COMP2811 User Interfaces
Coursework 3: Design Document

Team members:
Domantas Dilys - Andrew Parkes - Ciaran Brennan -
Jonathan Alderson - Matthew Cumber

1. Description of the Application

The application that has been developed for this project is designed for the user to explore a Git repository. Upon loading the program, the user is greeted with a page that informs them whether or not there is a git repository in the current working directory. The layout of the application is styled as a folder, with the navigation located at the top of the page as clickable tabs, with the active tab highlighted to the user. The brand logo located at the far left of the navigation menu also functions as a button that redirects the user back to the main page.

The design was chosen to be minimalist and intuitive for users with any experience level, trying not to overwhelm the user with options whilst displaying the functionality of the application clearly. The layout of each tab is set out with the same design principles in mind. The colour scheme, font styling and sizing persists throughout the application which helps to streamline the user experience. The main functionality of each tab is displayed centrally in a smaller sub-page style, with consistent margins and spacing, and any user input performed in standard text input forms. Any error, warning or confirmation messages will be communicated to the user via pop up dialog boxes, which are styled appropriately, relative to the message displayed.

2. Statistics Tab

Author : Jonathan Alderson
Username : sc17j3a

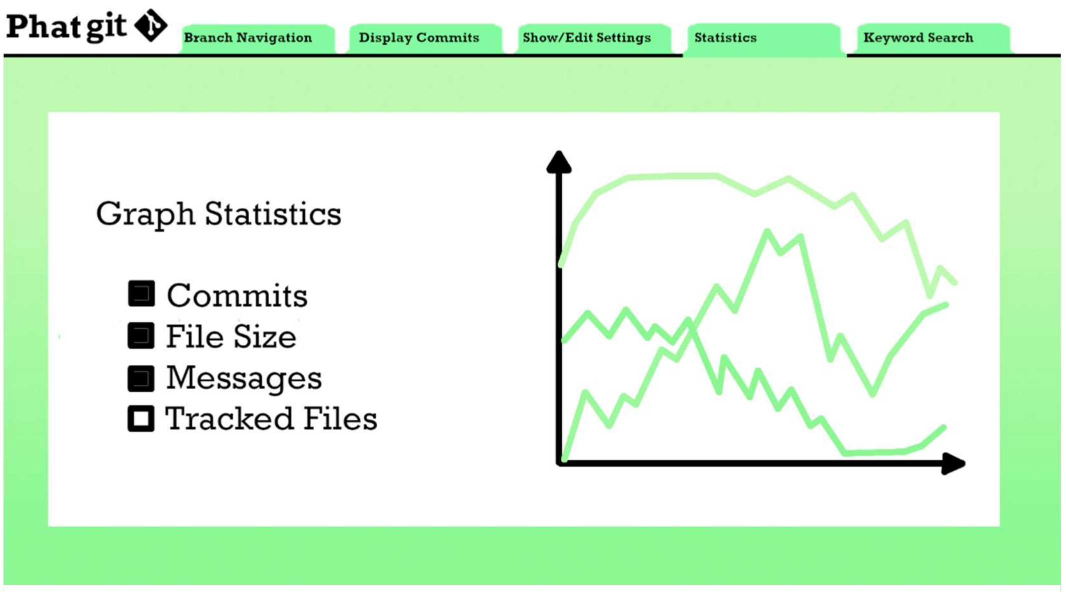
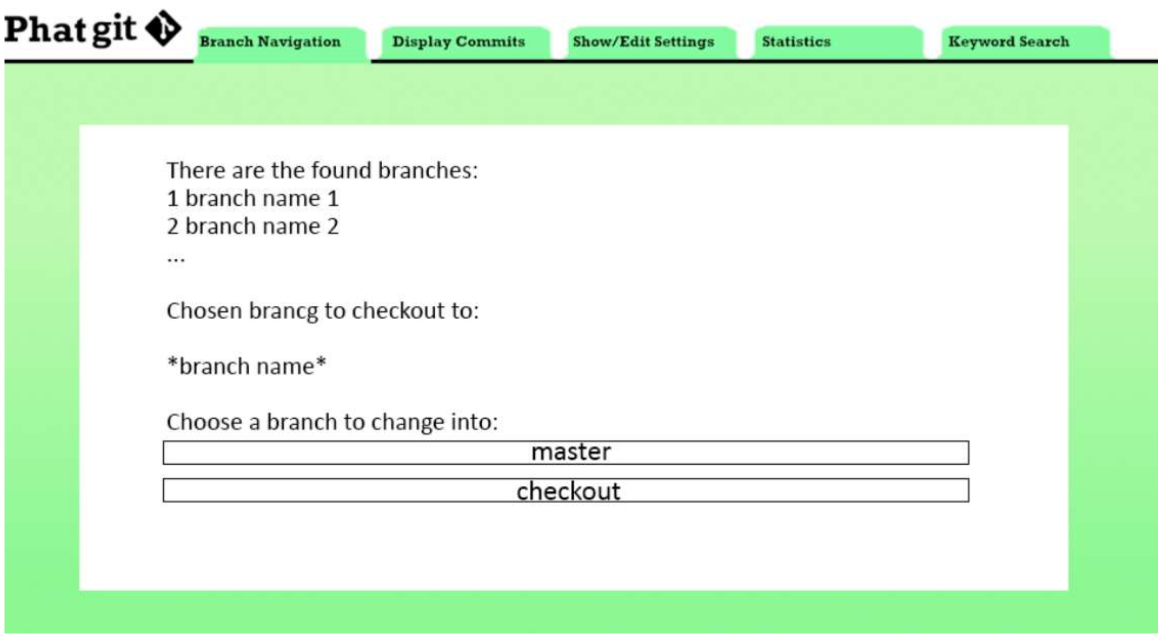


Figure 1 Sketch of the Statistics Tab

Statistics Representation

Upon loading the statistics page (*Figure 1*), a blank graph is shown and the user is given options to select the data displayed by clicking on the checkbox option list shown next to the graph. When an option is selected, the graph updates showing the correct information for the currently selected repository. Multiple options can be chosen at once if the user chooses to do so, and selecting more than one option has the effect of layering the data on the graph. The displayed data can be removed again by deselecting the relevant check box. The options include the amount of commits completed, the file size of the commit, the length of the commit message and the number of tracked files.

3. Branch Navigation Tab
   Author : Matthew Cumber
   Username : ll16m23c



There are the found branches:
1 branch name 1
2 branch name 2
...

Chosen brancg to checkout to:

branch name

Choose a branch to change into:

Figure 2 Prototype design of Branch Navigation tab

Branch Listing

This tab will start by listing all the branches (Figure 2). These will be formatted into a numbered list with a heading such as “List of found branches”. This list will have no other functionality other than to provide quick information to the user of the branches found in the current repository.

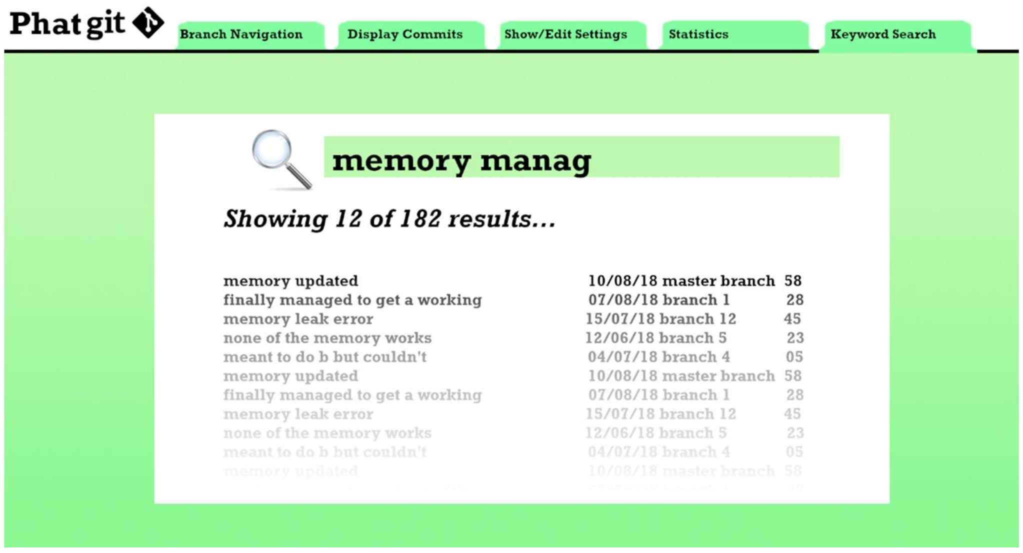
Branch Checkout

A user will be able to checkout a branch from this tab. a drop down menu will be used to select a branch. The drop down menu will contain the list shown previously at the top of the tab. This eliminates error in user input. This design is also easy to use and is standard practice for many user interfaces so users will know how to use a drop down menu. There should be some visual feedback to the user on which option in the menu is chosen. This is done by a label that has the text corresponding to the selected branch in the menu. That way it is clear to a user which branch is sleeted at any given time. As for example when first switching to the tab the drop down menu will display a branch but this isn’t a selected branch by the user. As such the label will tell the user to select a branch before checking out using the menu below.

To checkout a branch a user will press a button. This is because buttons typically perform actions, in this case branch checkout. With a clear button name such as “checkout” users will have no dispute over the functionality of this button. The button will be at the end of the tab as this is the last piece of functionality for this tab.

4. Search Commits Tab

Author : Ciaran Brennan
Username : sc17cjb



memory manag

Showing 12 of 182 results...

memory updated	10/08/18 master branch	58
finally managed to get a working	07/08/18 branch 1	28
memory leak error	15/07/18 branch 12	45
none of the memory works	12/06/18 branch 5	23
meant to do b but couldn't	04/07/18 branch 4	05
memory updated	10/08/18 master branch	58
finally managed to get a working	07/08/18 branch 1	28
memory leak error	15/07/18 branch 12	45
none of the memory works	12/06/18 branch 5	23
meant to do b but couldn't	04/07/18 branch 4	05
memory updated	10/08/18 master branch	58

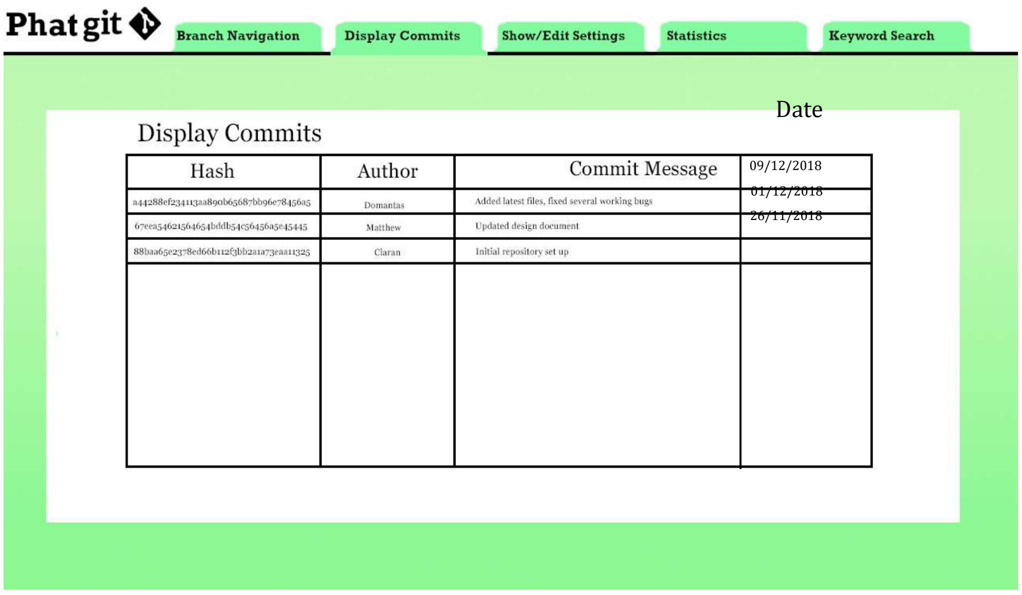
Figure 3 Sketch of the Search Commits Tab

Search Functionality

The keyword search tab (Figure 3) displays a window with a search field at the top. This allows the user to find a particular commit of the repository. As the user enters text the search field, the results appear below. The user can then view the results, and quickly view which user has made the commit.
The action for the user is indicated by the title of the button 'Search' as it indicated what action is performed on-click and what should be inputted in the text field.
The received feedback is commits that match the words in the text field.

5. Display Commits Tab

Author :	Domantas Dilys
Username :	sc17dd



Hash	Author	Commit Message	Date
a44288ef234113aa890b65687bb96e78456a5	Domantas	Added latest files, fixed several working bugs	09/12/2018
67eea54621564654bddb54c56456a5e45445	Matthew	Updated design document	01/12/2018
88baa65e2378ed66b112f3bb2a1a73eaa11325	Ciaran	Initial repository set up	26/11/2018

Figure 4 Sketch of the Display Commits Tab

Displaying Repository Commits

The display commits page (Figure 4) shows all the commits in the current repository in a table with four columns: the commit's hash, that uniquely represents the commit, the author – person, who pushed it to the repository, the commit message, which gives context to the changes being made and the date the commit was pushed. The commits are sorted by date, since it is usually the case that the most recent changes need to be seen first.

The tab's functionality should be evident to the user from its name and the initial table setup – since there are no buttons and the table loads instantly, there will be no confusion about the purpose or usage.

6. Edit Configuration Tab
Author : Andrew Parkes
Username : ll16ap

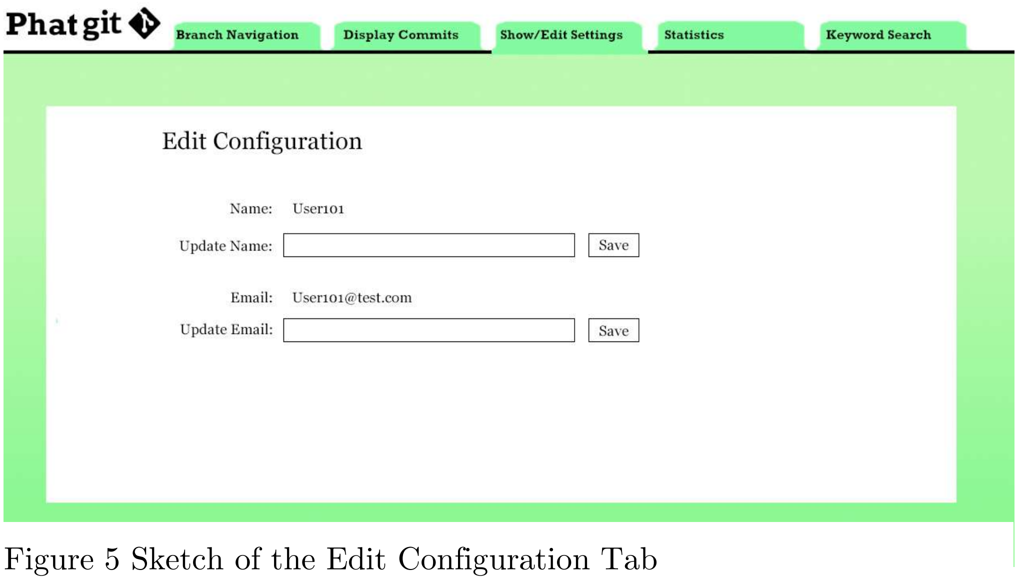


Figure 5 Sketch of the Edit Configuration Tab

User Configuration

The edit configuration page (Figure 5) allows the user to view and change the details stored in the config file of the git repository. When the page is loaded, the user is presented with the name and email stored in the config file of the currently checked out repository. Under each of the fields, there is a text box that takes user input for updating the relevant information. The user enters their details then when the save button is pressed, the data is updated and the user is alerted with a dialog box showing a message confirming that the changes have been made, or if there was any error whilst saving. The layout of the page is clear and follows the minimalist styling of the application, whilst providing clear instructions to the user on the actions available to them.