**PROJ3**

**PART C - Security**

admin Passwords use one-way encryption

Source IP whitelist to restrict access to admin panel

Text

Description automatically generatedText

Description automatically generated

Test application over HTTPS connection, (screenshot evidence)

<https://rachelpac.com/build/>

Graphical user interface, application, Teams

Description automatically generatedGraphical user interface, application

Description automatically generated

**PART D - Metacognition**

**Comment on each of the 3rd party frameworks used, why was it chosen**

React was chosen because of the way it renders the user interface by reacting to user interactions as the user interface for this app was heavily conditional on user interaction React was the best front-end framework.

**What other technologies did you investigate in order to settle on a path**

Vue was also investigated as a possible option but it was found React was the best choice due to the reasons listed above.

**Describe the rules by which your authentication restricts access. Comment in code.**

Authentication restricts access by first checking to see if a user is logged in (checks to see if secession variables are set), if they are not logged in, they will be presented with a message asking them to log to view the restricted pages.

Certain user accounts have also been restricted access to certain actions, in the same way as the code above, session variables are checked to see the type of user that is logged in. This session information is sent to the interface and stored in local storage. Although the session variables are checked again when the authenticated user sends a request back to ensure the information from local storage has not been tampered with.

When the user enters the correct username and password the are authenticated, session variables are set, and they are granted access to restricted pages.

**Describe why you chose this particular encryption technology**

The PHP password\_hash function was used to encrypt user passwords in registration.

password\_hash() creates a new password hash using a strong one-way hashing algorithm. password\_hash() is compatible with crypt(). Therefore, password hashes created by crypt() can be used with password\_hash().

PASSWORD\_DEFAULT – Uses the bcrypt algorithm.

The used algorithm, cost and salt are returned as part of the hash. Therefore, all information that's needed to verify the hash is included in it. This allows the password\_verify() function to verify the hash without needing separate storage for the salt or algorithm information.