Web Technologies Assignment: Final

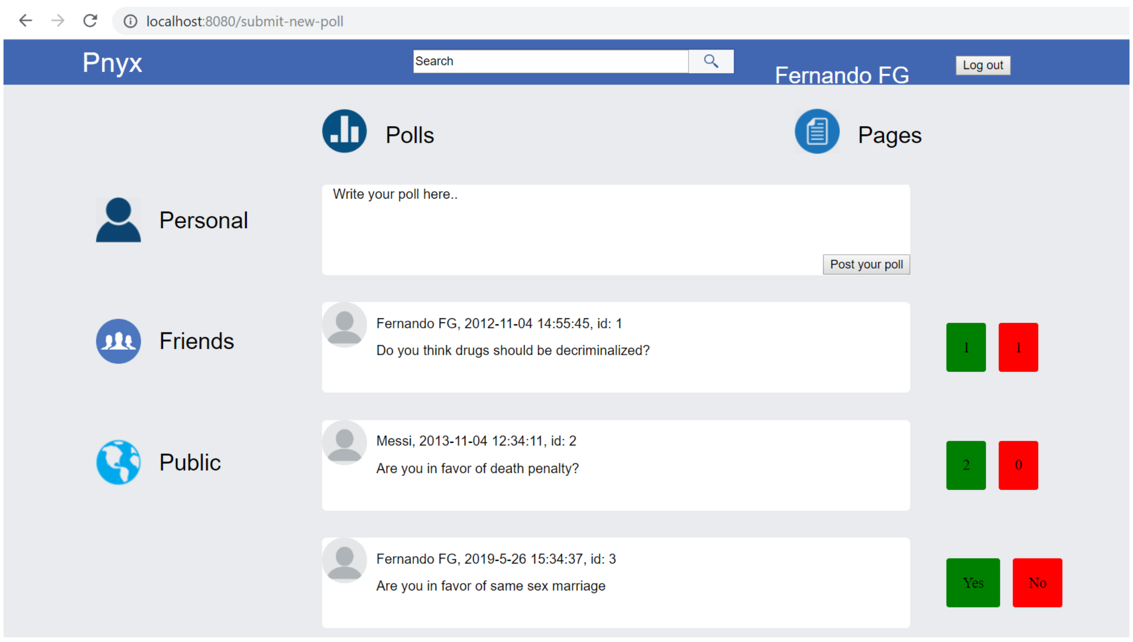
Fernando Fortes Granado[1]
[1] *yt18331@bristol.ac.uk, Candidate 71820 (Study abroad – Electrical Engineering)*

May 26, 2019

1 The purpose of the website

The original objective of the website was to be a large space for the users to share opinions and information. However, duo to time constraints and the fact that I was alone, only one of the desired functionalities was implemented.

Said that, the website submitted is a space for the users to create polls for other users to answer, and to answers other user's polls. For example, someone could make a poll "Do you think drugs should be decriminalized?", and the other users would be able to vote either yes or no. After voting, the user would see how many votes both options accumulated so far.



Fernando FG, 2012-11-04 14:55:45, id: 1
Do you think drugs should be decriminalized?

Messi, 2013-11-04 12:34:11, id: 2
Are you in favor of death penalty?

Fernando FG, 2019-5-26 15:34:37, id: 3
Are you in favor of same sex marriage

Figure 1: screenshot of the page where users can create and vote in polls.

2 The details of the implementation

The website contains two pages: (1) a login/signup page, illustrated in figure 2, and (2) the page where the users can create and vote in polls, illustrated in figure 1.

2.1 The database

The database diagram is represented below:

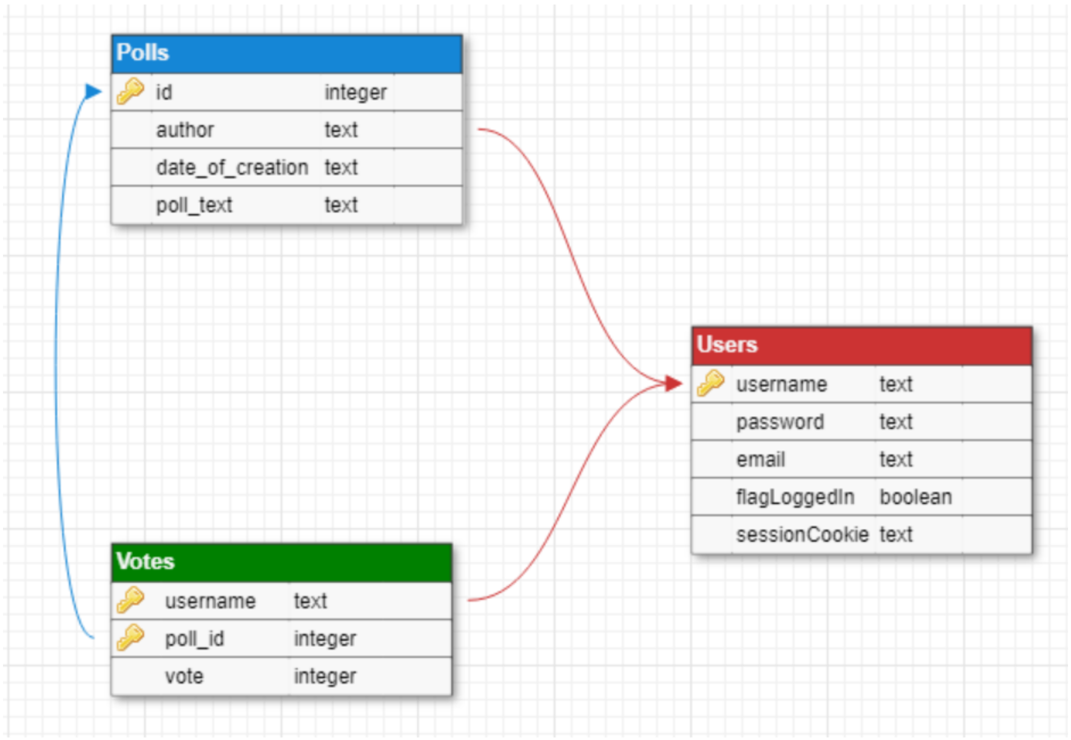


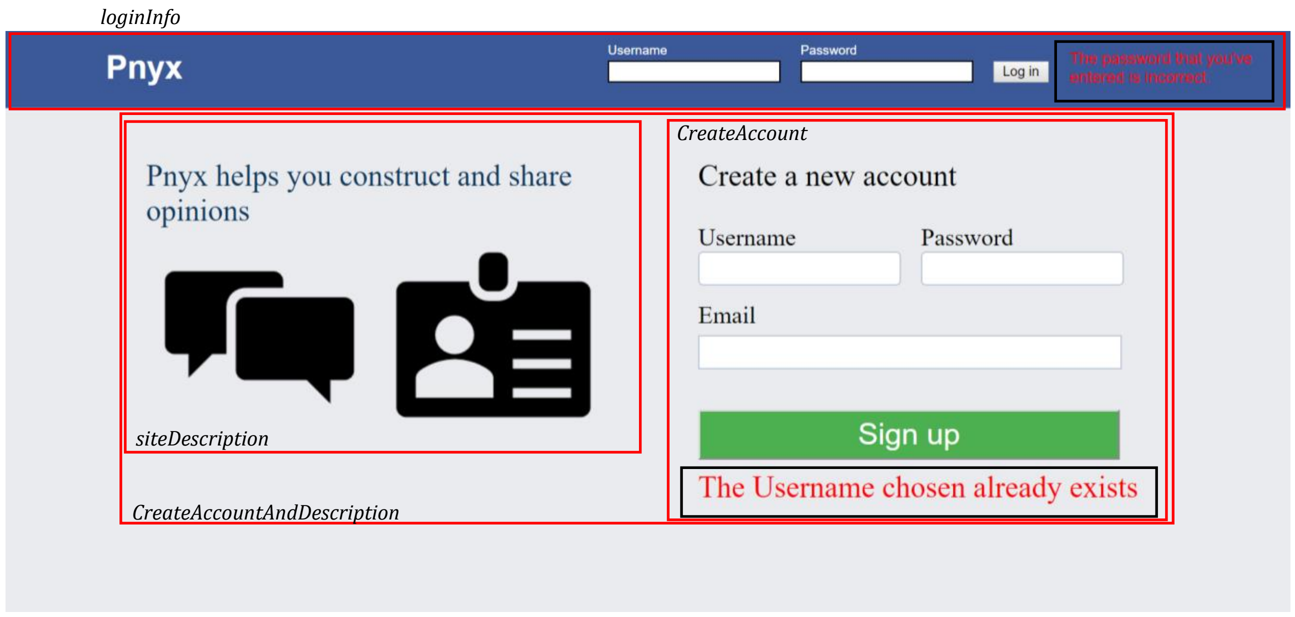
Figure 2: database diagram.

The table Users contains the information of the users created so far. The table Polls contains information about the polls created so far. The table Votes contains all the votes, “1” meaning “yes”, “0” meaning “no”.

2.2 HTML/ejs templates and CSS

To create the pages, it was used ejs templates and CSS written from scratch. The images below have boxes showing the most important elements of the page. The red boxes show the static elements, while the black boxes show the placeholders of the template. However, the developer’s tool is the easiest way look at all the details.

The login/signup page has 2 main elements: “loginInfo” and “CreateAccountAndDescription”. The “loginInfo” has a blue background, the Logo element and the login form inside of it. It also has a paragraph element (see black box), where the server can write an error message before delivering the page to the client.



The password that you've entered is incorrect

Pnyx helps you construct and share opinions

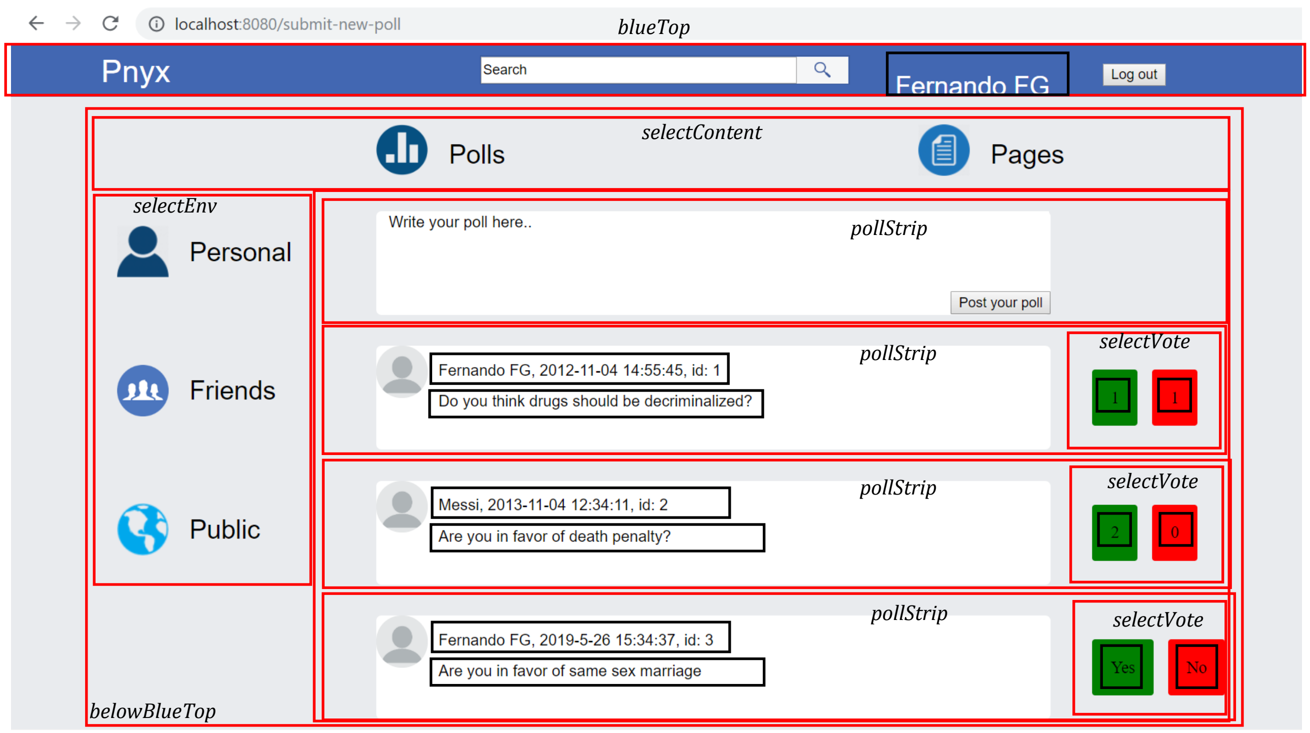
Create a new account

The Username chosen already exists

Figure 3: representation of the main elements of the login/signup page index.ejs. The red boxes show static elements and the black boxes show placeholders in the template.

The "CreateAccountAndDescription" element has two children elements: the "siteDescription" and the "CreateAccount" elements. The "CreateAccount" element contains the signup form and an error message placeholder. The server can write an error message in the placeholder before delivering the page, if necessary.

Now, let's analyse the main elements of polls.ejs template.



Fernando FG, 2012-11-04 14:55:45, id: 1
Do you think drugs should be decriminalized?

Messi, 2013-11-04 12:34:11, id: 2
Are you in favor of death penalty?

Fernando FG, 2019-5-26 15:34:37, id: 3
Are you in favor of same sex marriage

Figure 4: representation of the main elements of the polls page polls.ejs. The red boxes show static elements and the black boxes show placeholders in the template.

Similarly to index.ejs, the poll.ejs contains a blue strip element "blueTop", and, under it, and element "belowBlueTop". The important point here is that each element "pollStrip" has placeholders for author, date, id of the poll and the poll itself, as well as the number of votes associated to each poll. When the server deliver this page to the user, it writes the appropriate values in each of these placeholders.

The functionalities referenced in "selectEnv" and "selectContext" were not implemented, as well as the "Search button".

In terms of **animation**, the page poll.ejs uses javascript to change image opacity, add/remove underline and bold letters and change text color. This is done by calling javascript functions, which get elements by id and change their properties.

2.3 Sessions

Session is a key point in this website. When implementing it, the objective was to do it from scratch, to have a better understanding of the most basic underlying mechanisms. However, consequently, many simplifications had to be made to fit the deadline, and the technology implemented turned out to be insecure in many ways.

Said that, when a user is logged in an “Fernando FG”, the client has a “sessionCookie” set with some value. In the table Users in the database, the property “sessionCookie” associated with the username “Fernando FG” holds the same value. On the other hand, when the user is logged out, the property “sessionCookie” is NULL. This way, to check if a given user is logged in, the server makes a query in the Users table searching for a username with the same cookie value as the client.

When the user tries to access any page of the website, the server verifies if the client is logged in. If it is logged in, the server sends the poll.ejs page. Otherwise, the server sends the index.ejs page, where the user can login/signup.

2.4 Server handles login

The login/signup page contains two HTML forms: one to submit a login, if the user already has an account, and another to submit the signup information otherwise. Both forms are submitted via POST, to “/submit-login” and “/submit-signUp” respectively.

When the user submits the login form, the server makes a query in the Users table in the database to find if there is a pair (username, password) in the database. If so, the server generates a random 20byte string to be a session cookie and sets the cookie in the client. The server also updates the database, setting the sessionCookie property of the corresponding Username in the Users table. After that, it redirects to the polls page (logged in). If (username, password) is not found, the server sends the login/signup page with an error message (“The password that you've entered is incorrect.”).

Similarly, when the user submits the signup form, the server makes a query in the Users table in the database to check if the Username already exists. If not, the server generates a random 20byte string to be a session cookie and sets the cookie in the client. The server updates the database, creating a new entry in the Users table and setting the sessionCookie property of the new Username. After that, it redirects to the polls page (logged in). Otherwise, if the Username already exists, the server sends the login/signup page with an error message (“The Username chosen already exists”).

The cookie that keeps users logged in is the called “sessionCookie”.

2.5 Handling new poll

When a new poll is submitted, the server compares the client’s “sessionCookie” with the database (property “sessionCookie” in the Users table) to know which user made the new poll. After that, the server inserts a new entry in the Polls table (author, date of creation, poll text), and render the poll.ejs page again.

If someone not logged in tries to create a poll by sending post data to the server, it will not be accepted.

2.6 Sending and handling a new vote

When the user clicks the vote button, the javascript function “submitVote(pollId, vote)” will send the vote as POST.

In the same way as with new polls, the server reads the client’s cookies to know which user is logged in. Then, it inserts a new entry in the Votes table (username, poll_id, vote). After updating the database, the server makes a query to count the number of “yes” and “no” votes of that poll and sends this information to the client as Json.

The client receives the Json element and uses it to dynamically update the page, writing the number of “yes” and “no” associated to the poll.

2.7 Sending and handling request for more polls

When the server renders polls.ejs, the number of polls is limited to 20. However, if there are more than 20 polls in the database, the user can request for more polls by clicking the button at the bottom of the page. When the user clicks the button, a request is sent to the server at "/ask-more-polls", giving as a "parameter" the number of polls rendered so far.

The server queries the database (in a nested query with Joins and Group by) and returns a Json element containing the extra polls' information (author, date, poll_text, number of "yes" votes, number of "no" votes). What makes the query reasonably complex is that, if the user has already voted in the poll, we want to send the number of "yes" or "no" of the respective poll, but if the user has not voted the poll, we do not want to show that information.

The client receives the json element and uses it to dynamically create new "pollStrip" elements, holding the information given by the server.

2.8 Server handles logout

When the user clicks the logout button, the server updates the database by writing NULL on the "sessionCookie" property of the corresponding username. After that, it redirects to the index.ejs page.

2.9 Using Gimp

Following a tutorial, I used Gimp to change the background of a random image found on the internet (a butterfly) from black to white. To do it, I had to use the contiguous selection tool from Gimp to extract the butterfly from the background. Then, I used the eraser to correct the silhouette of the extracted butterfly. Finally, I inserted a white filter to be the background of the butterfly. Below are some screenshots of the process.

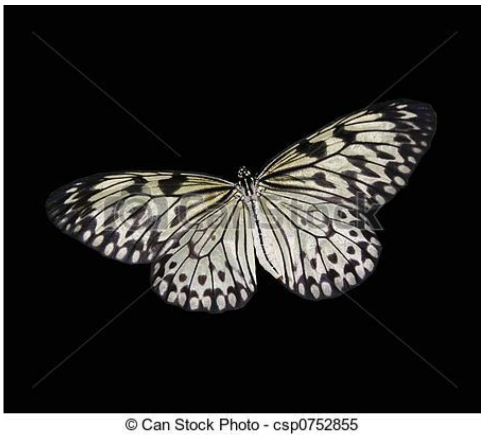


Figure 5: original image. Source https://www.canstockphoto.com/butterfly-on-black-0752855.html

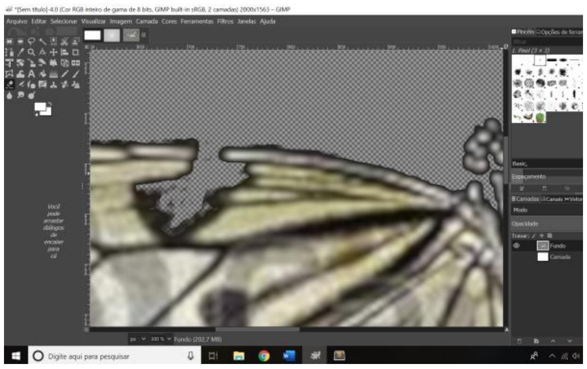


Figure 6: using eraser to correct the silhouette of the extracted butterfly.

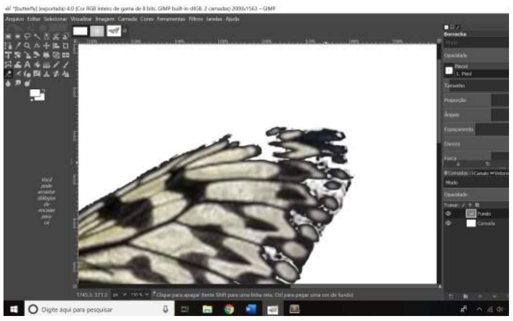


Figure 7: adding the white filter under the butterfly and correcting the silhouette with the eraser.

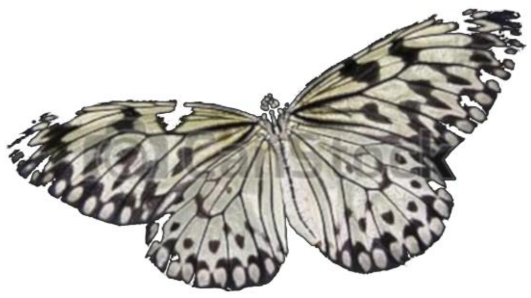
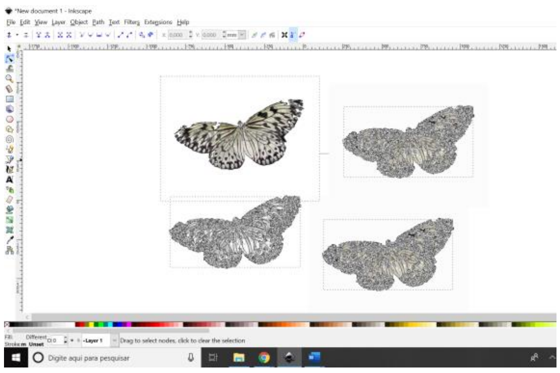


Figure 8: final image.

2.10 Using Inkscape

Inkscape was used to automatically vectorize the new butterfly created with Gimp. And to draw a horse from scratch. The images below show the process of creating both images.


*New document 1 - Inkscape
File Edit View Layer Object Path Text Filters Extensions Help
Drag to select nodes, click to clear the selection
Digite aqui para pesquisar

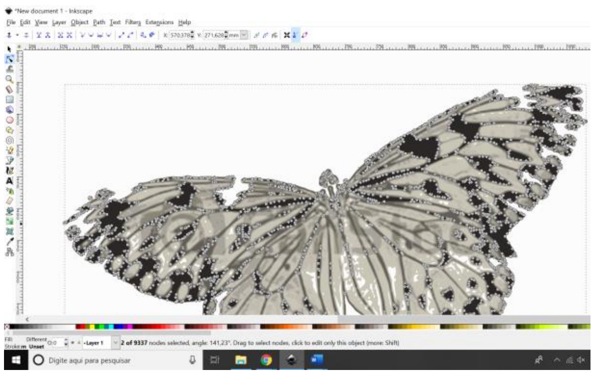
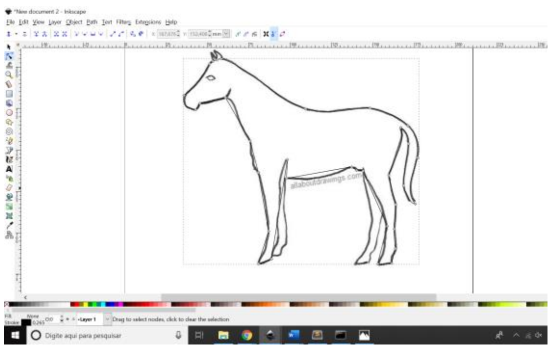

*New document 1 - Inkscape
File Edit View Layer Object Path Text Filters Extensions Help
Digite aqui para pesquisar

Figure 10: vectorized butterfly

Figure 9: vectorized butterfly


*New document 2 - Inkscape
File Edit View Layer Object Path Text Filters Extensions Help
Drag to select nodes, click to clear the selection
Digite aqui para pesquisar

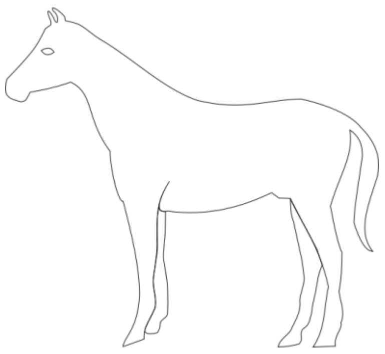


Figure 11: drawing a horse using a sketch

Figure 11: final drawing of a horse

3 Mark Estimation

```
<ul>
<li>A for HTML</li>
<li>A for CSS</li>
<li>C for JS</li>
<li>C for PNG</li>
<li>C for SVG</li>
<li>B for Server</li>
<li>B for Database</li>
<li>B for Dynamic pages</li>
<li>X for Depth (out of 40)</li>
</ul>
```