

# CSCI 4131 – Internet Programming

## Homework Assignment 6 (Version 2, Updated 4/6)

**Due Friday, April 17<sup>th</sup> at 3pm**

**Late Submissions (with Penalty) accepted until Saturday, April 18<sup>th</sup> at 6pm**

### Description

Assignment 5 provided an introduction to 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 application framework for doing so. Typical features of Express are:

- routing: a way to map URLs and http verbs to code paths
- 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](#)
  - [https://www.tutorialspoint.com/nodejs/nodejs\\_express\\_framework.htm](https://www.tutorialspoint.com/nodejs/nodejs_express_framework.htm)
- Additional Referenced
  - [Express website](#)
  - [FAQ](#)
  - [Routing in Express](#)
  - [API Reference](#)
  - [Building a Website with Node.js and Express.js](#) (Video tutorial, LinkedIn Learning)

This assignment will also introduce you to SQL and the MySQL database.

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/sql/sql_ref_mysql.asp)
- [https://www.w3schools.com/nodejs/nodejs\\_mysql.asp](https://www.w3schools.com/nodejs/nodejs_mysql.asp)
- SQL/MYSQL: Chapter 13 Sebesta

## 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 or VOLE).
2. Most of the CSE lab machines run version 12.16.1 of Node.js
3. Open the terminal and type the following command to add the Node.js module:  
`module add soft/nodejs`
4. The next step is to check the availability of Node.js. Type the following command to check the version of Node.js on the machine:  
`node -v`
5. This will display the current installed version.
6. To use the MYSQL database, you will need a database user id and password. Your MYSQL information can be found on the class Canvas site in your grades, in the column named MySQL Database and Password. Open the comments section. The numeric portion of your MySQL user name/id and your all NUMERIC password are separated by a space.
7. At the terminal, type the following command to add MySQL module:  
`module add soft/mysql`
8. At the terminal, type the following command to login to MySQL and check whether its active:

```
mysql -uyour_database_user -hcse-larry.cse.umn.edu -P3306 -p your_database_user
```

As specified above, you will find the numeric portion of your database user id and all NUMERIC database password on Canvas with your grades in the feedback section of the item named: MySQL Database and Password.

Replace `your_database_user` with the database id provided to you before hitting enter. **NOTE:** `your_database_user` will be in the format: C4131S20UXXX, where XXXX is the numeric portion of your database id/username (the first number in the comments section).

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

9. After successful login, you should see **mysql>** prompt.

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

10. Open the terminal on a CSE lab machine.
11. Create a directory named `<x500id_hw05>` by typing the following command:  
`mkdir yourx500id_hw06`

12. Go inside the directory by typing the following command:

```
cd yourx500id_hw06
```

13. 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*

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

```
name: (yourx500id_hw08) 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>
```

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

16. Listing all the available files in the directory should display the following:

```
-rw-r--r-- 1 he000242 CS-Grad 452 Mar 23 23:53 package.json
```

17. Install Express by typing the following command:

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

18. 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:

- mysql (*npm install --save mysql*)
- body-parser (*npm install --save body-parser*)
- express-session (*npm install --save express-session*)

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

**NOTE: We have provided a sample file for the server (index.js) which can be used for reference.**

### III. Database setup:

1. The following files have been provided to you for this assignment:
  - create\_accounts\_table.js
  - insert\_into\_accounts\_table.js
  - create\_contacts\_table.js
2. Download these files and move them to `yourx500id_hw06` directory.
3. Edit each of the files to include your database id and numeric password, which you can find on Canvas in your grades in the comments portion of column named: **UserID\_Password**
4. Set the permissions on the files to `rw-r-xr-x` (e.g., **chmod 755 filename**)
5. 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.

6. 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 chosen 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

7. At the terminal, type the following command to create the MySQL table: `tbl_contacts`

```
node create_contacts_table.js
```

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

At this point you are ready to start the website development.

## **3 Functionality**

Your website will have 5 pages:

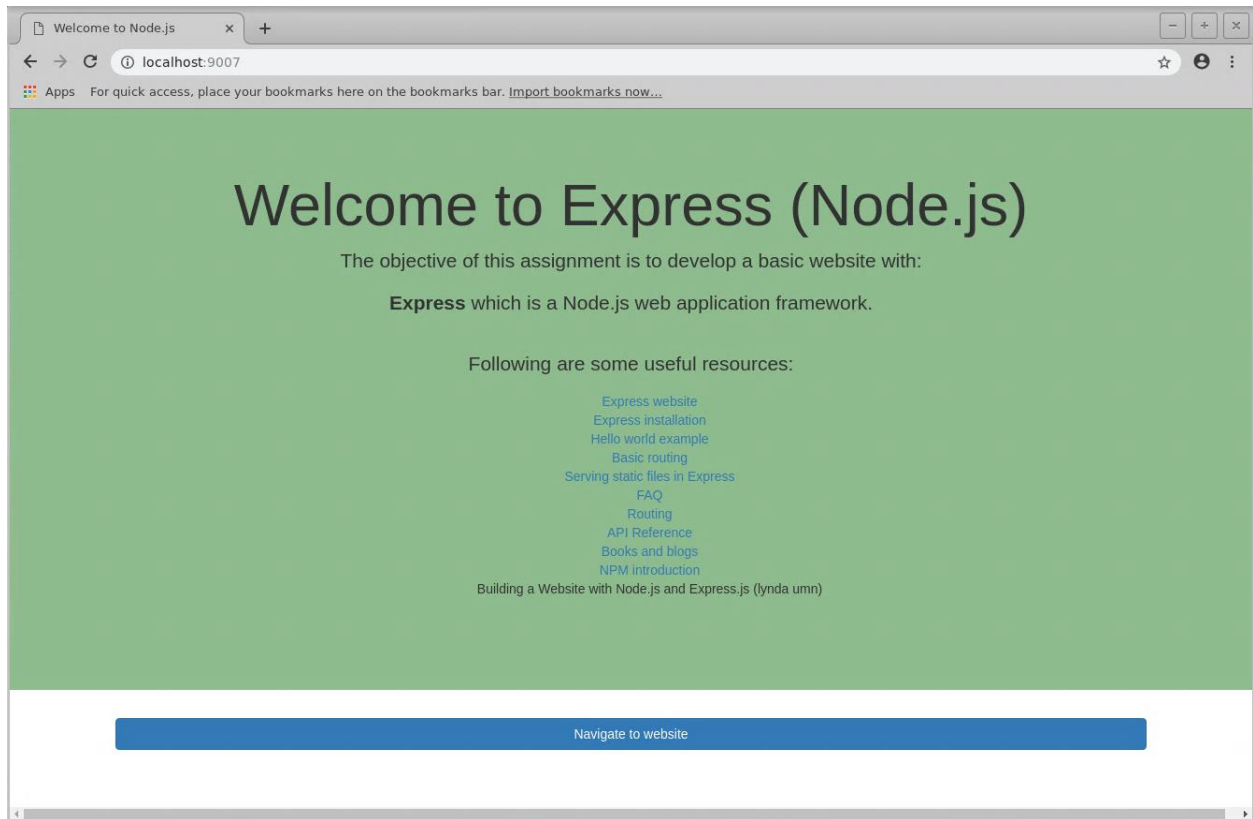
- A **Welcome** Page (provided)
- A **Login** page
- A **Contact** page
- An **Add Contact** page
- A **Stock** page

The Contact, Add Contact, Stock pages will have a **navigation bar with a logout button**.

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

**Pages 5 through 9 below specify the functionality provided, and the functionality you must develop.**

## Welcome Page



- The Welcome page is already provided to you and is displayed when the default route “/” is called.
- When you click on the Navigate to website button, the /login route on the Express (Node.js) will be called. You need to develop all the remaining functionality.

## Login page

Welcome to Node.js x +

localhost:9007/login

Apps For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...

# Login Page

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

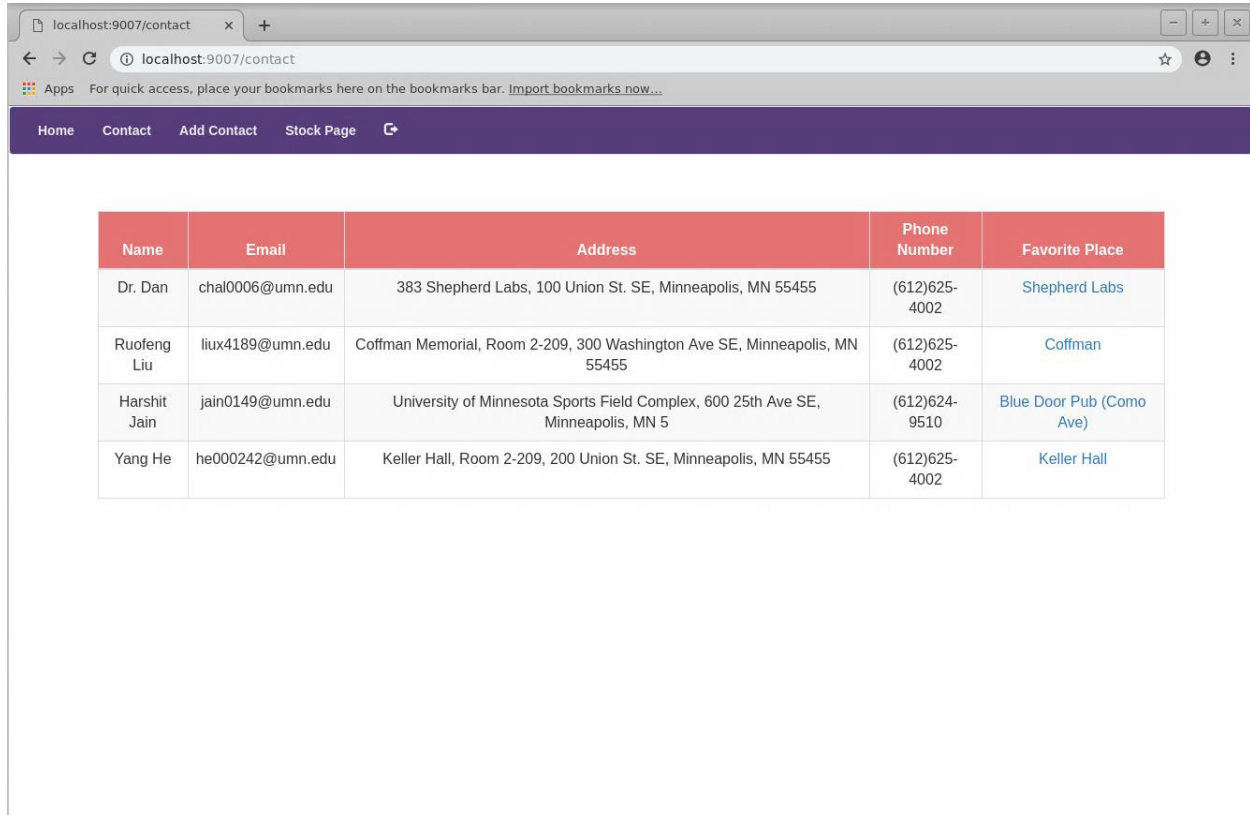
User:

Password:

Log In

- The Login page should have a form with two fields: “User”, and “Password”
- Both these fields are mandatory (required).
- When the submit button is clicked, the values entered for “User” and “Password” should be sent to the server for validation before allowing further access to website.
- The server will validate the values obtained from the form against the acc\_login, and acc\_password fields stored in tbl\_accounts. Passwords are stored in the MySQL database in a SHA256 hashed format in tbl\_accounts. The Server should hash the password string it obtains from the form using the SHA256 hash algorithm and compare it to the SHA256 hashed password stored in the database table tbl\_accounts.  
*(Hint: you can use crypto module to create the SHA256 hash. An example indicating how to use the SHA256 hashing algorithm can be found in the file we gave you to insert login information into the database named: **insert\_into\_accounts\_table.js**)*
- Upon successful validation, server should
  - Create a user session (*Hint: you can use 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.
- If successful response is received from server, user should be routed to “Contacts” page, otherwise an appropriate error message should be displayed to the user (Check screenshots towards the end of this assignment)

## Contact page



Name	Email	Address	Phone Number	Favorite Place
Dr. Dan	chal0006@umn.edu	383 Shepherd Labs, 100 Union St. SE, Minneapolis, MN 55455	(612)625-4002	<a href="#">Shepherd Labs</a>
Ruofeng Liu	liux4189@umn.edu	Coffman Memorial, Room 2-209, 300 Washington Ave SE, Minneapolis, MN 55455	(612)625-4002	<a href="#">Coffman</a>
Harshit Jain	jain0149@umn.edu	University of Minnesota Sports Field Complex, 600 25th Ave SE, Minneapolis, MN 5	(612)624-9510	<a href="#">Blue Door Pub (Como Ave)</a>
Yang He	he000242@umn.edu	Keller Hall, Room 2-209, 200 Union St. SE, Minneapolis, MN 55455	(612)625-4002	<a href="#">Keller Hall</a>

- If a user tries to access this page without a valid login, the user should be routed to the “**Login**” page.
- The page should have a navigation bar with a logout button.
- The table in this page should be dynamically populated.
- To achieve this, the server should provide a GET API which returns the list of contacts. This API will be very similar to the one developed in assignment 5. It will get the list of events by querying the table: *tbl\_contacts*.
- The client will call this API and populate the table using the data received from the server.

## Add Contact page

localhost:9007/addContact

← → ↻ localhost:9007/addContact

Apps For quick access, place your bookmarks here on the bookmarks bar. [Import bookmarks now...](#)

Home Contact Add Contact Stock Page

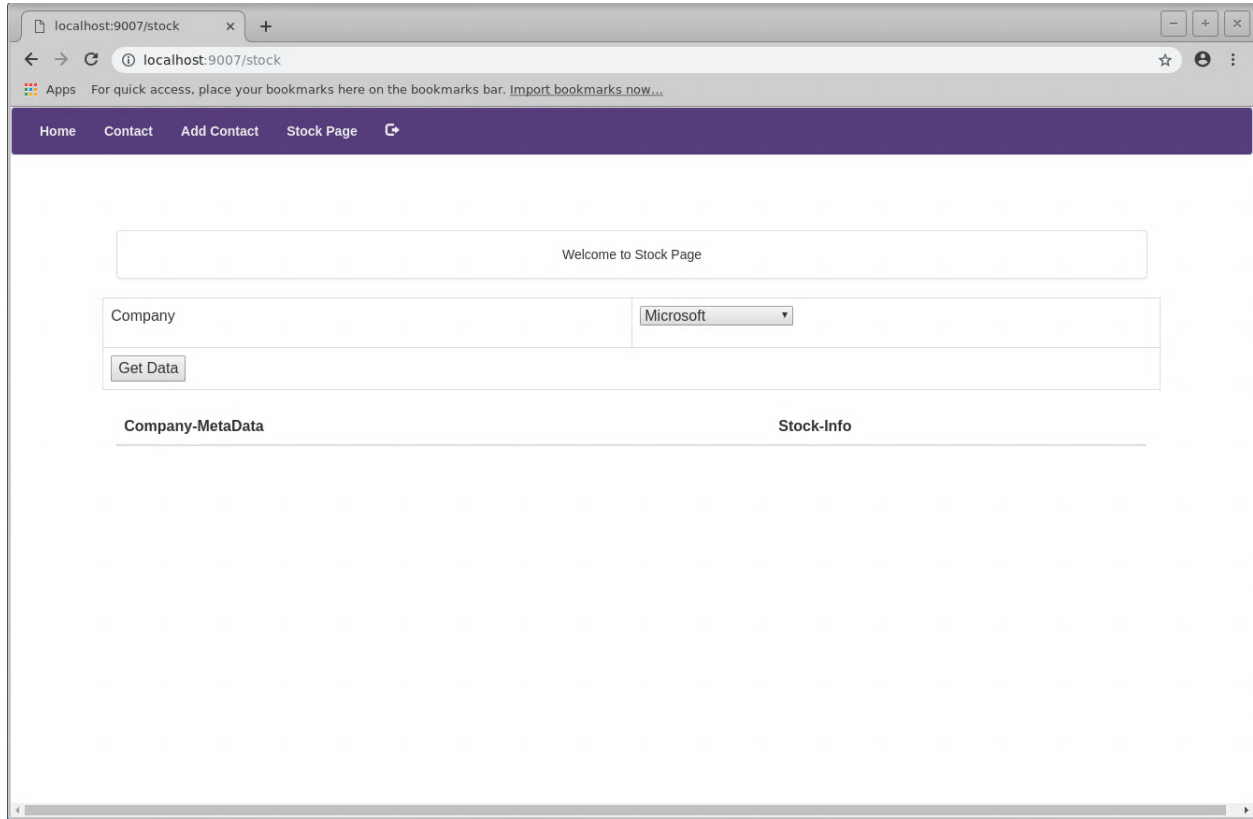
Name	<input type="text"/>
Email	<input type="text"/>
Address	<input type="text"/>
Phone Number	<input type="text"/>
Favorite Place	<input type="text"/>
Enter URL for the Favorite Place	<input type="text"/>

Submit

- You can use the form developed in the earlier assignments for the ‘Add Contacts’ page.
- 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.
- The server should insert the received data into the following table: *tbl\_contacts* (Hint: you can use mysql module)
- The mapping between form fields and table columns is:
  - contactName: **contact\_name**
  - email: **contact\_email**
  - address: **contact\_address**
  - phoneNumber: **contact\_phone**
  - favoritePlace: **contact\_favoriteplace**
  - favoritePlaceURL: **contact\_favoriteplaceurl**
- Upon successful insertion, the server should return a redirect response to the browser to display the “Contact” page.



## Stock page



You can use the ‘Stock’ page developed in the earlier assignments.

- 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.

## 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.



## 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 `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 login to your website. Ensure that these values are correct and can be used to access your website.

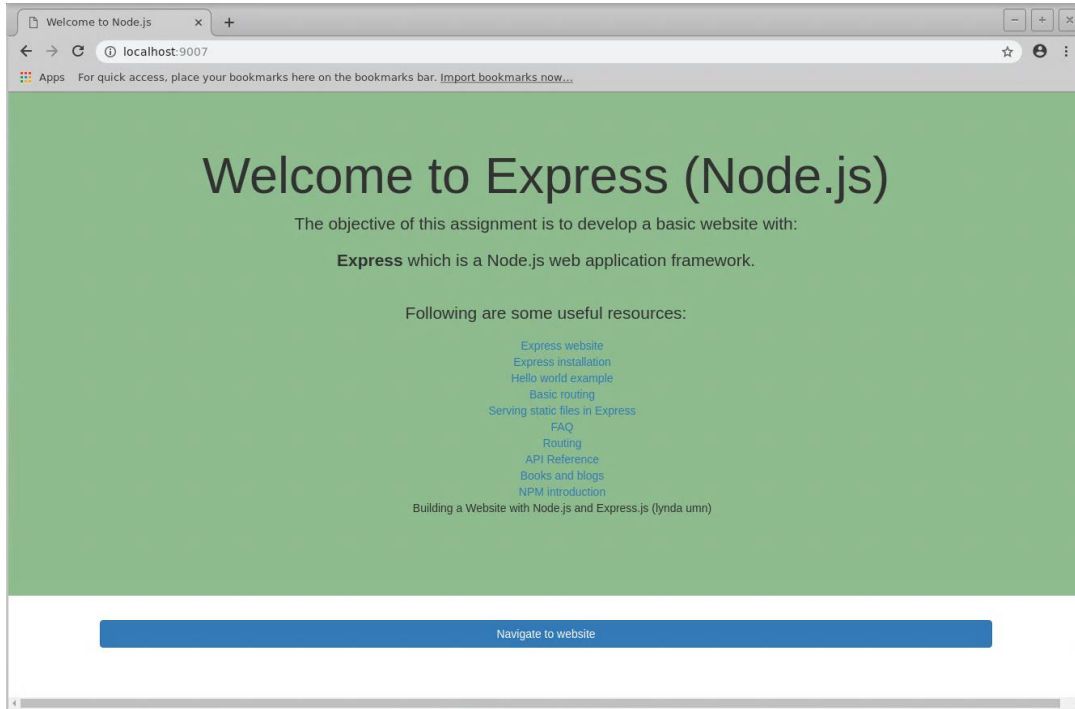
## Evaluation

Your submission will be graded out of 100 points on the following items:

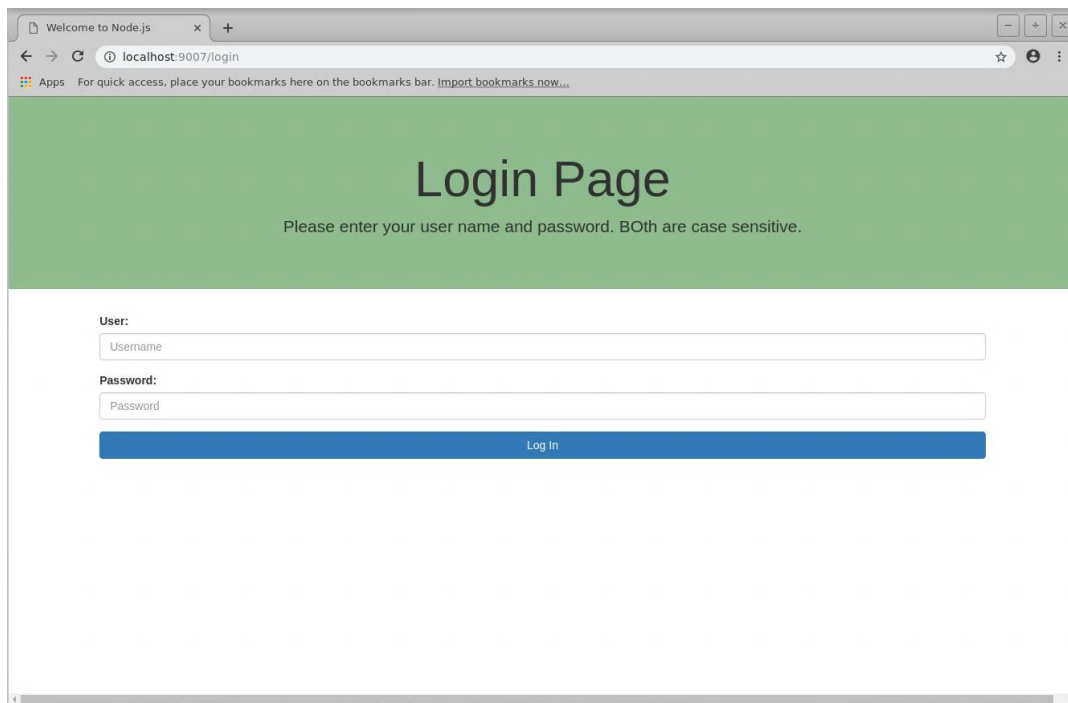
- **Submission instructions are met. (5 points)**
- The "Contact" and "Add Contact" and "Stock" pages of your website redirect the user to "Login" page automatically if a user accesses them before successfully logging in. **(10 points)**
- The "Login" page shows the form elements and submit button. **(5 points)**
- After successful login validation by the server, the "Login" page redirects the user to the "Contact" page. **(10 points)**
- If server login validation fails, an error message is displayed on the "Login" page and the browser displays the login page and error message. **(10 points)**
- After successful login, the "Contact" page displays correctly and has an operational navigation bar. **(5 points)**
- The "Contact" page gets the list of contacts from the server (which the server gets from the database). These contacts are dynamically added to the table displayed in the user's browser. **(20 points)**
- The user can add a new contact to the database using the form present in the "Add Contact" page. **(15 points)**
- The "Add Contact" page displays and has an operational navigation bar. **(5 points)**
- When a new contact is added using the "Add Contact" page, the contact data is stored in the MySQL database. Then the user is redirected to the "Contact" page, and the user's contacts are correctly displayed. **(5 points)**
- The "Stock" page displays and has an operational navigation bar. **(5 points)**
- The logout functionality works correctly. **(5 points)**

**Additional Screenshots (See the following pages for examples)**

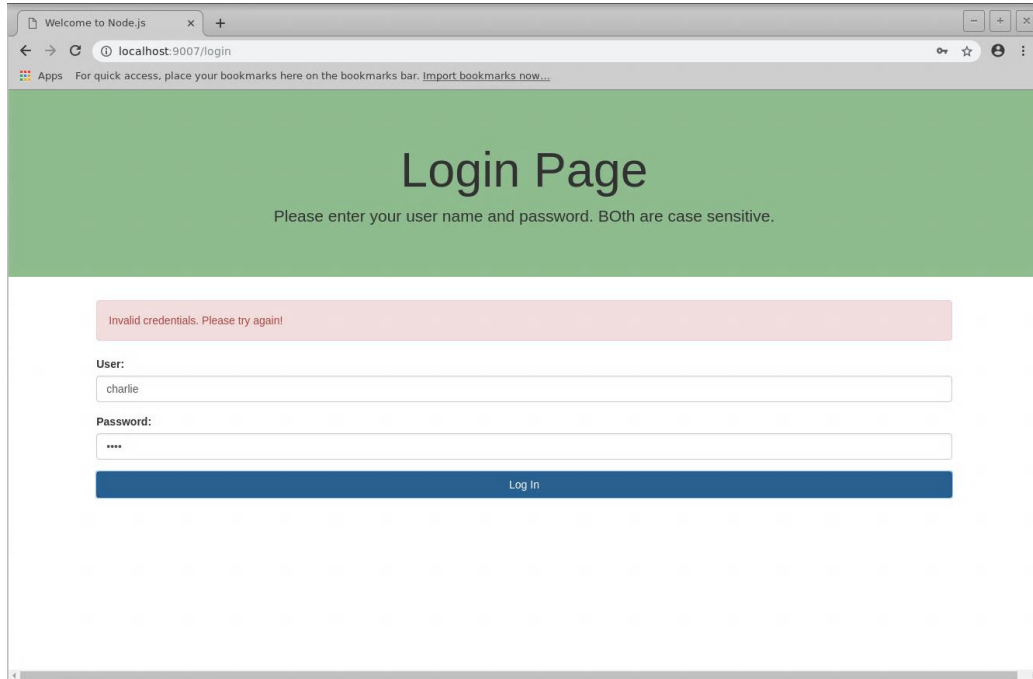
## Welcome Page



## Login Page



## Invalid Login



The screenshot shows a web browser window with the address bar at `localhost:9007/login`. The page has a green header with the text "Login Page" and "Please enter your user name and password. BOTH are case sensitive." Below this is a red error message box that says "Invalid credentials. Please try again!". There are two input fields: "User:" with the text "charlie" and "Password:" with masked characters "\*\*\*\*". A blue "Log In" button is at the bottom.

Welcome to Node.js x +

localhost:9007/login

Apps For quick access, place your bookmarks here on the bookmarks bar. [Import bookmarks now...](#)

# Login Page

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

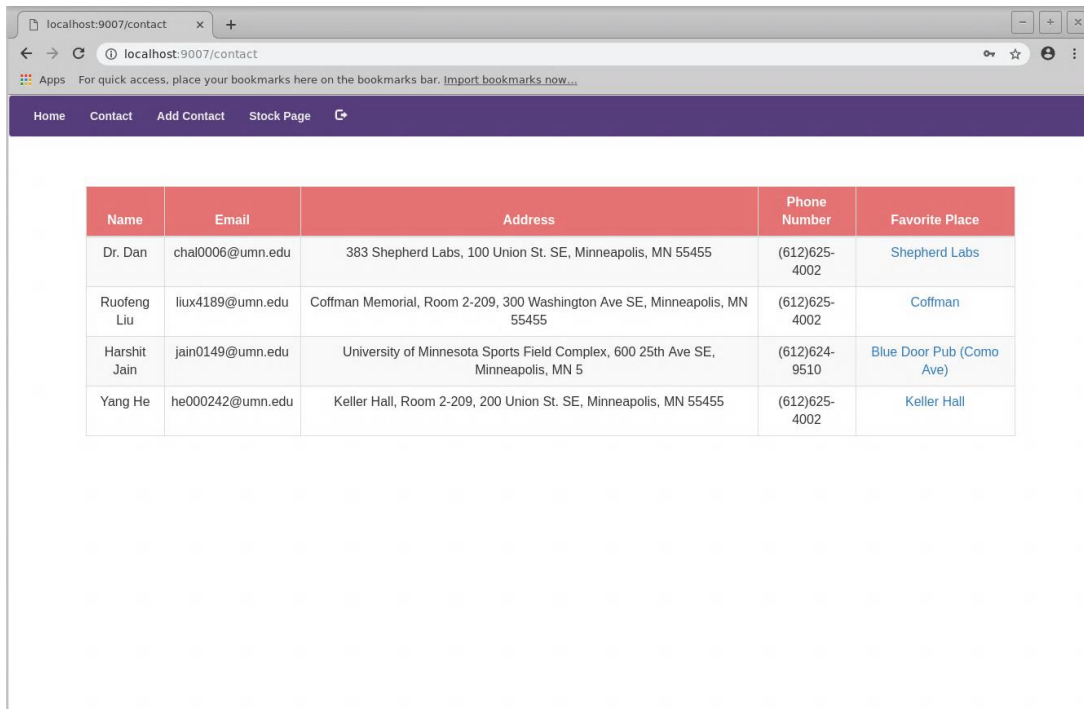
Invalid credentials. Please try again!

User:

Password:

Log In

## Contact Page



The screenshot shows a web browser window with the address bar at `localhost:9007/contact`. The page has a purple navigation bar with links: Home, Contact, Add Contact, and Stock Page. Below the navigation bar is a table with 5 columns: Name, Email, Address, Phone Number, and Favorite Place. The table contains 4 rows of contact information.

localhost:9007/contact

localhost:9007/contact

Apps For quick access, place your bookmarks here on the bookmarks bar. [Import bookmarks now...](#)

Home Contact Add Contact Stock Page

Name	Email	Address	Phone Number	Favorite Place
Dr. Dan	chal0006@umn.edu	383 Shepherd Labs, 100 Union St. SE, Minneapolis, MN 55455	(612)625-4002	<a href="#">Shepherd Labs</a>
Ruofeng Liu	liux4189@umn.edu	Coffman Memorial, Room 2-209, 300 Washington Ave SE, Minneapolis, MN 55455	(612)625-4002	<a href="#">Coffman</a>
Harshit Jain	jain0149@umn.edu	University of Minnesota Sports Field Complex, 600 25th Ave SE, Minneapolis, MN 5	(612)624-9510	<a href="#">Blue Door Pub (Como Ave)</a>
Yang He	he000242@umn.edu	Keller Hall, Room 2-209, 200 Union St. SE, Minneapolis, MN 55455	(612)625-4002	<a href="#">Keller Hall</a>

## Add Contact page

A screenshot of a web browser displaying the 'Add Contact' page. The browser's address bar shows 'localhost:9007/addContact'. The page has a purple navigation bar with links: Home, Contact, Add Contact, and Stock Page. The main content area contains a form with the following fields: Name, Email, Address, Phone Number, Favorite Place, and Enter URL for the Favorite Place. Each field has a corresponding text input box. Below these fields is a blue 'Submit' button.

## Stock page

A screenshot of a web browser displaying the 'Stock' page. The browser's address bar shows 'localhost:9007/stock'. The page has a purple navigation bar with links: Home, Contact, Add Contact, and Stock Page. The main content area features a 'Welcome to Stock Page' message in a light gray box. Below this is a form with a 'Company' label, a dropdown menu showing 'Microsoft', and a 'Get Data' button. At the bottom, there are two tabs: 'Company-MetaData' and 'Stock-Info'.

## Navigation bar

