**Report by Group 24**

Section 1.0: Prior to Installation

To ensure the following sections facilitate a smooth process, the user has to do three critical steps. These steps make sure the necessary installations are downloaded directly into the project directory.

1. Find and copy the directory path where the project is located
2. Open the command prompt
3. Enter cmd along with the directory path on the same line

Section 1.1: Installation Procedure

In order to successfully run the project, the user should first install all the necessary modules and libraries. These installations are required for the whole code to work including updating the database, having server-side authentication, and ultimately setting up sessions to access the admin pages. With the open command prompt, the user should enter each command below separately.

* npm install express --save
* npm install body-parser multer --save
* npm install mongodb --save
* npm install passport express-session --save
* npm install passport-local --save
* npm install connect-flash --save
* npm install locutus --save

Section 1.2: Connecting to the Local Server

After the user has installed the following packages, they are now able to connect to the server. So, to finally connect to the server, the user should activate the node application by entering “node” along with the NodeJS filename “myNode.js” on the same line in the command prompt. More frankly, the command that should be typed onto the command prompt is “node myNode.js”.

Section 1.3: Opening the Project Website

The user should now be able to see the project website by typing “http://localhost:8080/” on the address field of their respective website browser. Upon submission, the link will redirect the user to the main page of the project website.

Section 1.4: Navigating the Project Website

By using the menu bar at the top of the website, users can freely navigate through the categories of the website. There is also a responsive navigation bar where the website recognizes the small screen and puts all the menu buttons in the top right corner.

Section 1.5: Logging in to Access Admin Dashboard

To edit the contents of each category in the website, the user has to login with their admin credentials; respectively, the username “admin” and the password “grouptwentyfour123”. To access the login page, the user can either click the “Julie Thorpe” logo located on the top left of the navigation bar, or they can manually type “http://localhost:8080/login” on the address field of their browser. Both ways will successfully redirect the user to the login form.

Section 1.6: Using Admin Privileges

After the server has verified the user’s credentials to be the admin, the server will then create a continuous active session for them to use and redirect the admin from the login page to the admin dashboard. From the dashboard, the admin can access the different admin pages, which the admin can use to essentially update the contents of the website. Also shown in the dashboard are the main pages of the website, and the logout button which terminates the admin’s session. As a side note, since the server recognizes the user to be the admin, the “Julie Thorpe” logo found on the website’s menu bar will now redirect the admin to the admin dashboard instead of the login page.

Section 1.7: Terminating Admin’s Session

There are two ways for the admin to end their session. The first way is by clicking the logout button located on the top right of the admin dashboard. As mentioned, the admin can access their admin dashboard by either clicking the “Julie Thorpe” logo on the website’s navigation bar, or manually typing out “http://localhost:8080/admin” on the address field of their browser. The second way, although not recommended, is ultimately terminating the connection to the server. A user can do that by opening the command prompt, that is currently running the connection to the local server and pressing the shift key and letter “c” key together.

Design Explanation

One of the main reasons why we decided to code our program in NodeJS’s express rather than PHP was that overall, it was less complicated to implement the required installations we needed in NodeJS. Specifically, the packages of passport and express-session. With these two packages, implementing sessions, and user authentication in our code was straightforward and easier. As per our research method, we mostly used and read articles to find the best way to implement sessions and user authentication. After numerous researches, we found express-session and passport to be appropriate; while express-session helps the server create active sessions for our users, passport helps determine if a user is logged in, and thus be able to use a session. Additionally, we used MongoDB to help update our database as we found it was easier to organize data that way. Also, not to mention, we have more experience working with MongoDB in our labs and lectures. Thus, we did not use another application to manage our databases.

Technical Challenges Faced

One of the technical challenges that we faced while troubleshooting had to do with adding content as opposed to only updating the content. Since the layout of our website is complicated, we could not think of a way of adding content via the admin page without manually creating a template for the newly added content. Therefore, our solution was to limit the admin to update respective content by only using the existing positions and template of the layout. This solution has also brought a problem of its own, we also had a problem with the positioning of the updated content. At first, we just had a switch statement to arrange the positions of the content. However, we came to realize that the previous update would be removed when the new update was shown; due to the code just looping through the database once. Therefore, we tried a for loop as a solution so that the previous update will not be replaced, and instead the code will loop through each document in the collection; it successfully assigned each information to its respective angular scope variable based on its position number.

# **References**

Codingflag. (2020, September). Create a Simple Popup / Modal using HTML, CSS & JavaScript. https://codingflag.blogspot.com/2020/09/create-a-simple-popup-modal-using-html-css-and-javascript.html.

CodingNepal. (2020, September 29). Animated Login & Signup Form Design using HTML

CSS & JavaScript. Retrieved November 18, 2020, from https://www.codingnepalweb.com

CodingNepal. (2020, September 29). Responsive Personal Portfolio Website using HTML CSS & JavaScript. Retrieved October 20, 2020, from https://www.codingnepalweb.coml

CSS Buttons. (n.d.). Retrieved October 20, 2020, from https://www.w3schools.com

CSS Navigation Bar. (n.d.). Retrieved October 20, 2020, from https://www.w3schools.com

Garcia, V. (n.d.). Create Responsive Google Maps on Any Website. Retrieved October 20, 2020, from https://www.ostraining.com

How TO - Filter Elements. (n.d.). Retrieved October 20, 2020, from https://www.w3schools.com

How TO - Image Overlay Title. (n.d.). Retrieved October 20, 2020, from https://www.w3schools.com/howto/howto\_css\_image\_overlay\_title.asp

How TO - Timeline. (n.d.). Retrieved October 20, 2020, from https://www.w3schools.com/howto/howto\_css\_timeline.asp

Hanson, J. (2019). Documentation: Authenticate. Passport.js. http://www.passportjs.org/docs/authenticate/.

Hanson, J. (2019). Documentation: Configure. Passport.js. http://www.passportjs.org/docs/configure/.

Hanson, J. (2019). Documentation: Username & Password. Passport.js. http://www.passportjs.org/docs/username-password/.

Hanson, J. (2019). jaredhanson/passport. GitHub. https://github.com/jaredhanson/passport.

Locutus. (2020, November 19). PHP's htmlspecialchars in JavaScript. https://locutus.io/php/strings/htmlspecialchars/.

npm. (2012). connect-flash. npm. https://www.npmjs.com/package/connect-flash.

npm. (2020, April). express-session. npm. https://www.npmjs.com/package/express-session.

WebDevSimplified. (2019, July 9). WebDevSimplified/Nodejs-Passport-Login. GitHub. https://github.com/WebDevSimplified/Nodejs-Passport-Login.