October Milestone

Tech Stack

a. Frontend - ReactJS
b. Backend - Django
c. OS - Linux
d. Web Server - Nginx
e. Database - MySQL/Sqlite

User Login and Authentication

We made a Django authentication using a custom user model from scratch. Using our model, we are able to send the user a verification code on their mail, which we use for 2-factor authentication. During this implementation, we faced multiple difficulties, the biggest one being implementing our own authentication system and linking it to the SMTP client. Our user model consists of very basic and essential information, for now, i.e., Name, Email ID, password and blood group. We have checks in place to ensure every email id and username used is unique. We are also implementing csrf tokens to protect against malicious requests that could pretend to originate from our site.

File Upload

Due to VM SSH failures, we could not connect to HTTPS and were unable to configure our web server NGINX, so we locally hosted the website for now. On running with the command localhost:8000, login shows up, but upon typing localhost:800/file, the file upload system is shown where one can upload their required documents, and the admin can access these files and review them. The documents are also stored in a media file(created in the src folder upon document submission) which is created once a document is uploaded on the website.

All users and documents uploaded are stored in the Django default SQLite database and can be viewed, edited and deleted in the Django admin page.

References:

https://medium.com/analytics-vidhya/build-custom-django-authentication-with-your-own-user-model-from-scratch-with-email-verification-54a3dad2130d