🔐 Cybersecurity Data Protection System — Design Document

1. Project Summary

This project focuses on developing a Data Protection System designed to help users securely store, access, and manage sensitive information. The product matters because cyber threats, such as data breaches and unauthorized access, continue to rise, and individuals and organizations need user-friendly tools that enforce strong security practices.

2. Problem Statement

Users often store sensitive files and passwords without proper encryption, which exposes them to hacking, identity theft, and data loss. This product solves the challenge of protecting data even if a device is hacked or stolen.

3. Use Case

This product will be used by students, professionals, and small businesses who need to secure confidential data. They will use it to encrypt files, store passwords, back up important documents, and control access to sensitive information.

4. Goals and Objectives

1. **Create an easy-to-use tool for securely encrypting sensitive files.**
2. **Provide secure password storage and two-factor authentication to protect user accounts.**

5. Key Features and Functions

1. **File Encryption Module – Users can upload files and encrypt them using AES-256 encryption.**
2. **Password Vault – A secure vault that stores passwords using hashing and encryption.**
3. **Access Logs – Tracks all login attempts, alerts users to suspicious activity.**
4. **Two-Factor Authentication (2FA) – Sends a verification code for login security.**

5. **Secure Cloud Backup – Optional encrypted cloud storage for files and passwords.**

6. Tech Stack and Tools

- **Frontend: HTML, CSS, JavaScript**
- **Backend: Python (Flask) or Node.js**
- **Database: MongoDB or PostgreSQL**
- **Encryption Libraries: PyCryptodome / bcrypt**
- **Tools: GitHub, VS Code, Postman, Docker (optional)**

7. Algorithm (High-Level)

File Encryption Algorithm

1. **User selects a file to upload**
2. **System generates encryption key (AES-256)**
3. **File is encrypted using encryption library**
4. **Encrypted file stored in local storage or cloud**
5. **Decryption only occurs when user enters correct key/PIN**

8. Flowchart (Text Version — I can convert to diagram if you want)

Start → User Login → 2FA Check
→ Choose Action (Encrypt File / Password Vault / Access Logs)
→ If Encrypt File → Upload File → Encrypt → Save → Success Message
→ If Password Vault → Add/View Passwords → Save → Logout
→ End

9. Timeline (High-Level)

Month 1: Research + Planning + UI Wireframes
Month 2: Build Login System + Password Vault
Month 3: Build File Encryption Module
Month 4: Integrate Cloud Backup + Access Logs
Month 5: Final Testing + Documentation

10. Risk Mitigation

Risk: Weak user passwords may lead to account compromise.
Mitigation: Enforce password strength rules + 2FA + password hashing.

11. Evaluation Criteria

1. **Users can encrypt and decrypt files without errors.**
2. **Login system blocks unauthorized access and detects suspicious attempts.**
3. **Password vault stores and retrieves passwords securely with encryption.**

12. Future Considerations

- **Maintenance Need: Regularly update security libraries to patch vulnerabilities.**
- **Future Feature: Add biometric authentication (fingerprint/Face ID).**