CRYPTGO

25JE0716

Work completed so far

So far, I have started building my project **CryptGo**, which is an End-to-End Encrypted File Storage platform.

I created the backend using **Python Flask** and connected it to a **PostgreSQL** database. I created a database connection file (db.py) and tested it to make sure the server and database are working properly.

After that, I created a **user signup feature** on the backend. User passwords are hashed before storing them in the database so that plain passwords are never saved.

On the frontend, I learned the basics of **HTML and CSS** and created a simple homepage explaining the project. I also created a **Signup page** using HTML and CSS.

Using basic **JavaScript** (signup.js), I connected the signup page to the backend API. I tested the complete flow and was able to successfully create users in the frontend and store them in the database. I even tried adding the same user again and faced an error message saying “User already exists”.

Challenges faced

One major challenge was learning frontend technologies like HTML, CSS, and JavaScript from scratch. Initially, it was difficult to understand how the frontend communicates with the backend.

I also faced issues related to **CORS errors** when connecting the frontend to the backend, which took time to understand and fix. I took the help of LLMs to understand the issue and found the solution to the problem.

Another challenge was setting up **PostgreSQL** and understanding how database connections and queries work in a real application. Debugging backend errors and understanding error messages was challenging but helped me learn better.

Research / concepts learned

During this period, I learned the basics of:

- HTML and CSS for creating web pages
- JavaScript for handling form submissions and API calls
- Flask for creating backend APIs
- PostgreSQL for storing user data
- Password hashing for secure authentication
- How frontend and backend communicate using JSON

I also learned the importance of security practices like hashing passwords and handling user input carefully.