**Project Proposal**

**“Secret Memo“**

**Introduction**

Nowadays, effectiveness and privacy of the communication are as important as never before because of the development of the informational technologies. Secret Memo’s plans to meet this need by allowing users to create encrypted notes and share them via the internet. The notes read here shall be modified to wipe out after the reading process, assuring the privacy of the information passed. Developed from the Django web framework and integrating a SQLite database, Secret Memo will also allow users to register/login to the website and personalise their profiles’ security needs.

**What?**

Secret Memo is a full-stack application that is intended to cater the secure sharing of Temporary Note. Using this application, users are going to be able to effectively write notes and have them stored on a server while being encrypted and only accessible through an individual URL. When used, these notes self-expose and self-erase so that any information that is deemed sensitive does not hang around when it does not have to. Furthermore, the application will offer users with the possibility to create accounts and sign in, so they will be able to personalise notes and profiles.

**Why?**

Proprietary Secret Memo’s purpose is to share the temporary information securely. In the current world of increased data loss threats, it is crucial to make sure that the confidential data is shared and stored securely. Some of the specific uses that Secret Memo provides include encryption and automatic deletion of notes which appeal to users who care about privacy. In addition, user authentication is used to provide additional layers of security so that some areas of the application can only be accessed by certain users.

**List of Supported Features**

* **Secure Note Creation and Sharing**: Users can create the notes that will be encrypted and can be opened only when a user opens the URL.
* **Encrypted Data Storage**: That involves encrypting all notes with clients in a database, which they accept.
* **Automatic Note Destruction**: To complement paper based notes, the notes taken automatically self erase upon being read by the recipient.
* **User Registration and Login:** They can welcome membership registration and enter the site, and operate corresponding account login and data.
* **Profile Management**: There, users can edit the personal information and check the activity history.
* **REST API Support**: Every function is exposed via RESTful API endpoints, thus enabling other services to link perfectly with it.

**Not Supported Features**

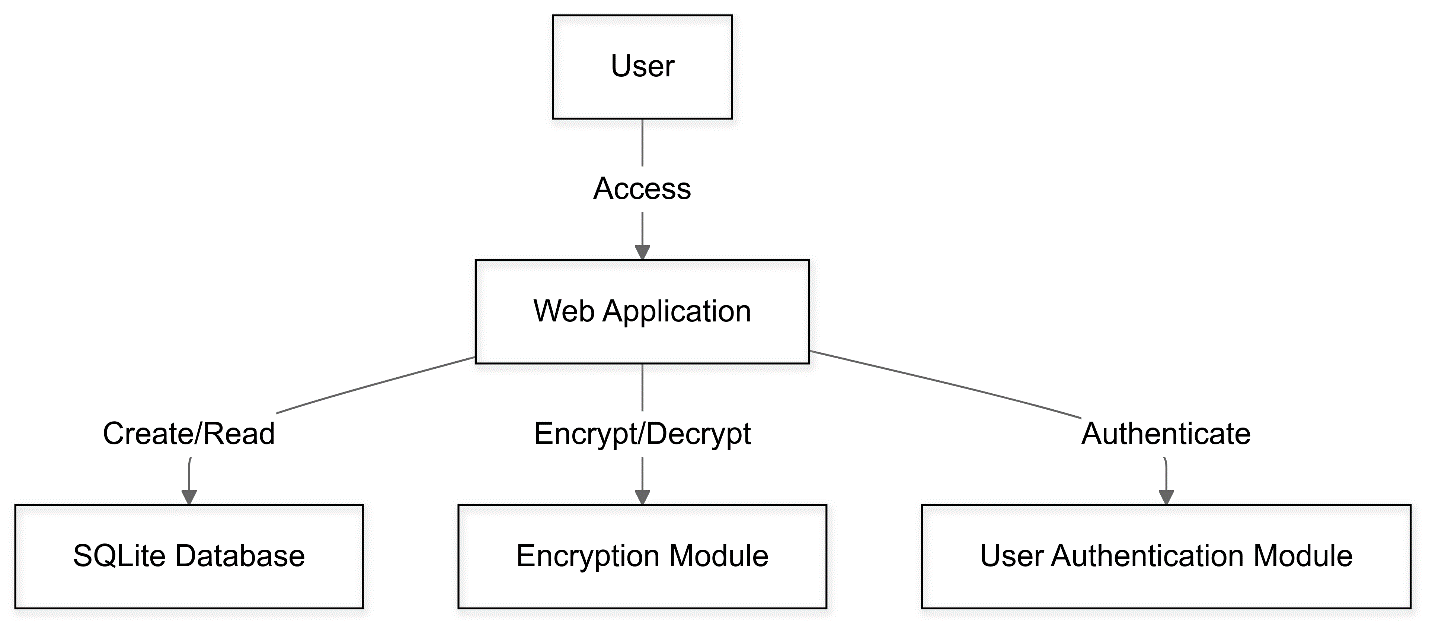
* **Mobile Application:** However, the access will be provided through the website at first with the mobile application functionality under consideration.
* **Multi-language Support**: The first release is going to be done in English, only.
* **Real-time Collaboration:** Users will not be able to edit notes real time.

Below is a list of features that are planned to include in later future:

* **Mobile App Integration:** Their expansion on a web platform, internal development of applications for Android and iOS.
* **Multi-language Support:** Extending functionality to include multiple language for non English speaking users around the world.
* **Advanced Encryption Options**: Allowing users to be given more options as to encyption styles.
* **Real-time Collaboration**: Letting users work at notes in different capacities at the same time.

**How**

**High-Level Diagram**



**List of Components/Modules**

* **Web Application Frontend**: It is a user interface designed using HTML, CSS, and one of the most popular programming languages JavaScript.
* **Web Application Backend:** The server-side logic using Python language with the help of Django.
* **Encryption Module:** Responsible for encrypting as well as decrypting the notes.
* **User Authentication Module**: Handles the registration process of users and their login details and profile authorization.
* **Database Management**: Stores users data and notes in the SQLite database.

The languages to be used for each of the modules just as stipulated is as follows:

* **Frontend**: HTML, CSS, JavaScript
* **Backend**: Python (Django)
* **Database**: SQLite

**Third Party /Open Source Module**

* **Django**: A python based MVC web framework which is high level and increases the speed of development.
* **PyCrypto**: It has enumerated secure hash functions and a lot of encryption algorithms.

**Table of Licenses**

|  |  |
| --- | --- |
| Module | License |
| Django | BSD License |
| Django REST Framework | BSD License |
| PyCrypto | Public Domain |

**List of any 3rd Party Services/APIs**

* **None initially planned;** all functionalities will be developed in-house to maintain control over the application's security and performance.

**REST API Endpoints with Payloads**

* **POST /api/notes/:** Create a new note
  + Payload: { "content": "string", "expires\_in": "integer" }
* **GET /api/notes/{id}/:** Retrieve a note using its unique URL
* **POST /api/register/:** Register a new user
  + Payload: { "username": "string", "password": "string" }
* **POST /api/login/:** Log in an existing user
  + Payload: { "username": "string", "password": "string" }

**Build Steps/Scripts**

* **Install Python and Django:** Ensure the system has Python installed, then install Django using pip.
* **Set up a Virtual Environment:** Create and activate a virtual environment for the project.
* **Install Required Packages:** Use pip install -r requirements.txt to install all necessary dependencies.

**Install Steps/Scripts**

* **Clone the Repository:** Use Git to clone the project repository to your local machine.
* **Run Migrations:** Execute python manage.py migrate to set up the database schema.
* **Start the Server:** Use python manage.py runserver to launch the development server.

**GitHub Information**

* **Repository URL:** https://github.com/Avinash-code77/comp5130/upload/week3
* **Branching Strategy:** The main branch will contain stable releases, while the develop branch will be used for ongoing development and feature integration.

**References**

* **Django Documentation:** [Django Official Docs](https://docs.djangoproject.com/)
* **SQLite Documentation:** [SQLite Official Docs](https://www.sqlite.org/docs.html)
* **PyCrypto Documentation:** [PyCrypto Documentation](https://www.dlitz.net/software/pycrypto/" \t "_blank)

This proposal provides a detailed roadmap for the development of Secret Memo, ensuring that all key aspects of the project are addressed.