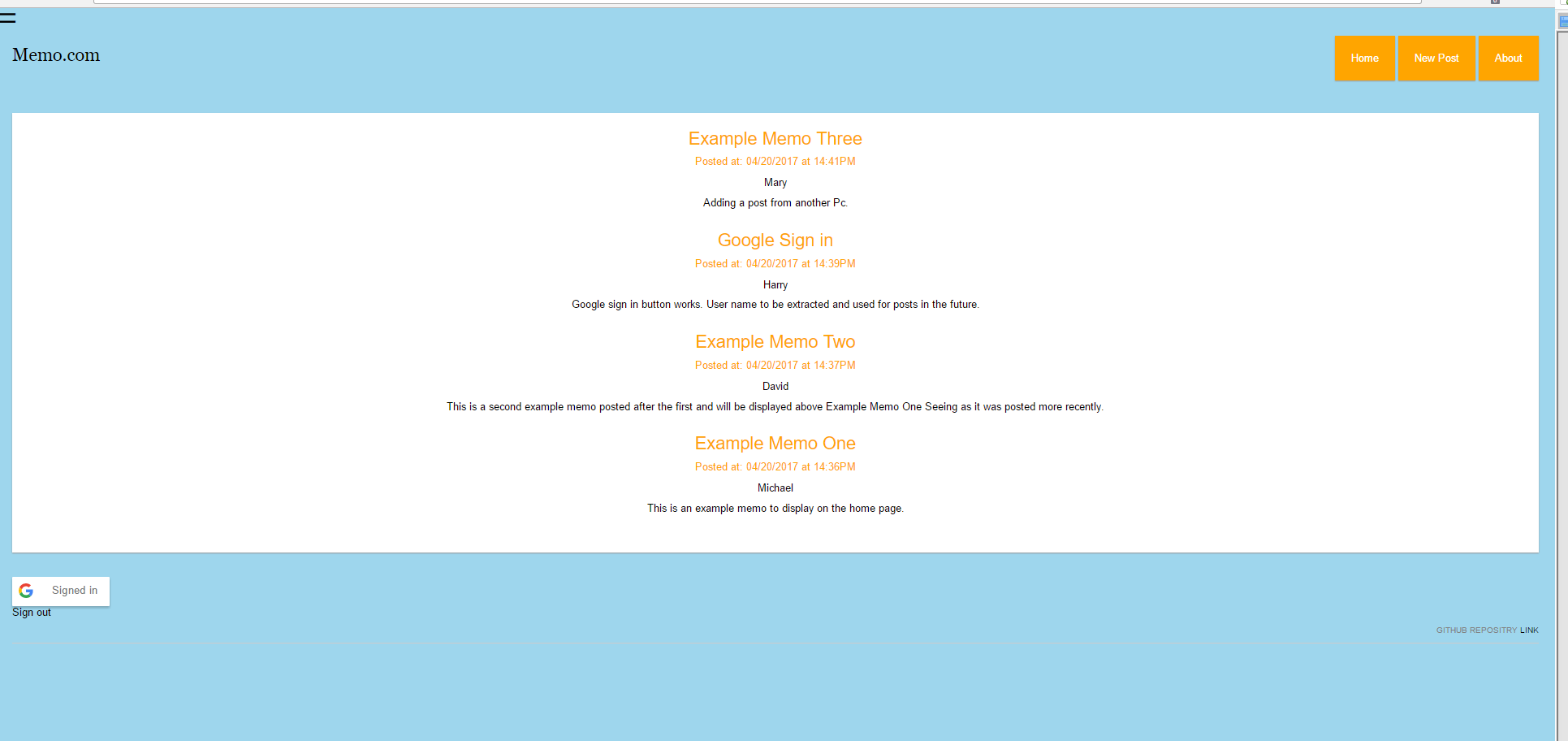
**Design Document**

# Introduction

This project is a dynamic web app based on a memo/post system. I decided to do this because I had previously worked on a group project on a basic dynamic web app In the module Data Representation and querying by Ian McLoughlin at Gmit and felt I could make something somewhat better or slightly more technical than what we had submitted.

The App itself is available online at <http://g00325746.pythonanywhere.com/> . For live testing and or viewing. There is also a live demo of it on Youtube at <https://youtu.be/7EHCJf1hPz0> . Which demonstrates some core functionality.

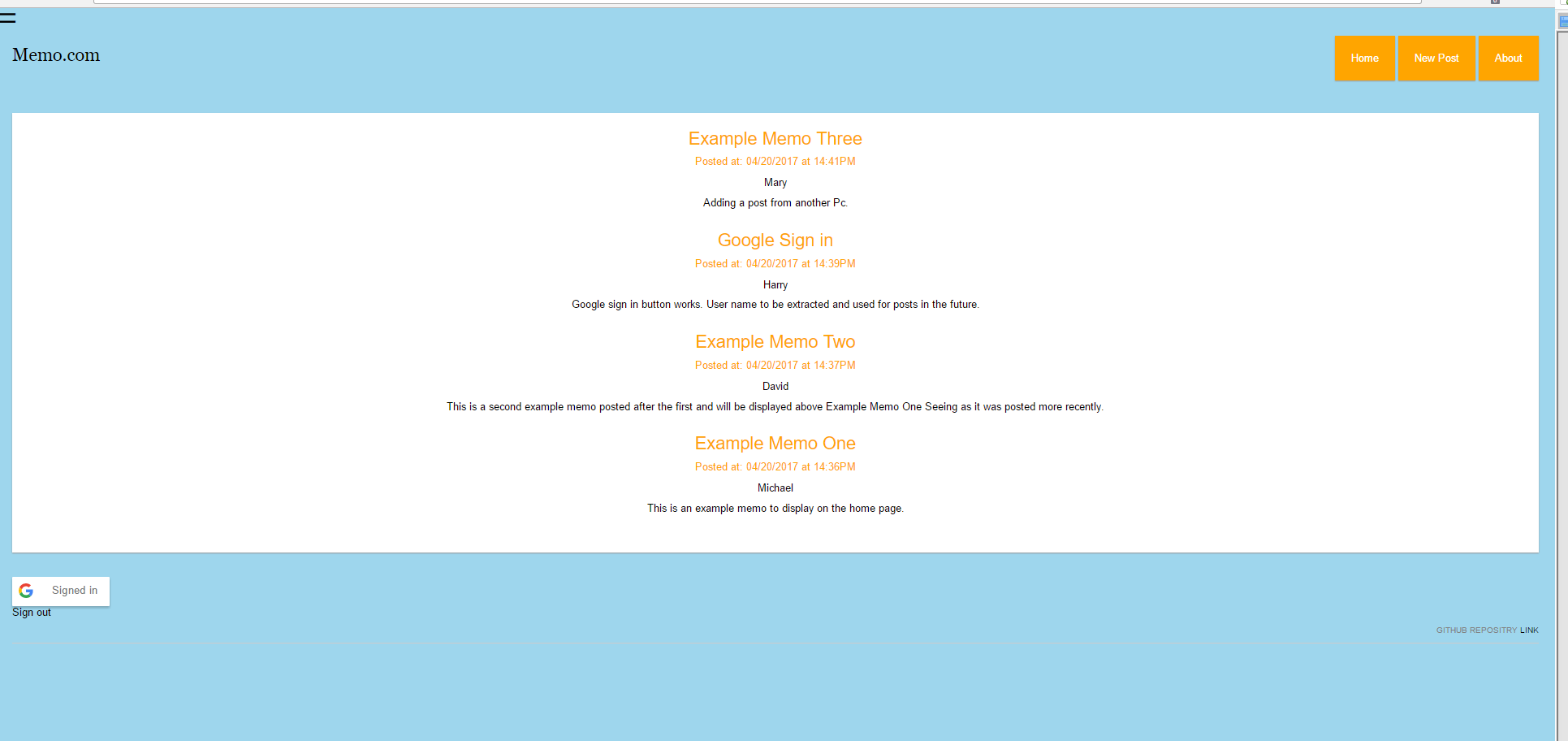
The function of the app is to allow users to post quick notices for others to view it is ordered by publish time and shows the most recent post to it from anywhere whenever the home page is brought up again. With more work, Implementation-wise it could be setup for separate companies or groups to only allow them to view and add to the notices of their own company this could be validated through the google sign in. Available on the page but not yet fully functional the sign in is available but there is no system in place currently to direct users to their sections.

# Layout

The layout of the web app contains three major templates the Home.html, the NewPost.html and the About.html . With a consistent navigation scheme across all three pages and footer.

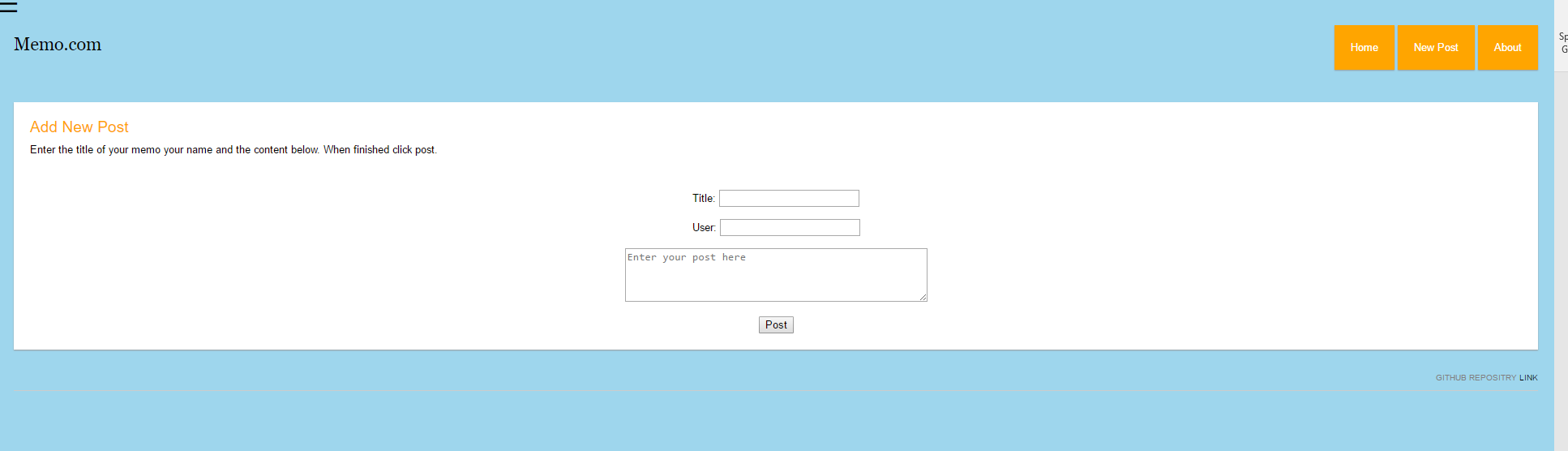
Home

When the user opens the page, they will be presented with the home page on this page there will be a google sign in button. This button when pressed will present a popup screen allowing the user to enter their google sign in credentials. Also, a sign out button for if the user wishes to sign out. A Top navigation and a side navigation (a side navigation was added for easier navigation for mobile users and top navigation is more convenient for mouse ) option to the other pages as-well as a view at the current posts they will have to come back to the home page to see if something was posted while they were at the home page. The page will automatically adjust in length to cater to a large amount of older posts. Newest will remain at the top and older further down.



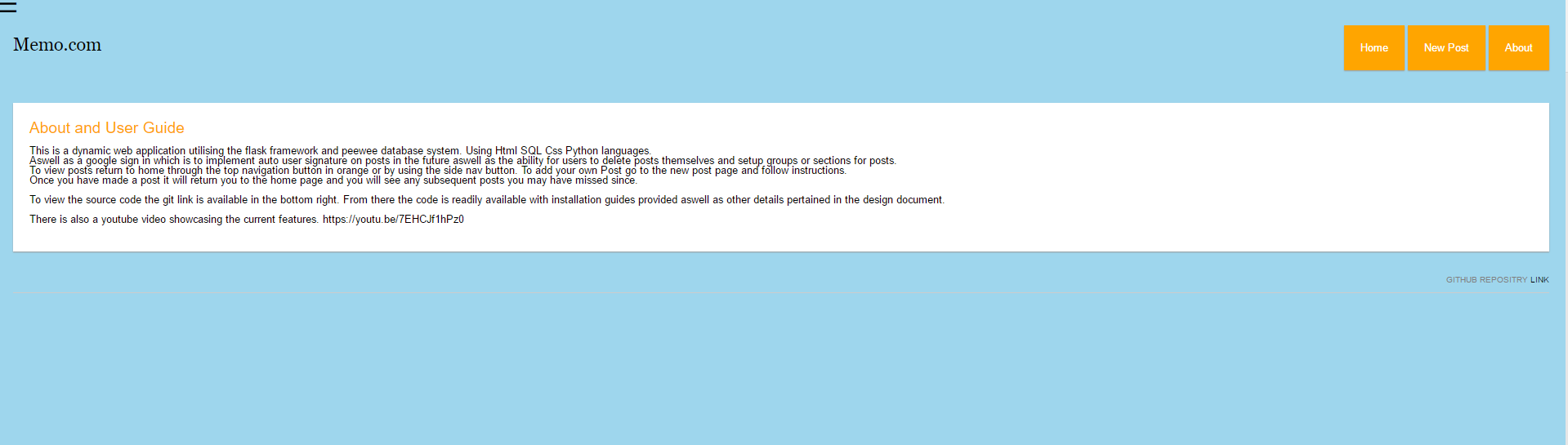
New Post

Under the new post section, they will be presented with a form to make a new post. This includes the same navigation layout and footer layout consistent throughout the project alongside a forum. This forum contains the Title field the user field and the content field which will all be saved to the database as well as being displayed on the home page. It includes the post button which deals with those tasks.



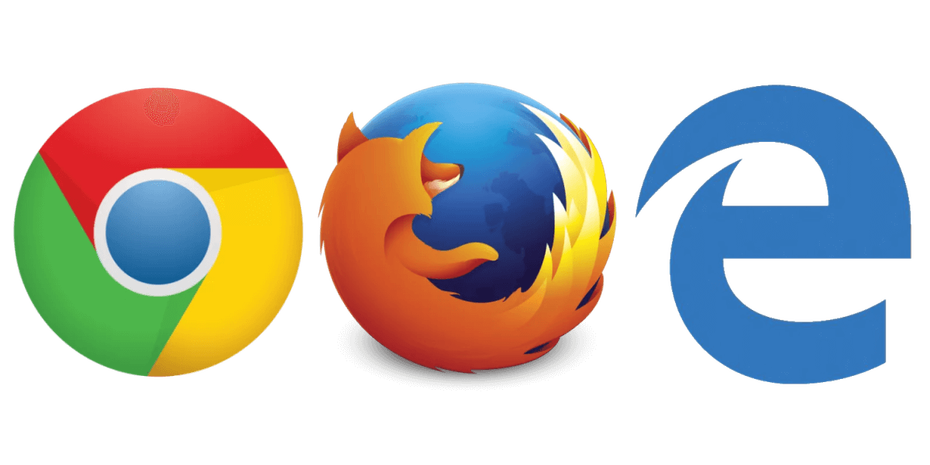
About

Finally, the about page just gives some information on site like what languages were used or processes in the development of the site and a simple User Guide to interact with the site .The page also uses the same navigation and footer layout as the previous two.



# System Requirements

To view the live app no requirements are needed specifically it has been tested with some different browsers .Browsers the application will work on include Google Chrome, Microsoft Edge and Firefox. The site used for hosting provides python functionality and from their console I was able to install all the necessary frameworks and other imports needed. These included flask and the directories it contains as well as peewee. On the other hand if you are running this locally.



To run locally you will need to download the project folder from the github link available here or through site <https://github.com/MichaelGeraghty/3rd-Year-Project.git>

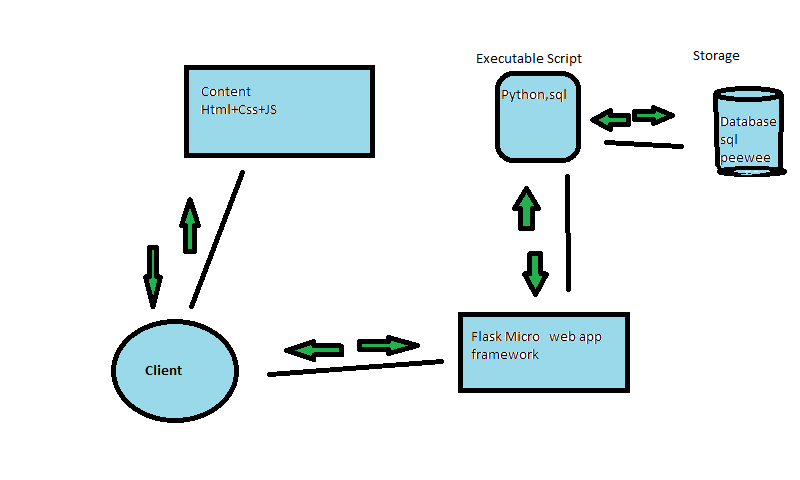
Have Python 3.6 Installed then from the console in the folder location install flask and the few other imports its mentioned in the git repository but things like “pip install Flask”, “pip install peewee” etc the console will notify you of any smaller imports if they were not picked up. I recommend the latest version of google chrome when running on local host.

# Technology Used and Why

Seeing as it is a dynamic web app The main technologies used here were python and the flask micro web framework specifically. I used these because I had some prior experience with them. I also find python to be quite straightforward at times as-well the ease at which it is to make a flask web app. I used peewee for the database peewee is a small ORM(Object relational mapping) tool it is built with mysql and sqlite in mind and it works really well with a flask app. Also the size of the database needed is not extremely large and this is more than sufficient. It also is all programmed into the app and models files , its structured simply using a class like structure. The database files can then be edited added to removed in any program that can interact with them eg. MySql or even using something as basic as a text editor .

I then used pythonanywhere.com for hosting because github and others would only host static pages . Also python anywhere allowed me to install anything extra needed for the site through their console system.

# Architecture of the Solution



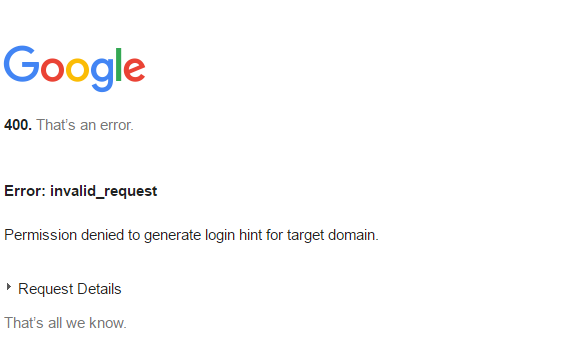
This diagram is in reference to a typical web app architecture used for blogs. Content is fed to the user as or if needed the python scripts are being fed as called, flask as a web framework links the executable script and use of the database including the posting or retrieving of data to what the user see’s , which is the html, css and the javascript.

# Limitations

At the moment there is a google sign in but have not implemented the auto complete of filling in the users name when they post they have to enter a name themselves. There is no deletion of a post currently also but edits and deletions can be made by an admin with access to the database. The site is not under constant need of uptime so the site can have these edits or deletions made with no bearing on the site . Currently if the site had consistent traffic throughout it the means at which an edit or deletion would be made would be the denying access to the site for a minute while the database is edited and the newly formed file could be uploaded. This is not ideal yet this would have had a higher priority in the development if in a real site as well as the users ability to display their email on the post when they post.

# Known Bugs

If run locally the google sign in returns an error this is because you have to add specific instances with google that use a google generated id these can be setup easily with urls but some slight complications with localhost. The bug is currently being overlooked because it does not affect the implementation of the google sign in through the site and is not needed in the running of the app locally. Another Bug is the button in the top left for navigation not being displayed properly on certain mobile browsers. This however does not interfere with the function of the button and there is still an element present to be pressed.



# Features

This is a multi-user system . Multiple users may actively post to the site and be able to read other users posts in the same session. Users can navigate using buttons on top navigation bar which is easier for mouse users or if they have their hand placed there. If users are viewing this from a handheld device there is a button in the top left for navigation also. There is one data entry screen on the site, google provide a data entry screen for a google sign in. The purpose of the database currently is to store and retrieve the posts it contains an id which is a primary key field, date which is a date time field, title ,text and content which are all text fields.

# Recommendations for Future Development

Memo pages will be setup under different groups who have their own unique id’s and passwords to join groups where they can leave posts that they can only see. Also, allowing an admin who created the group to self-adjudicate over posts to be edited or deleted.

When a user signs in their name associated with their email will be used. Currently that info is accessible in console but I am yet to implement it fully.

The option to delete a post through the site itself. Other than the database being edited and put in place of the current database.

Adding a feature similar to Twitter and Facebook where at the top of the last post it will notify you of newer posts that have been submitted to the home page.

# Conclusions

In conclusion at the start of this project I set out to improve my ability with developing dynamic web apps, to implement an external login like google, to use a database and to have it hosted online with full functionality. Knowing that if I had spent a little more time earlier on it I could have implemented the current recommendations for future development.