School of Computing
CA326 Year 3 Project Proposal Form

SECTION A

Project Title: Noted

Student 1 Name: Ali Ahmad **ID Number:** 22312403

Student 2 Name: Kaushal Sambhe **ID Number:** 22388443

Staff Member Consulted: Prof. Stephen Blott

Project Description:

The name of our project is Noted. It is a Markdown based note-taking application that provides a unified experience for both web and terminal users. The need for developers to create and manage notes on the terminal is well known, but most of these tools that exist do not have a web interface for broader accessibility. Our project, Noted, aims to bridge this gap by allowing users to create edit and organise Markdown notes from the terminal using a Go-based command-line client as well as from a React web application.

Users will be able to login via Google or GitHub OAuth or can login by creating an account. Key features include a built-in version control system that allows users to view and restore previous versions of their notes. This will create a “Git” like interface, allowing users to interact with their notes in a way that resonated with most developers. The interface on the terminal client or the web will offer a simple way to manage changes, making it intuitive for users to track their note evolution and revert to earlier states when needed.

Additionally, we will implement end-to-end encryption to ensure the security and privacy of user notes. This means that notes will be encrypted on the user’s device before being stored on the server, ensuring that only the user can access the content. Each note will use a shared key, which can be securely shared by collaborators using the public/private key pair generated for each user upon account creation.

Real-time collaboration will also be added allowing users to work on notes simultaneously, regardless of whether they are using the terminal or web client. By using the shared key, the editing of notes will remain secure, allowing users to see changes while the confidentiality of the note is maintained.

Users can also create tags for notes, allowing them to categorise and organise notes by various topics. This feature complements the full text search functionality that will be implemented, allowing users to easily find notes based on their tags, title and

even the content within the notes. This is especially beneficial for the terminal client, where users can quickly search for specific notes without having to go through the painful experience of sifting through each note individually.

An issue with most Markdown based note apps have is image handling. To solve this, we will integrate Amazon S3 to store images. When a user wants to upload an image, a link will be generated which the user can then use in the Markdown file. This process reduces storage requirements and keeps the app lightweight. By using S3, users can access their images from any device, across both the terminal and web clients.

Another one of our big features is the ability to preview the Markdown in both the terminal and web client. While live preview is only available on the web, users can still see how their note looks as they write it. Users can immediately see how their formatting will appear, increasing productivity compared to other markdown note-taking apps.

This project will be useful for developers who prefer working from the terminal but still want the convenience of a web interface. The different features aim to solve the problem of notes being scattered across different platforms.

Division of Work:

While we are both interested in our own topics, we wanted to split the work effectively to make the most of the learning experience with the new frameworks and languages.

Ali Ahmad

- Implement the backend of the application, creating APIs for communication.
- GitHub and Google OAuth.
- Create and configure API unit tests.
- Set up the version Control System in MongoDB.

Kaushal Sambhe

- Implement the frontend design and the markdown editor and renderer.
- Set up connection between the backend and the database.
- Write and configure unit tests for the frontend and the terminal client.
- Create the full text search functionality.

Shared Tasks:

- Implement the terminal client.
- Write and configure end-to-end tests.
- Develop real time collaboration for terminal and web.
- Develop end-to-end encryption.

Programming Languages:

- The backend of this project will be written in JavaScript using Express.
- The frontend will be written in JavaScript using React.
- The CLI will be written in Golang.

Programming Tools:

- **Backend:** Express, MongoDB, JWT, OAuth
- **Terminal Client:** Cobra, Glamour, Crypto
- **Web Development:** React**,** Redux.js**,** Tailwind CSS, markdown-it, codemirror, Y.js for collaboration, crypto.js
- **Image Hosting:** Amazon S3
- **Version Control:** Git, GitLab
- **Testing Frameworks:** Playwright, Jest

Learning Challenges:

We expect to face several challenges during this project. The first would be learning new programming languages and frameworks. We have no experience in Golang or in Cobra – the framework used to build command line interfaces. We are also new to building backends with Express and Node.js. In previous modules where we used Django, everything was built in, so using MongoDB as an external database will be a big change. Figuring out how to connect the backend to the frontend and how everything links together will be quite challenging. We plan to use this as a learning experience, trying to learn new languages and frameworks to increase our knowledge while doing this project .

Implementing OAuth with Google and GitHub as a separate method for user authentication will be tough. We will need to understand how OAuth works and how we can keep and persist user sessions securely between the frontend and the CLI.

Another challenge we will come across is creating an efficient version control system in MongoDB. We need to find the best way to store and access different versions of notes without slowing down the app.

The encryption and real time collaboration will certainly cause challenges as this is something we have never seen before. We will need to learn how to securely manage encryption keys and ensure that our real-time updates don’t compromise data security.

Hardware / software platform:

- **Hardware:** Linux
- **Software:** VsCode

Special hardware / software requirements:

No special hardware / software requirements are needed.