Task 1:

Screenshot of WordPress dashboard visited from web server browser.

Graphical user interface, text, application

Description automatically generated

Screenshot of the wp\_users table in database server

A screenshot of a computer

Description automatically generated with medium confidence

Task 2:

Searching “National University of Singapore” on Bing

Graphical user interface, application

Description automatically generated

Searching “National University of Singapore” on Google

Graphical user interface, text, application, Teams

Description automatically generated

Google prevents using its search engine within an iframe and requires the search to be on a new window. This could be because Google is sending an X-Frame-Options: SAMEORIGIN response header requiring page in the frame to have the same origin as all ancestor frames.

Task 3:

In the page created, the code for task 2 is added into the wordpress page in the <body> tag, then nested in the <main> tag with the id “site-content”. It is then nested in the <div> class “post-inner thin”, then “entry-content” and lastly in the <p> tag. The iframe, which is also nested in this <p> tag, will then load “https://nus-cs5331.github.io/search.html” into the iframe DOM. This includes the script in search.html which is nested in the <body> and then <script> tag.

After clicking on the “Search” button in the page, the Javascript code first find the iframe element with the id “frame1”, then with the contentWindow property, will call the postMessage method. Before returning the postMessage method, the Javascript code will check which radio button, either google or bing, has been selected using getElementById, then call JSON.stringify method to store the value associated with bing or google search engine and the text in the search term also using getElementById method and searching for the id “text1”. This JSON data will then be stored as the “message” argument in the postMessage method. “\*” would then be the value in “targetOrigin” argument. The postMessage method will then send a post message to https://nus-cs5331.github.io/search.html which is listening for a message sent. When a message is received and contains the correct JSON data, it will send a http GET request to either search engine with the search content.