The most difficult part of this program was the writing the queries. Many of the queries involved variables, and nearly all of them required some single quotes in the final result. This led to many confusingly long statements. Furthermore, if these statements did not work, there would not be an error message. I often had to guess which statement did not work. To solve this problem, I began testing the sql query statements in phpMyAdmin before adding them to my php code.

The second problem I had was how to add a name to the list if it wasn’t already there, and if it was there, how to increment the vote counter. I decided to query for the votes of the name that had been submitted regardless of whether it already existed and then check if the number of votes were greater than 0. This would tell me whether the name already existed, so that I could take the proper action to either create it and initialize and vote counter, or update the number of votes.

One thing that I thought was interesting about how I wrote my code that was not entirely intentional, is that both genders can have the same name but different votes. This keeps the program from causing errors, even if the same name is used twice.

Finally, I decided to complete the ajax portion of the code. I thought this code would be difficult, until I saw the example of a similar thing on the w3schools website. I was able to use much of the same code, which made the problem much simpler. I did, however, need to add two new files (ajax.js and gethint.php) to get it to correctly work. I also had to look up a way for it to work as a dropdown menu. I was able to use the datalist tag in html to create the dropdown. Then, I had the ajax code run whenever the textbox changed. This did cause a problem that because the datalist and the textbox were connected, it was constantly updating, and making the datalist unusable. I had to add an if statement to the JS code that said if the testbox value was the same as the last time, don’t run the code.