as **yourx500id_express**.

Create a README file inside yourx500id_express directory. This README file should include Your x500id, acc_login, and acc_password values from the insert_into_accounts_table.js file. Finally, compress (e.g., tar or zip) the **yourx500id_express** directory and submit it **via Canvas.**

We will use the acc_login and acc_password values to log in to your website. Ensure that these values are correct and can be used to access your website.

Please remove the **node_modules** folder from your submission! Do not leave it in your submission! Also, make sure that naming conventions are followed as outlined in previous sections!

AND, submit something no later than the late submission deadline to ensure you get credit for any functionality you have working. Submissions after the late submission deadline will not be accepted and will be assigned a grade of ZERO.

Evaluation

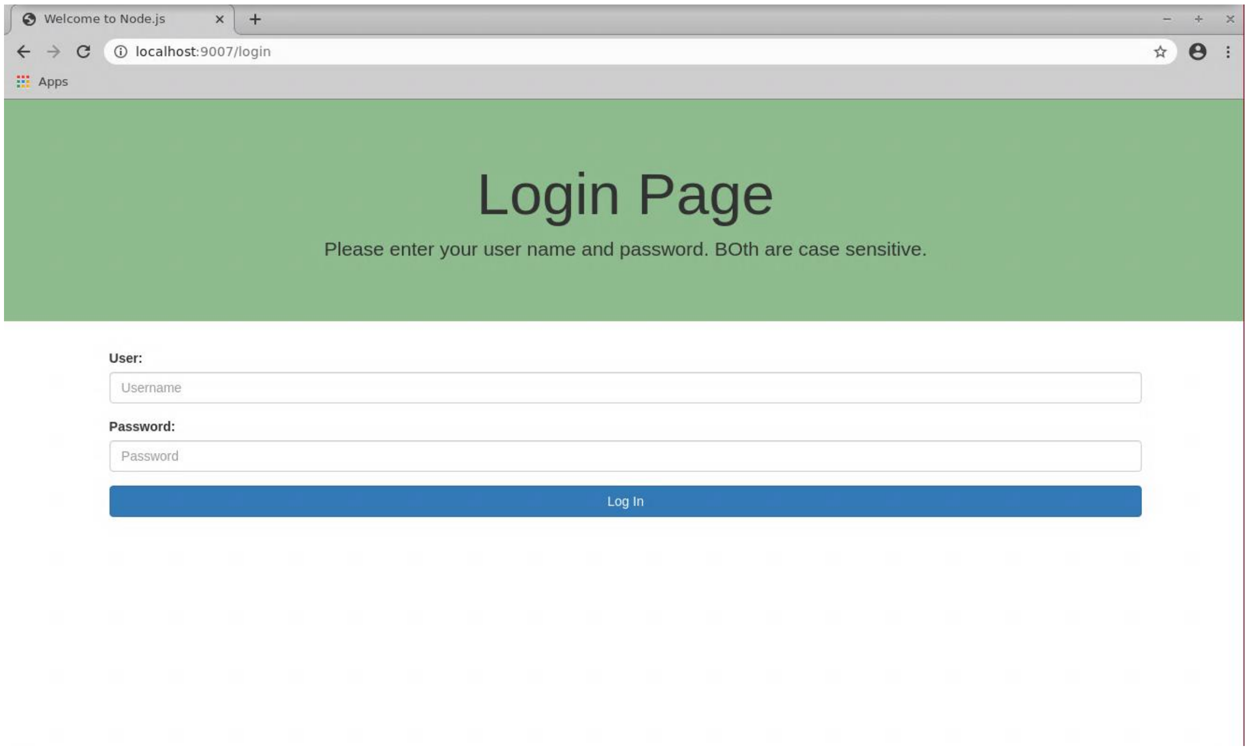
Your submission will be graded **FUNCTIONALLY** out of 100 points on the items listed below. Functional grading means that to receive credit, the functionality has to work correctly.

1. Submission instructions are met. **SPECIFICALLY** - remember to remove the **node_modules** folder **(5 points, really 5 points!)**
2. The “Schedule” and "Add Event” pages of your website redirect the user to the "Login" page automatically before authentication. **(10 points)**
3. The "Login" page shows the form elements and the submit button. **(5 points)**
4. Use Fetch login to send user credentials to the server, and redirect the user to the “Schedule” page after successful login validation by the server. **(10 points)**
5. If server login validation fails, an error message is displayed on the "Login" page and the browser displays the login page and error message. (check the additional screenshots)**(5 points)**
6. After successful login, the "Schedule" page displays correctly. **(5 points)**
7. The "Schedule" page gets the list of events from the server with Fetch (which the server gets from the database). These events are dynamically added to the table displayed in the user’s browser. **(10 points)**
8. The events shown on the “Schedule” page are correctly sorted. **(5 points)**
9. The user can add a new event to the database using the form present in the "Add Event" page. **(10 points) – changed: you can use fetch or a post request to an endpoint!!!**
10. The “Schedule” and “Add Event” pages display and have an operational navigation bar. **(5 points)**
11. When a new event is added through the “Add Event” page, the event data is stored in the MySQL database. Then the user is redirected to the "Schedule" page. **(5 points)**
12. All html is code is written in a pug template and not native html **(10 points)**
13. The logout functionality works correctly. **(5 points)**
14. **Use express to handle all http requests (10 points)**

15. **The BONUS Quick Setup functionality works correctly. (15 point BONUS)**

Additional Screenshots (See the following pages for examples)

Login Page


Welcome to Node.js

localhost:9007/login

Apps

Login Page

Please enter your user name and password. BOth are case sensitive.

User:

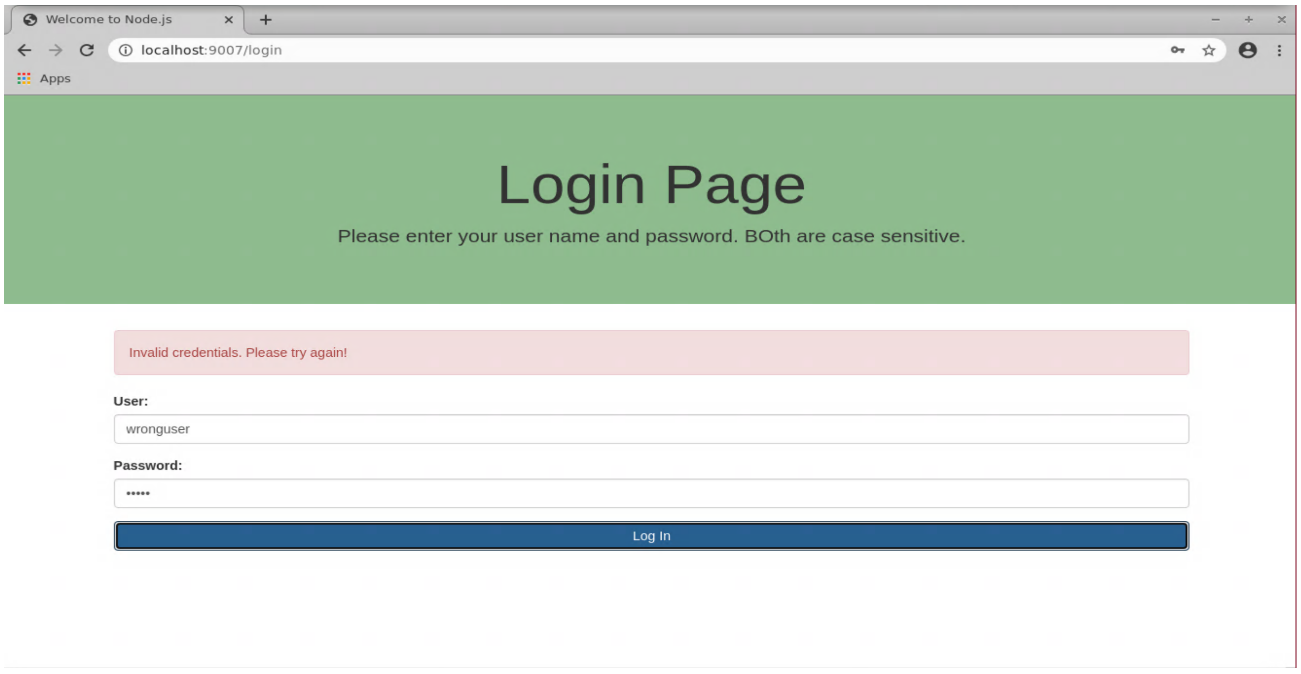
Username

Password:

Password

Log In

Invalid Login


Welcome to Node.js

localhost:9007/login

Apps

Login Page

Please enter your user name and password. BOth are case sensitive.

Invalid credentials. Please try again!

User:

wronguser

Password:

•••••

Log In