Functional Specification

Project Title: GitMD

Student 1 Name: Ciaran Skelly ID Number: 20324213

Student 2 Name: Aaron Crawford ID Number: 20336753

Date of Completion: 16/11/2023

Table of contents

1. Introduction **1**
1.1 Overview 1
1.1.1 High-level Overview Diagram 2
1.2 Business Context 2
1.3 Glossary 2
2. General Description **3**
2.1 Product / System Functions 3
2.2 User Characteristics and Objectives 3
2.3 Operational Scenarios 4
2.4 Constraints 5
3. Functional Requirements **6**
4. System Architecture **8**
5. High-Level Design **9**
6. Preliminary Schedule **10**

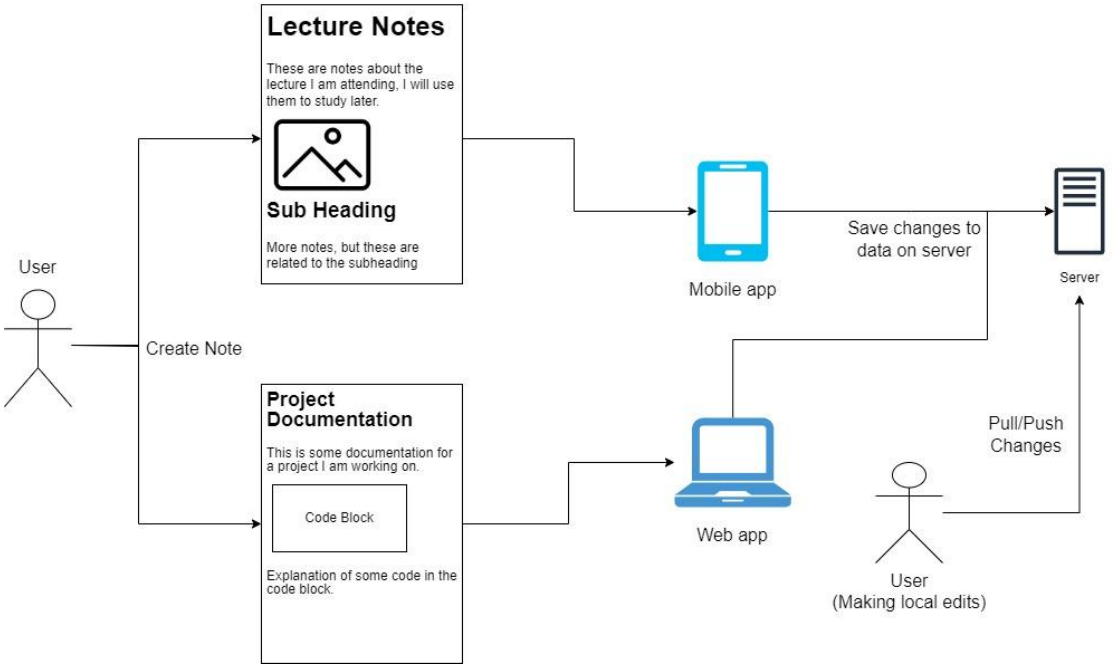
1. Introduction

1.1 Overview

The project involves developing an online note taking application capable of storing and managing notes. The system will comprise of a backend, git based server and clients for the web and Android platforms with notes being written in markdown. Users will be able to create, edit, view and delete notes through an intuitive ui on both android and web platforms. Git's version control will allow users such as writers, researchers or developers to maintain a history of their work while also serving as a backup system if notes are mistakenly deleted. Git repositories will allow users to access their notes on multiple devices and collaborate with other users on the same markdown files by sharing a repository ideally for

teams working on projects or research groups. Notes can also be accessed and edited offline with any changes made being synched and merged when an internet connection is available.

1.1.1 High-level Overview Diagram



1.2 Business Context

We don't have any plans to make our project into a business. If we desired to do so, we could implement some sort of user charging system based on the storage space users would require to store their notes, similar to how Google and other companies charge users for cloud storage. To appeal to students that might use our system for storing their notes we could have discounted student rates.

1.3 Glossary

Git is a version control tool that saves your work's history, so you can undo changes, collaborate with others, and keep track of different versions of your project.

Markdown is a simple way to format text using symbols for headings, lists, emphasis, and links, making it easy to write for the web without complex coding.

2. General Description

2.1 Product / System Functions

This project aims to allow users to create, edit, view and delete notes that are stored as markdown files. Building on a Git backend will allow users to use it's version control. This means users will have access to their notes history to see progress and to act as a roll back system in case of error or deletion. Gits repository will also allow users to access their notes across multiple devices and share their notes with other users by giving them access to the gits repository. Users will also be able to access their notes offline and edit them, then when they have an internet connection their notes will be synched and merged to those attached to their account. Users will be able to access our project on an android app, web app or pull their notes to their own machine using git.

2.2 User Characteristics and Objectives

(Use Cases: Students working on projects / assignments, Software Engineers working on documentation for a product / feature launch)

Students who are given assignments or projects will need to be able to easily create documents and share those documents with other students they are working with. They will require a consistent UI across different devices where they will be editing or viewing their documentation. Our project will have to be available both as a web application and mobile application to assist students in accessing, editing and viewing their documents in as many places as possible. Students will also require lecture notes that can be made clear to read and include visuals such as images.

Software engineers will require a history of work and security while working on documentation for products or features. We will ensure that when sharing a document with another user they can have either viewing access or editing access to ensure no loss of work. Through Git we will also verify that there is no loss of work when two users are working on the same document. Markdown will allow code blocks throughout the document to showcase specific features.

2.3 Operational Scenarios

The use case diagram below shows the relationship between the User and our application, and how they will use each feature that we provide.

First a user will have to create an account and this function will ensure that the credentials (username, email, password) that they have provided are unique to this user. When a user attempts to log in we ensure that the correct credentials for that user have been entered, if they are incorrect we display an error message and if they are correct we log that user into their account.

If a user wishes to create a markdown file we ensure that they have included both a title for their file and content in the file otherwise the file is not created as to avoid error in the file creation process. When they have completed that they may save the file and when they save the file a Git Commit is made so the user will be prompted to add a commit message, which will help us with the version control of the file. Once a git message has been added the file is pushed to our git server.

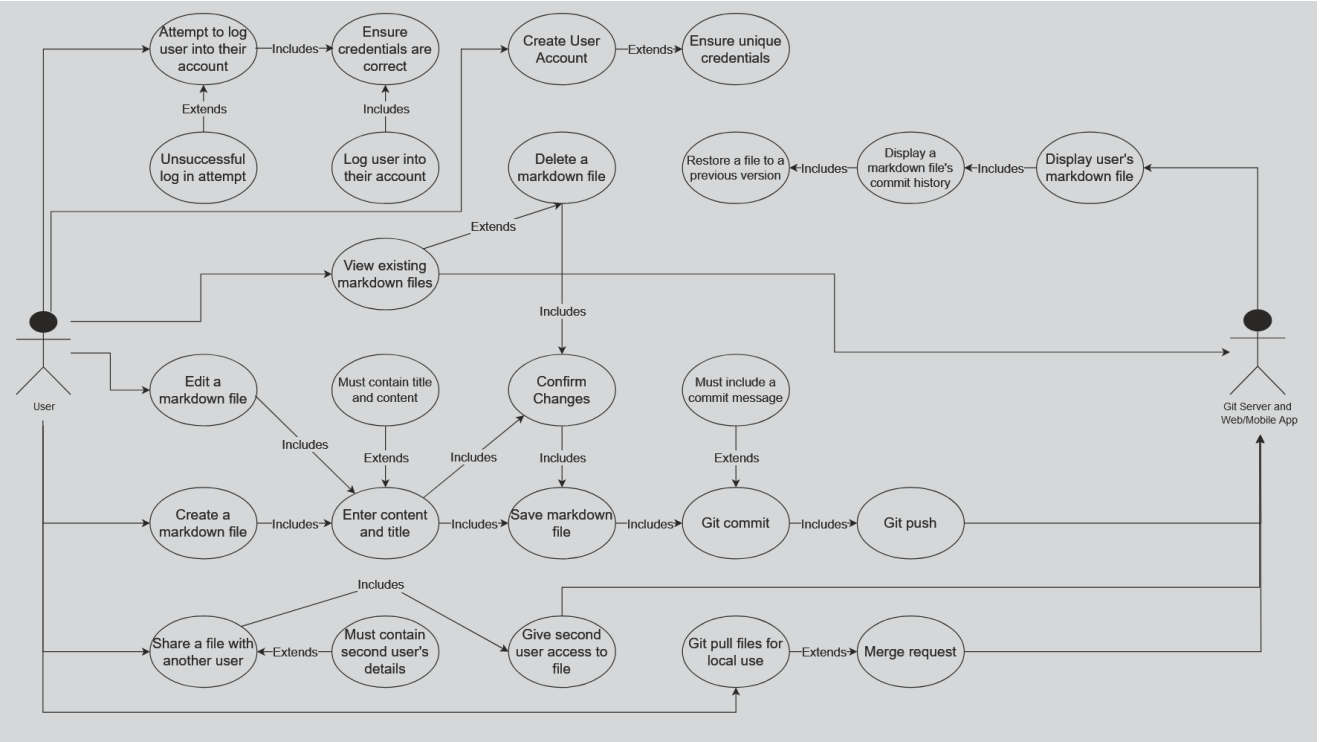
A user can view all their markdown files and from here they have the option to delete a markdown file by making a git commit and supplying a commit message followed by a push to our git server.

Editing a markdown file follows the same path as creating a markdown file where changes are saved, committed and pushed to our git server.

If a user wants to share file access with another user they must supply that user’s email which is attached to that user's account which will then allow that user to edit that file.

A user can pull their markdown files to their local machine to edit offline however if the user has the files on the machine already and they are not matching the pulled files they will be served with a merge request to resolve any differences.

Finally the git server will be able to offer the user the previous versions of their markdown files through the previous commits. A user can use this to restore a file to a previous version.



2.4 Constraints

UI Constraints

As this product is targeted towards a wide range of users the user interface has to be clear and easy to read. We have to assume that not all potential users are tech savvy.

Time Constraints

We will have to start work early to ensure we have a complete and finished project without having fourth year exams and modules taken away from our project.

Consistency Constraints

Our project will be available on web and mobile so we will have to ensure that the features and layout of both are consistent.

3. Functional Requirements

Requirement	Save Notes
Description	Changes made to notes will be able to be saved.
Criticality	Highly critical, as it is a core functionality to be able to save any changes.
Technical Issues	Must sync with other devices.
Dependencies	A new note must be created or an existing note must be edited.

Requirement	Commit changes
Description	Saved changes will be committed to the users repo on the server.
Criticality	Critical, as changes must be committed so that they can be synced across platforms.
Technical Issues	
Dependencies	Must have saved changes to commit.

Requirement	Create new notes
Description	Create a new, blank note file.
Criticality	Critical, as users must be able to make new notes
Technical Issues	
Dependencies	New notes depend on Saving and committing so they won't be lost.

Requirement	Delete notes

Description	Delete existing note files.
Criticality	Less critical, but would be an annoyance to not be able to delete files.
Technical Issues	
Dependencies	Must be existing files to delete.

Requirement	Share notes
Description	The ability to share notes with other users so that they can edit and view notes.
Criticality	Not critical, the application could work without the ability to share notes.
Technical Issues	Access control with Git.
Dependencies	Must be an existing note to share and another account to share with.

Requirement	Edit notes
Description	Make changes to the text or layout of an existing note.
Criticality	Critical, it's a core feature of the note taking application to be able to make changes to notes.
Technical Issues	
Dependencies	Depend on Saving and committing so changes won't be lost they won't be lost.

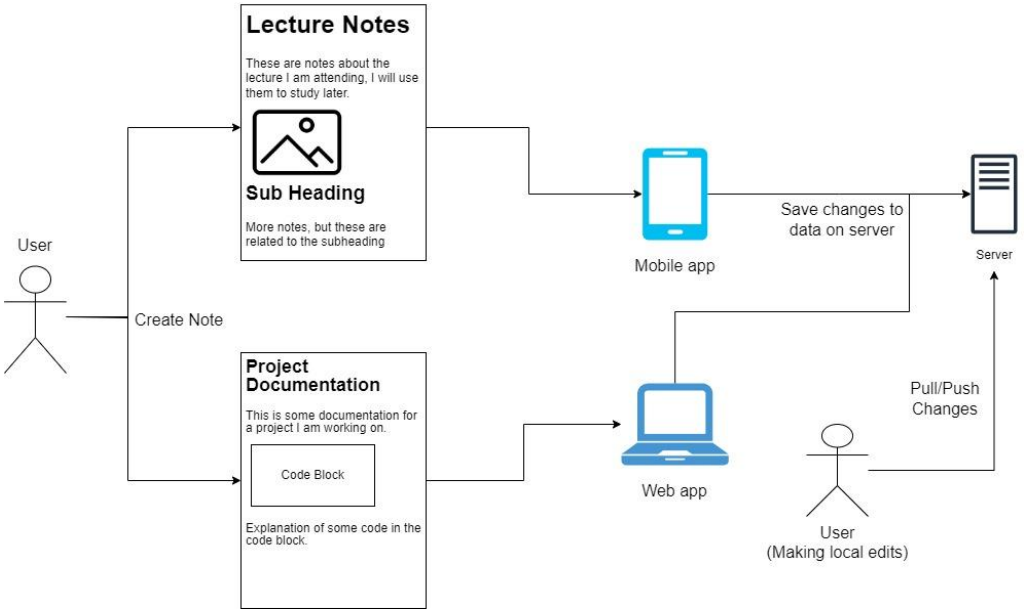
Requirement	Pull files for local use
Description	The ability to pull the repo down so you can edit it locally. This means you would be able to make edits to notes using your own text editor and then push those changes to sync them with what is on other platforms.

Criticality	Not highly critical, it is more of an extra feature that should not affect the overall system.
Technical Issues	
Dependencies	

Requirement	User account login
Description	Each user will have an account they must log into so they can access their notes.
Criticality	Critical, must be able to separate access to notes by user
Technical Issues	The login must be secure and will require users to set up a password.
Dependencies	

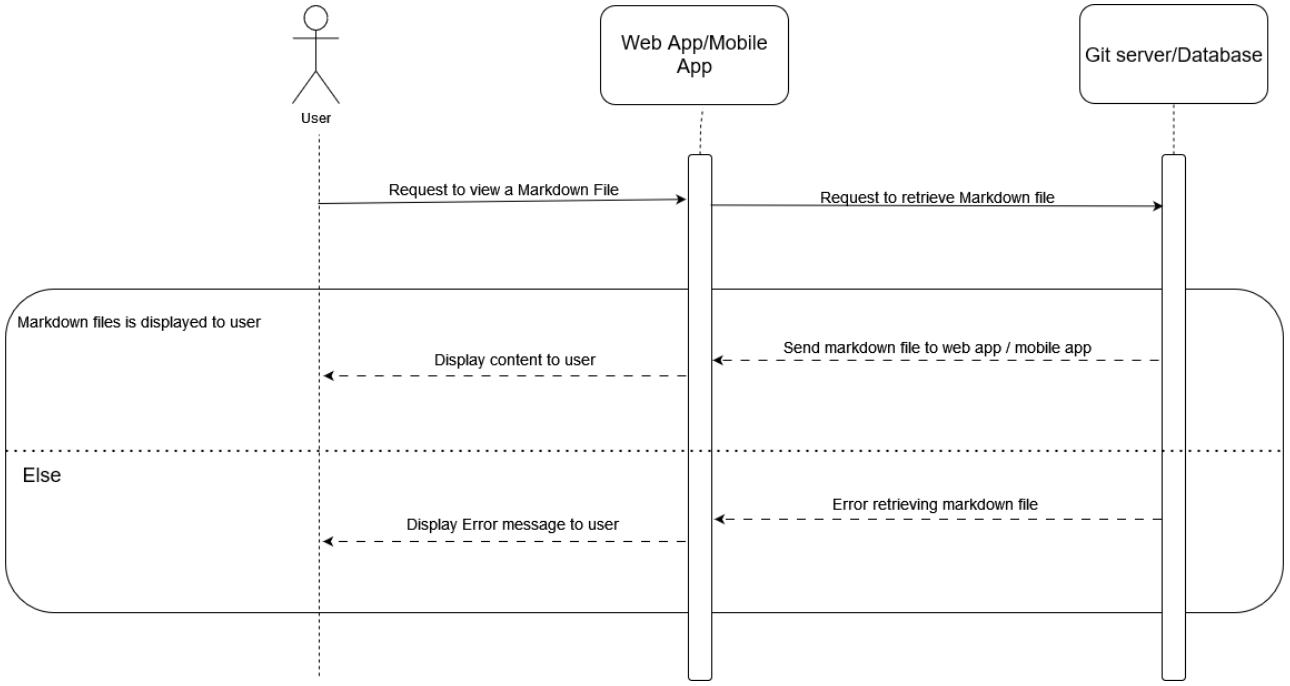
4. System Architecture

Our system architecture has 4 components, users, mobile and web applications and the server. Users will interact with the system using the mobile or web application. Using the application they will be able to create, read, update and delete notes. When note changes are saved a request is made to the API which will handle making the changes to the files on the server using Git commands. Users can also make edits to notes using their own text editors and then push those changes to their note repo on the server.



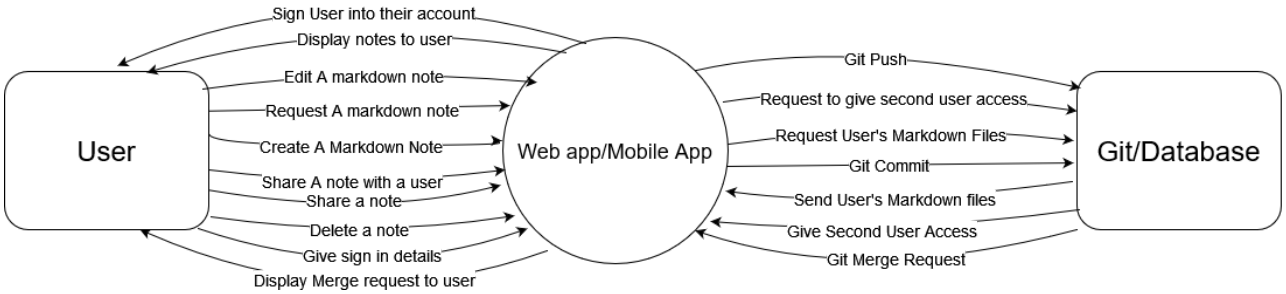
5. High-Level Design

Sequence Diagram:



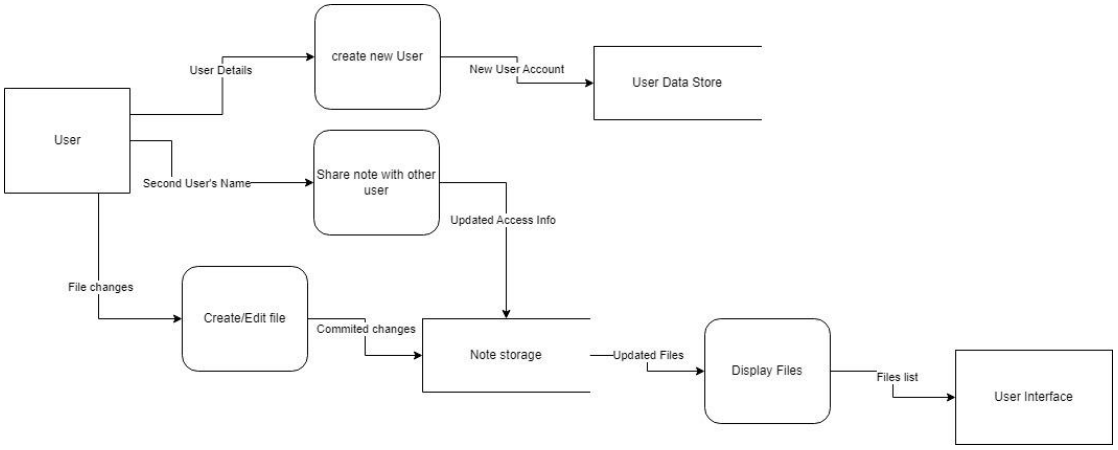
The sequence diagram above shows how the user would view the contents of a specific markdown file. First the user would interact with the frontend of the web app or mobile app, this would then send a request to the backend database to where the markdown file is stored to retrieve it. The app will then receive either the requested markdown file or an error message which could be if the user does not have access to the file or the file does not exist and either of these options will be displayed to the user.

Context Diagram:



The context diagram shows the elements in the system and how they interact with each other. It shows how the user controls the overall system and how each function would be completed by them.

Data Flow Diagram



This data flow diagram provides a high level representation of the flow of data through our system.

6. Preliminary Schedule