Project Plan and Requirement Analysis
Group-3 Team-2

Problem: Develop and implement a fail-safe Web-based system that can:
- Ensure private encrypted communication between two users in the form of basic text messages.
- Store and safeguard any previous communication between the users.

Requirement Engineering

Expectation: The system needs to implement asymmetric encryption in a way which nullifies the possibility of any entity(other that the 2 communicating users) intercepting the communication between the users.

Functional requirements of the Web-based application:
- The Web-based application should allow the site-visitor to create an account in the database and access it after **authentication** of required credentials.
- The application will create a folder in the user's local storage to store and access the private key and message history.
- The application will store and access the credentials from the cloud-based storage.
- The application should allow the user to **alter** the credentials after thorough authentication.
- The application will access the cloud-storage and save any newly received messages in the local storage after decryption, every-time the user Logs-in.
- The application will assign a public key and a private key to each user. The public key will be stored in the cloud storage and the private key will be locally stored.
- The web application allows the user to chose some of the existing users as his friends with their consent.
- The application will allow the user to communicate only with the users present in corresponding list of friends.
- The application will allow the user to view message history with a particular friend.
- The application will store and access the message history from the user's local storage.
- The application will allow the user to send messages to a friend, these messages will be stored in encrypted form in the cloud.
- The application will allow the user to receive messages from friends, the messages will be decrypted and stored in the local storage of the user.
- The application will keep the record of time at which a particular message is sent or received.
- The application will show the message history in order of time.

Project Plan

Members and Roles:

- Anurag Singh Naruka(IMT2020093) : Website Design and Implementation
- Ashish Gatreddi(IMT2020073) : Storage (Cloud and local)
- Keshav Goyal(IMT2020101) : Storage (Cloud and local)
- Riddhi Chatterjee(IMT2020094) : Cryptography
- Teja Janaki Ram(IMT2020100) : Message(logs, tags, message history)

Time required for various deliverables:

- Website Design and Implementation : 2 weeks (Due on: 27 Feb, 2021)
- Storage(Google API setup and management) : 2 weeks (Due on: 27 Feb, 2021)
- Cryptography : 1.5 weeks (Due on: 23 Feb, 2021)
- Message Database management : 1.5 weeks (Due on: 23 Feb, 2021)

Resources Required:

- **Libraries:**
  - **Django** : Django is a free and open source web application framework, written in Python. It provides a set of components that help to develop websites faster and easier. It provides solutions to handle user authentication (signing up, signing in, signing out), a management panel for the website, forms, a way to upload files, etc.
  - **PyCryptodome** : PyCryptodome is a self-contained Python package of low-level cryptographic primitives. It will be used to implement RSA or Asymmetrical encryption in the project. PyCryptodome provides solutions for Encryption, Decryption, generating keys etc.
- **Python 3.6**
- **GIT**
- **API:**
  - **Google Drive API** : The Google Drive API will allow us to leverage Google Drive storage. It will help integrate the web-based application with Drive. It will provide cloud-space and the ability to download, upload, read and edit files from the cloud-storage.

References

- https://developers.google.com/drive/api/v3/about-sdk : Instructions to set-up Google Drive API and make use of various available functionalities.
- https://medium.com/@ashiqgiga07/asymmetric-cryptography-with-python-5eed86772731 : Information regarding the application of PyCryptodome.
- https://docs.djangoproject.com/en/3.1/ : Information for Django setup and implementation.
- https://gist.github.com/syedrakib/241b68f5aeaefd7ef8e2 : A sample of RSA-encryption using PyCryptodome.
- https://www.datacamp.com/community/tutorials/web-development-django : Information for Django setup and implementation.
- https://django-googledrive-storage.readthedocs.io/en/latest/ : Google Drive API and Django integration.
- https://pip.pypa.io/en/stable/reference/pip_download/ : Information regarding PIP setup for installation of required packages.
- https://cheapsslsecurity.com/blog/what-is-asymmetric-encryption-understand-with-simple-examples/ : Explanation of Asymmetric Encryption.