**Draft for Text Editor Report**

Motivation

Build something that I can use, that can be used to share code interactively. Can be used to manipulate files, to edit code, to run code through terminal.

GitHub

Setup:

* Git ignores
* Branches

Feature branches:

* Layout
* File management

Electron

* Reasoning
* Node js and modules
* IPC
* Package.json

Layout

The starting point for building my text editor. This includes basic HTML (sorting out divs, class and id names) and CSS.

A screen shot of a computer

Description automatically generatedSidebar on the left will be for file management, and more functionality tbc. The top bar will be for tabs. The white area will be text area for coding. This gives an extremely basic layout of the text editor, in order for me to add and test functionality. After functionality has been added, the layout and design will be improved.

File Management

I have set up inter-process communication (ipc) modules so that the renderer and main class can communicate. In this case I will use this to send files from the backend to the front end (expand on this).

Open > open document

Create new > new tab with text

Save > save to current file, save as if not currently saved

Save as > save as new file.

Meeting with Ian – 05/02/24

* Chat implementation
* Commit messages
* Plan, build on interim report
* Look back at intro if a refresh is needed

To do – 25/03/24

* Tabs
  + Close tab
  + Check if tab is already open
  + Don’t make new tab if tab is already open – just open current tab
  + Switch between tabs
  + When one tab is closed, switch to different tab
* Files
  + Make “new file” create an unsaved tab that can be saved as a new file
* Folders

Tabs

Folders

To implement folders, I needed to use recursion to create a tree of files and subfolders to send from main to renderer. To display the contents of the folder I used recursion again to unpack each file/subfolder.

*The use of depth shows me what file is in which folder for the css*. No longer use depth, the css works it out itself due to the fact that files of a subfolder are children.

Menu

* Menu template creation – different for windows and mac

Editor

* Editor of choice – Monaco editor, why?
* Had to use requireJS in order to use the module for it
* Monaco editor models – explain

Questions for Ian –

Be positive about about features I wasn’t able to add – explain how I’d add things that I haven’t, show that I know how to do it, and have adapted my time for the more important things.

* Executable for application?
  + If I can
* Referencing – modules
* Professional considerations – needs a chapter Bcs considerations, code of conduct, site it. No ethical considerations for this project.

Code- explain interesting and difficult parts

Something taking more time- for example the folders - will take up more of my time, pushing other work back to further work.

Nobody checks interim requirements. Say a requirement is optional if I haven’t got it done.

Sections for report

* Introduction – 690 words, could increase
* Motivation – 153 words
* Background Research – 542 words
* Professional Considerations – 93 words, should increase
* Requirements analysis – 262 words
* Choice of technology – 1431 words
  + Electron
  + Comparison of technology choices
* System Design
  + Folder structure
  + Files –
    - main,
    - renderer
    - preload
    - node\_modules
    - package.json
  + Node Modules
    - Electron
    - IPC
    - requireJS
    - Monaco
    - Xterm and node-pty
    - os
    - fs
    - path
  + JavaScript Module types
  + Version control - github
* System Implementation
  + Basic layout
  + Explorer
  + File handling
    - Open file
    - Save file
    - Create file
  + Folder handling
    - Open folder
  + Tabs
    - Style
    - Functionality
  + Editor
  + Terminal
  + Menu
* System Evaluation
  + Testing
  + Further work (what I haven’t been able to do – make positive)
    - Landing page
    - Interactive coding features
    - Auto-checking folder contents
    - Improve UI
* User Interface screenshots

Remaining words: ~7400 words