WEBD6201 – Client-Side Scripting

Lab 4

Express Portfolio Site

Due: Week 13 (Friday April 16, 2021) @ midnight

Value 15%

Express Portfolio Site **Maximum Mark: 100**

Overview: Working alone or with a partner, you will create a Fictitious Portfolio Website using **ExpressJS** and implementing the **EJS templating engine**. Your site must be hosted live on a cloud service such as **Microsoft Azure**, **Heroku,** or **Digital Ocean**. As an option, if working alone, you may use this assignment as a chance to craft your own personal portfolio website.

Your site will also include a secure area that displays a list of business contacts. Users will have to provide a username and password to access this area. The Contact List and your authentication credentials will exist in a MongoDB database. You will be able to perform full CRUD for your contact list so you will need to create all required views.

Instructions:

This Express site must include the pages from your Personal Portfolio 5 pages – your **Home page**, an **About Me** page, a **Projects page**, a **Services page**, and a **Contact Me** page.

Part 1: Site Content (10 Marks)

1. You Site must include the appropriate content for a **Personal Portfolio (10 Marks: Content)**
    a. You must include a **Navigation Bar** or other Navigation scheme that allows the user to view each page of your site. (2 Marks: Content).
    b. You must include a **Custom Logo** for your site, this should be placed in or around the main Navigation bar. The **Custom Logo** can be as simple or artistic as you desire (e.g. you could use a primitive colour-filled shape like a triangle or hexagon with your initials positioned inside). Please do not use a logo that belongs to another company or person. (1 Marks: Content).
    c. Your **Home Page** should include some sort of welcome message and link or button that allows the user to redirect to the About Me Page and / or other pages. I recommend also including some sort of **Mission Statement** (1 Marks: Content).

d. Your **About Me Page** should include the fictitious persons **name**, an their **image** (I recommend a head and shoulders shot that you find or create), and a short paragraph about who the person is. If you are using this assignment to create a personal portfolio site keep this clean and simple as it may be viewed by perspective employers. (1 Mark: Content)
e. Your **Projects Page** should include images and information for at least 3 Projects you wish to highlight. These could be current projects you are working on or past projects you have completed. If creating this for a fictitious person, link to three projects that already exist (you may find interesting projects to highlight on the internet). Include an image for each Project and a short description of your role (if any) and the outcome. (1 Marks: Content).
f. Your **Services Page** should include a short list of services you offer (e.g. general programming, web development, mobile apps, etc.). I recommend including images that make this more appealing to view. (1 Marks: Content).
g. Your **Contact Page** should include the person's contact information in a panel or other construct. Again, you may include your own information if this site is for your portfolio (1 Mark: Content).
h. Your **Contact Page** should include a short interactive form that allows the user to send the owner a message and provide basic contact information (First Name, Last Name, Contact Number, Email Address, Message, etc.). This form does not have to be fully functional (e.g., sending an email to the site admin). However, it should be able to capture the information entered by the user and redirect them back to the Home Page (2 Marks: Content).

Part 2: Express Server Site Structure (10 Marks)

1. Your site will use the new structure created by the **Express Generator**. Your site files will be migrated to work within the **public**, **routes** and **views** folders **(6 Marks: Site Structure):**
   a. Generate your site structure with the Express Generator. **Note**: You must use the **-e** option to ensure that you implement the **EJS templating engine** for Express. You must also install all the modules required. (5 Marks: Site Structure).
   b. Your **JavaScript**, **CSS and Multimedia Asset Files** should be moved to separate folders within the **public** folder. Using the Twitter Bootstrap CSS framework is strongly recommended, though not required. **Note:** the **public** folder is part of the path and does not have to be referenced (2 Marks: Site Structure).
   c. All Your Code (HTML, CSS, JavaScript, jQuery, etc.) is error free (3 Marks: Site Structure).

Part 3: MongoDB Database Server, Mongoose and Passport (10 Marks)

1. Your site's back end will utilize MongoDB as a Database. You will setup a server using **MongoDB Atlas**: https://www.mongodb.com/cloud/atlas (or another online MongoDB server of your choice). You will use the **Mongoose** npm package to connect to the Database and perform basic CRUD operations (create, read, update and delete). You will use the **Passport** npm package (and other related packages) to allow the user to authenticate (Login, Register and Logout) (10 Marks: Site Structure)

a. Create an online Database with **MongoDB Atlas**. (1 Point: Site Structure)
b. Install the **mongoose** npm package and connect to the Database. (1 Point: Site Structure)
c. Create a **Contact Schema** and enable full **CRUD** operations (create, read, update and delete) by modifying the **Router** and **Controller** components of your site (3 Point: Site Structure).
d. Install the **passport** npm package (as well as other related packages). (1 Point: Site Structure)
e. Create a **User Schema** and enable Login, Registration and Logout functionality. (2 Point: Site Structure).
f. Create an **AuthGuard** method that protects secure pages from being accessed by unauthorized or unauthenticated users (2 Point: Site Structure).

Part 4: Server-Side Routing and Content Views (10 Marks)

1. Your Portfolio site will use **ExpressJS** and **NodeJS** and your site's content has been split across various **partial** files. You will integrate these partial files into the pages requested by the user by using the **EJS template engine**. You will use the existing index.js **View template.** You will create any required routes by using the Express middleware framework:
    a. Each page in the web site will be split into its own partial file and reside in the **Views/partials** folder (3 Marks: Functionality).
    b. An Express Route must exist for each page of your site. **Note**: You will need to use the **router.get(path, callback(req, res, next))** method structure with a **res.render(view, locals)** method call to render each view (4 Marks: Functionality).
    c. Your **page logic** should reside in a controller file named **index.js** that will reside in the controllers folder (3 Marks: Functionality).

Part 5: Site Security – Login/Register Views (12 Marks)

1. In addition to your site's main **Nav Bar** and **page template components**, your **Login and Register Views** should include the following components:
    a. **Login View.** A Form that includes a **username field**, a **password field** and a **Login Button.** (3 Marks: Functionality).
    b. **Register View.** A Form that includes **First Name**, **Last Name**, **Email Address**, **username** and **password** fields and a **Register Button**. Successful registration will create a new user. (3 Marks: Functionality).
    c. The application database will contain a **`user` collection.** The user Schema will include the user's **Display Name** (made up of the user's **First Name** and **Last Name**), **username**, **password,** and **Email Address**. The `username` and `password` credentials will be used for authentication (3 Marks: Functionality).
    d. If the user is authenticated, he will be taken to the **Business Contacts List View**, otherwise the user will be **redirected** back to the Login View if the username and/or password entered is incorrect (2 Marks: Functionality).

e. If a user attempts to access the **secure area** of your site, they should be redirected back to the Login View (1 Marks: Functionality).

Part 6: Secure Area: Contact List View (13 Marks)

1. Your **Business Contacts List View** should include the following components:
    a. A list of contacts should appear on this page. The connection to the database you created will display all the contact data in a table. The table will only include The **Contact Name**, **Contact Number** and **Email Address** (3 Marks: Functionality).
    b. Include an **Edit Button** and a **Delete Button** at the end of each Row of the Table. (2 Marks: Functionality).
    c. The Delete Button **removes** the contact from the database (2 Marks: Functionality).
    d. The **Edit Button** transfers the Contact Information (**Contact Name**, **Contact Number** and **Email Address**) from the related row in the **Contact List** to the **Update View.** When the user links to the **Update View** using the **Edit Button**, the header element will display "Edit Contact". The **Edit Button** in the Update View will display "Edit", and the **Delete Button** will be shown. (3 Marks: Functionality).
    e. Include an **Add Button** that will link to the **Update View**. When the user links to the **Update View** using the **Add Button**, the View's header element will be modified to display "Add New Contact". The **Edit Button** in the Update View will be modified to "Add", and the **Delete Button** will be hidden. (3 Marks: Functionality).

Part 7: Secure Area: Update View (10 Marks)

1. The **Update View** will include the following components:
    a. A **Form** will allow the user to update any contact details (**Name**, **Contact Number**, **Email Address**, etc.) (3 Marks: Functionality).
    b. The Form will include an **Edit Button** that will allow the user to send the user information entered in the form to the database. (3 Marks: Functionality).
    c. The Form will include a **Delete Button** that will remove the user from the database and return the user to the **Contact List View** (2 Marks: Functionality).
    d. The Form will include a **Cancel Button** that will return the user to the **Contact List View** without making any changes. (2 Marks: Functionality).

Part 8: Documentation, Version Control, Cloud Hosting and Video Demo

1. Include **Internal Documentation** for your site **(5 Marks: Internal Documentation):**
    a. Ensure you include a **comment header** for your **CSS and JavaScript files** that indicate: the **File name**, **Student's Name**, **StudentID, and Date,** (2 Marks: Internal Documentation).

   b. Ensure you include a **section headers** for all of your **HTML structure, CSS style sections,** and any **JavaScript functions** (1 Marks: Internal Documentation)
   c. Ensure all your code uses **contextual variable names** that help make the files human-readable (1 Marks: Internal Documentation).
   d. Ensure you include **inline comments** that describe your GUI Design and Functionality. **Note:** Please avoid "over-commenting" (1 Marks: Internal Documentation)
2. **Version Control.** Share your files on GitHub to demonstrate Version Control Best Practices **(5 Marks: Version Control).**
   a. Create an appropriately named GitHub Repository that you and your partner will use for this lab. Only one repository is required (1 Mark: Version Control)
   b. Your repository must include your code and be well structured (2 Marks: Version Control).
   c. Your repository must include commits that demonstrate the project being updated at different stages of development – each time a major change is implemented (2 Marks: Version Control).

3. **Cloud Hosting**. Host your site live on a **Cloud Provider** of your Choice (Heroku recommended) **(5 Marks: Cloud Hosting)**
   a. Your site's images should be visible on your Live site (1 Mark: Cloud Hosting)
   b. Your CSS Layout should function appropriately on your Live Site (2 Marks: Cloud Hosting)
   c. Your Script files should function appropriately on your Live Site (2 Marks: Cloud Hosting)
4. **Demo Video.** Create a Short Video presentation on YouTube or another streaming provider. You must include a short PowerPoint (or Google Slides) Slide Deck that includes a single slide to start your video **(10 Marks: Video)**
   a. The first (and only) Slide of your Slide Deck must include current images of you and your partner (no avatars allowed) that are displayed appropriately on the page. You must also include your Full Names, Student IDs, the Course Code, Course Name, and your Assignment information. (2 Marks: video)
   b. You or your partner will demonstrate your program's functionality. You must show your program working properly in the console (2 Marks: Video)
   c. You or your partner will describe the code in your files that drives the functionality of your program (2 Marks Video).
   d. Sound for your Video must at an appropriate level so that your voices may be clearly heard, and your screen resolution should be set so that your program's code and console details are clearly visible (2 Marks: Video).
   e. Your Short Video should run no more than 5 minutes (2 Marks: Video).

SUBMITTING YOUR WORK

Your submission should include:

1. A zip archive of your project uploaded to DCConnect
2. A working link to your complete project files on GitHub
3. A working link to your Live Site hosted on a Cloud Provider of your Choice (Heroku recommended)

4. A working link to your demo video posted on YouTube or another streaming provider.

Evaluation Criteria

Feature	Description	Marks
Site Content	Appropriate Content is added to each page of your portfolio site	10
Site Structure	You have generated your site with the Express Generator. Your Project adheres to the site structure described (including Assets, Content and Script folders). You have connected to MongoDB Atlas (or another online MongoDB database). You have installed the mongoose npm package to enable full CRUD operations. You have installed the passport npm package (and other related packages) to enable authentication features including user login, registration and logout functionality.	20
Functionality	Site deliverables are me and site functions are met. No errors, including submission of user inputs.	45
Internal Documentation	File header present, including site & student name & description. Functions and classes include headers describing functionality & scope. Inline comments and descriptive variable names included.	5
Version Control	GitHub commit history demonstrating regular updates.	5
Cloud Deployment	Ensure your Site is hosted live on a Cloud provider of your choice (Heroku recommended). All images, CSS and Scripts should function appropriately.	5
Video Presentation	Your short video must demonstrate your site and describe your code. Your audio must be at an appropriate level and your screen must be clearly seen.	10
Total		**100**

This assignment is weighted **15%** of your total mark for this course.

Late submissions:

- 20% deducted for each day late.

External code (e.g. from the internet or other sources) can be used for student submissions within the following parameters:

1. The code source (i.e. where you got the code and who wrote it) must be cited in your internal documentation.
2. It encompasses a maximum of 10% of your code (any more will be considered cheating).
3. You must understand any code you use and include documentation (comments) around the code that explains its function.
4. You must get written approval from me via email.