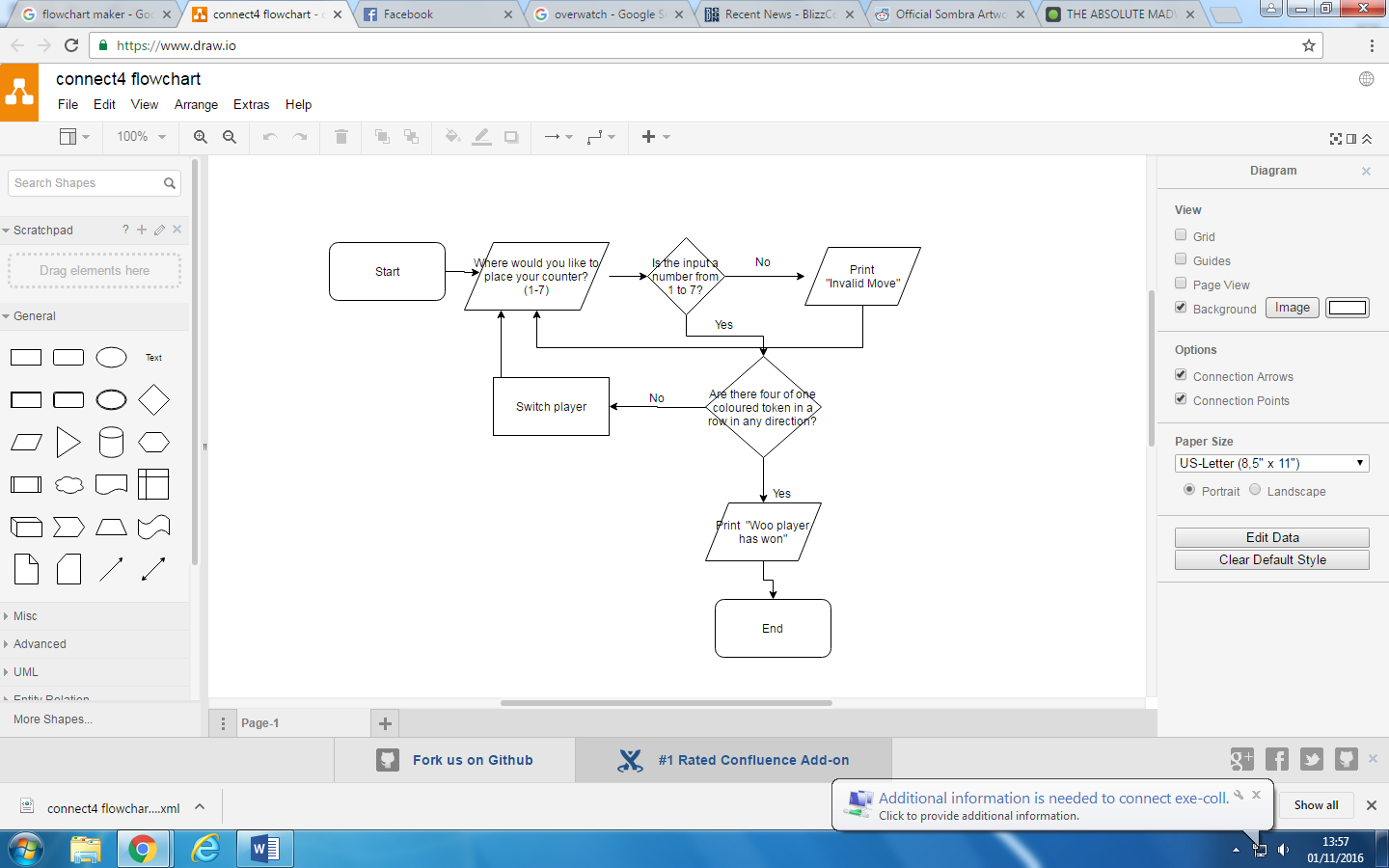
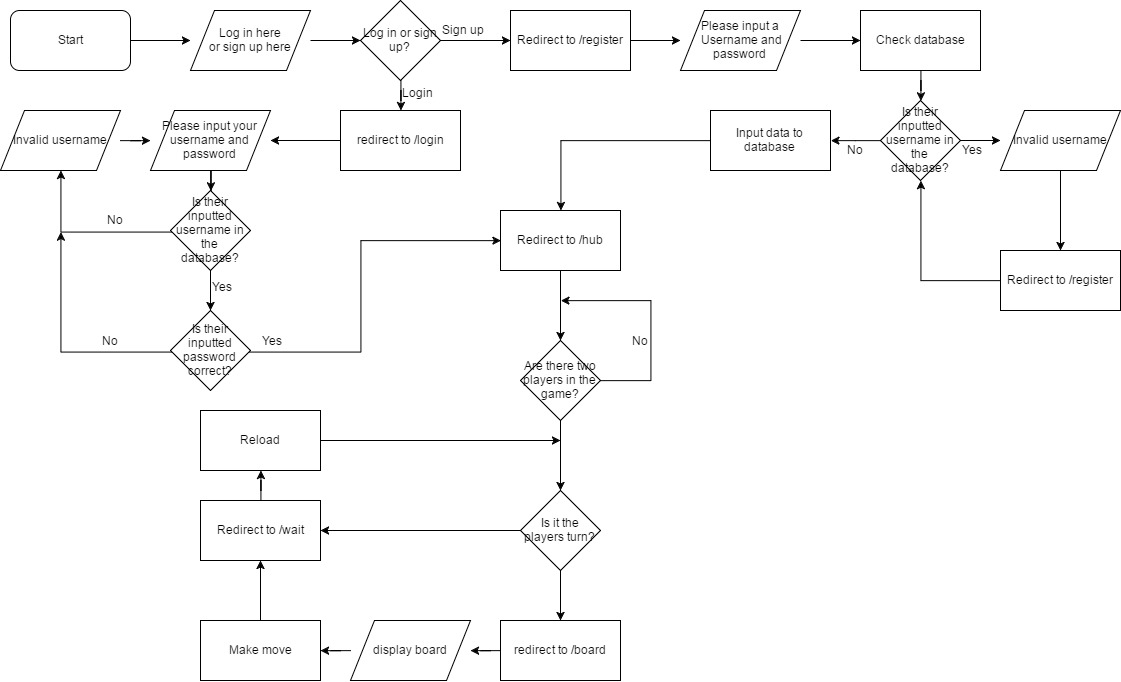
Documented design

This is a basic representation of how my FourInARow class will work.



A basic representation of how the networking for the website will work:

Connect Four backend:

In my program for Connect Four I used a one-dimensional array. Although I could have used a two-dimensional array, I chose to use a list so that I could more easily check adjacent spaces. I also added in some empty items to my list so that there would be no problem with winning by looping around the board, since there will also be an empty space next to each edge.

Object oriented:

FourInARow:

I chose to program the connect four part of the project as a Class. This was so that I could use my program as an object, and be able to have multiple games being played simultaneously on the network. This was done alongside my BoardStore Class, so that I would be able to store the data about each FourInARow board, without affecting any other ones. The main purpose of this Class is to get the game to run. It sets up a list, as mentioned above, and any time the makemove function is called, the board is checked and updated. ***(I can probably go more in depth here about it all)***

BoardStore:  
I wanted my program not to be limited to being able to play only one game at once. Because of this I needed a way of storing different FourInARowBoards’. I could have just made a global dictionary, and stored the data for each board in that, but I wanted to avoid global variables, since they can be affected from anywhere. Because of this I created a Class to store FourInARowBoards, which I have named BoardStore. Whenever you join a game, the main program checks the BoardStore class to see if there’s a game with any space and adds you to it if there is one. If there isn’t, then a FourInARow board is created and stored in the BoardStore dictionary, and the player is added to that board, assigned to red. The colour and index in the dictionary of the board is returned to the main program. The board can then be accessed at any time, and changed.

Flask:

I have researched into the web framework Flask. I have chosen this framework as it can be versatile but is very easy to use, and gives you a large amount of control. It also has a simple and easy to understand format, and the use of function decorators in it is very effective. Furthermore, it is a great learning experience, and quickly teaches you a lot about networking.

Hardware specification:

To play connect four the user will need a computer, a keyboard, a monitor and a router. As the game will be multi-player, you will want a friend, who also has access to the previously mentioned hardware

Database Structure:  
Since my project is able to play with other people online and store your win/loss rate, I have implemented a database into my project. This is so people can log on with a username and password, and compare skill level with their friends. My table for people’s login data is as follows: ID, username, password(serial, text, text) (\*Must look up how to actually write this). To create my database I used PostgreSQL, and creating it in the terminal of my computer.

File structure and organisation:

To keep my project in order I have been…..

Tools to help with my project:

For my project I have been using the operating system Debian. This was mainly to facilitate with downloading pip, and using pip to install the web framework Flask. Because I have been working on my computer from multiple different places/ computers, it would been unnecessarily time consuming to save and email my files to myself each time I did even a little bit of work on it. This is why I chose to use GitHub, so that I can access my files from any location at any time; it also acts as a way to back up my files in case my computer fails. I have used the integrated development environment Pycharm. This is because it is very useful for organising files, and it has integrated a method to easily push and pull files to and from GitHub without opening the app or going on the web page. This has made working with GitHub a lot easier, and has reduced the time by a surprising amount. Pycharm, as I have found throughout this project, is also very convenient: it is easy to access files, including other essential non python files, so I can access all my project files from one place; your program saves automatically; and when writing Classes it automatically inputs “self” into the necessary places, as well as automatically closing brackets and quotation marks, which has often stopped me from making syntax errors.

To do:

The first paragraph is a bit rambly and short on some information, so it will have to be developed. The flowchart is simplistic and is just a basic idea of what to do, but I wasn’t 100% sure in how much detail I was supposed to do it in. Will have to add more sections and lots more detail. I think the tense is wrong, i’m not sure whether to do it past, future or present (most likely future).