**MINOR PROJECT REPORT**

**ON**

**VAULTX - ENCRYPTED FILE STORAGE SYSTEM** Submitted in partial fulfillment of the requirements

for the award of the degree of

**BACHELOR OF TECHNOLOGY  
 IN  
 INFORMATION TECHNOLOGY**



Submitted By

**Rudransh Shukla , Anshu Kumari , Jasleen kaur , Vaibhav Pandita**  
  
 Under the Guidance of

**Dr. Rupinder kaur** , **Faculty Supervisor, IT Department**  
  
  
 Department of Information Technology  
 Dr. Akhilesh Das Gupta Institute of Professional Studies, New Delhi  
 (Guru Gobind Singh Indraprastha University, Dwarka, Delhi)

October-2025

## DECLARATION

This is to certify that the material embodied in this Minor Project titled “VaultX – encrypted file system storage” being submitted in the partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in 2026 is based on my original work. It is further certified that this work has not been submitted in full or in part to this university or any other university for the award of any other degree or diploma. My indebtedness to other works has been duly acknowledged at the relevant places.

By :-

Rudransh Shukla (00415607723)

Anshu Kumari (01115607723)

Jasleen Kaur (03915603122)

Vaibhav Pandita (00315607723)

**** **Dr. Akhilesh Das Gupta Institute of Professional**

**Studies, New Delhi**

**Department of Information Technology**

## CERTIFICATE

This is to certify that the work embodied in this Minor Project - Dissertation titled “VaultX – encrypted file system storage” being submitted in the partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Information Technology, is original and has been carried out by **Rudransh Shukla (00415607723) Anshu Kumari (01115607723), Jasleen Kaur (03915603122), Vaibhav Pandita (00315607723)** under my supervision and guidance.

It is further certified that this Minor Project - Dissertation work has not been submitted in full or in part to this university or any other university for the award of any other degree or diploma to the best of my knowledge and belief.

**Dr. Rupinder Kaur**

**Faculty Superviser, IT Department**

**Dr. Rupinder Kaur**

**Project Cordinator, IT Department**

**Sanjay Kumar**

**HOD, IT Department**

## ACKNOWLEDGEMENTS

It gives me immense pleasure to express my deepest sense of gratitude and sincere thanks to my respected guide **Dr. Rupinder Kaur**, **Faculty Supervisor** and to **Dr. Akhilesh Das Gupta Institute of Professional Studies, New Delhi**, for their valuable guidance, encouragement and help for completing this work. Their useful suggestions for this whole work and co-operative behavior are sincerely acknowledged.

I also wish to express my indebtedness to my parents as well as my family member whose blessings and support always helped me to face the challenges ahead.

|  |  |  |  |
| --- | --- | --- | --- |
| Signature: | Signature: | Signature: | Signature: |
| Name: Rudransh Shukla | Name: Anshu Kumari | Name: Jasleen Kaur | Name: Vaibhav Pandita |
| Roll no.: 00415607723 | Roll no.: 01115607723 | Roll no.: 03915603122 | Roll no.: 00315607723 |
| Date: | Date: | Date: | Date: |

## ABSTRACT

Cybervault is a computer program designed to secure and organize confidential files totally off the internet. While the majority of cloud applications transfer data between servers and computers, CyberVault is a local application, meaning that the data never leaves your PC. It is created using Electron, React, and WebAuthn-based cryptography, with AES-256 encryption complemented by neural biometric verification for secure storage while remaining user-friendly.

The system employs multiple levels of security—a master password, file PINs, and biometric logon via face, iris, or fingerprint recognition. This has it more difficult for unauthorized individuals to access it, even on group systems.As it does not rely on internet connectivity or third-party servers, the users themselves own and control their data. The web interface was developed utilizing React and Tailwind CSS, allowing for a simple dashboard for users to work with files by uploading, decrypting, encrypting, and managing them.

CyberVault uses established security standards like AES-256, PBKDF2, and webauthn. The encrypting and decrypting are done locally which mitigates security risks. Encryption and decryption are done locally, which reduces security threats. The architecture and process of the project are described through context diagrams, use case diagrams, and data flow analysis, which gives insight into how the system works internally. Testing included encryption precision, biometric consistency, high-load performance, and general usability.CyberVault eventually seeks to offer a real-world and privacy-first data protection solution. It combines contemporary encryption, biometric authentication, and a simple interface to provide users with a secure and self-sufficient means of taking control of their personal documents without relying on the cloud.

**TABLE OF CONTENTS**

DECLARATION ii

CERTIFICATE iii

ACKNOWLEDGEMENT iv

ABSTRACT v

**CHAPTER 1: INTRODUCTION 1-13**

* 1. Organisational Background
  2. Organisational Structure
  3. Motivation
  4. Statement Of The Problem
  5. Objective Of The Project
  6. General Objective
     1. Specific Objective
  7. Feasibility Study
  8. Economic Feasibility
  9. Technical Feasibility
  10. Operational Feasibility
  11. Political Feasibility
  12. Schedule Feasibility
  13. Significance Of The Project
  14. Merits of the Project
  15. Demerits / Limitations of the Project
  16. Applications of the Project
  17. Beneficiaries Of The System
  18. Methodology
  19. Fact Finding
  20. Requirement Collection
  21. Requirement Analysis
  22. Quantitative And/Or Qualitative Analysis Of
  23. Data
  24. Software Development Model
  25. Scope And Limitation Of The Project
  26. Tasks Of The Team Memebr In The Project
  27. Project Team Organizations
  28. Time Schedule For The Project
  29. Gant Chart
  30. Pert Chart
  31. Critical Path Method
  32. Cost And Effort Measurment
  33. Cost Measurement
  34. Effort Measurement
  35. Resource Management
  36. Project Organization

#### Chapter 2: Literature Review 14-15

* 1. Literature Review
  2. Related Work
  3. Identification And Classification Faults

#### Chapter 3: Business Area Analysis And Requiremnet Analysis 16-22

* 1. Introduction
  2. Description Of The Existing System
  3. Activities Provided By The Existing System
  4. Major Functions Of The Existing System
  5. Strength And Weaknesses Of The Existing System
  6. Player Of The Existing System
  7. Business Rule Identification
  8. Evidence In The Existing System
  9. Problems Statement In The Existing System
  10. Practice To Be Preserved
  11. The Proposed System
  12. Functioncal Requirements
  13. Non Functional Requirements
      1. Correctness
      2. User Friednly/System Interface
      3. Response Time
      4. Reliability
      5. Performance
      6. Robustness
      7. Maintainability
      8. Effectiveness
      9. Efficiency
      10. Portability
      11. Development Costs
      12. Operational Costs
      13. Security
      14. Compatibility
      15. Availability

#### Chapter 4: Analysis And Deliverables Of The New System 23-32

* 1. Case Study Of Proposed System
  2. Data Flow Diagram

4.2.1 Context Diagram of the System

* 1. Use Case Diagram Of The System
     1. Description Of Actors
     2. System Use Case
     3. Description Of Actors And Use Case
     4. Use Case Diagram
  2. Identification Of Good And Bad Classes
  3. Data Dictionary
  4. Class Diagram
  5. Object Diagram
  6. Sequence Diagram
  7. User Interface Prototype Of The New System
  8. Architecture Of Proposed System
  9. Development Process
     1. Software/Hardare Components
     2. Alogorithm And Programming Language Used
     3. System Functionality And User Interface

#### Chapter 5: System Design And Architeture 33-49

* 1. Introduction
  2. Entity Relation Diagram
  3. Logical Model
  4. Physical Model
  5. Mapping Of Schema
  6. Normalization Of Data Base
     1. First Normal Form
     2. Second Normal Form
     3. Third Normal Form
     4. Bcnf
  7. Activity Diagram
  8. Collaboration Diagram
  9. State Diagram
  10. Component Diagram
  11. Deployment Diagram
  12. Persistence Diagram
  13. User Interface Design

#### Chapter 6: Result & Future Scope 50-51

#### 

* 1. Conclusion
  2. Future Scope

**LIST OF ABBREVIATIONS 52-53**

**APPENDICES 54-55**

**RESEARCH PAPER PROOF 56**

**LIST OF FIGURES 57**

**LIST OF TABLES 57**

**CHAPTER 1: INTRODUCTION**

## 1.1 ORGANISATIONAL BACKGROUND

CyberVault is conceived as a response to the growing need for secure and user controlled data protection in an increasingly cloud dependent world. This project is led by a team of developers, researchers, and security architects who knows in depth about encryption technologies, biometric authentication, and privacy first software design. Operating within a decentralized, research driven environment, the team prioritizes innovation in offline first security models. The idea behind CyberVault is not a traditional business but a collaborative initiative focused on building tools which empower users to claim control over their digital assets again.

**1.2 ORGANISATIONAL STRUCTURE**

The CyberVault development team follows a modular and agile structure. At the core of this there is a team which oversees technical developments and documentations as well as academic reporting. Surrounding this role are dedicated units for frontend development (React Tailwind CSS) backend and cryptographic engineering (Python, Node.js) biometric integration (face api.js, WebAuthn), and user experience testing. Each module operates semi independently but always aligns through iterative review cycles and shared architectural goals. This structure ensures flexibility as well as more rapid prototyping, and high accountability across all layers of the systems.

**1.3 MOTIVATION**

The motivation behind CyberVault originates from a very simple but powerful realization that users deserve full sovereignty over their personal data. In professional and shared environments the traditional encryption tools are either too complex or rely heavily on cloud infrastructures exposing users to external vulnerabilities. CyberVault is made out of frustration with these limitations and a desire to create a tool which is both technically robust and even intuitively usable. The team aims to build a system where encryption, authentication and even file management happens entirely on the user’s device without compromise.

**1.4 STATEMENT OF THE PROBLEM**

Despite the proliferation of data protection tools, most solutions today are either cloud dependent or they lack multi layered authentication. Users are often forced to choose between convenience and security. Existing systems mainly fail to offer a seamless biometric verification, granular file level access control, and offline functionality in a single package. This gap leaves users sensitive data vulnerable to breaches and especially in environments where internet access is restricted or mainly unreliable. The problem is that it’s not just technical it is architectural and experiential. Users always needed a solution that is secure, local, andvery easy to use.  
  
  
**1.5 OBJECTIVE OF THE PROJECT**

The primary objective of CyberVault is to develop a secure and offline-first desktop application that enables users to encrypt or authenticate and even manage sensitive files locally. The project aims to integrate advanced cryptographic standards with neural biometric authentication to deliver a privacy first experience. It also seeks to establish a new benchmark for usability in secure applications as well as ensuring that both technical and non technical users can confidently protect their data without relying on any external servers or cloud services.

**1.6 GENERAL OBJECTIVE**

To design and implement a cross platform desktop vault which combines AES-256 encryption with the multi modal biometric authentication and local first architecture to ensure complete data sovereignty and a user controlled security.

1*.6.1 SPECIFIC OBJECTIVE*

**Implementing AES-256 GCM encryption** for all the stored files and also using PBKDF2 SHA256 for secure key derivation and integrity validation. **Developing a multi layered authentication system** which includes a master password file specific PINs and even biometric verification (face, iris and fingerprint). **Integrate neural authentication frameworks** using face api.js and OpenCV.js to enable realtime, device level biometric login. **Build a responsive and intuitive user interface** using React and Tailwind CSS, with a dashboard for file management and security operations.

**Ensure complete offline functionality**, where all encryption, decryption, and authentication processes occur locally without internet dependency. **Conducting rigorous testing** for encryption reliability, biometric accuracy or performance benchmarks and usability across diverse user profiles or devices. **Prepare for future enhancements** with OCR based file indexing then an encrypted cloud backup option and a hardware token integration (e.g. YubiKey).

**1.7 FEASIBILITY STUDY**

Before moving on the development of CyberVault a comprehensive feasibility study was conducted to study the practicality, sustainability and impact of the project. This study examined so many dimensions like economic, technical, operational, political and even scheduling to ensure that the proposed system could be successfully implemented and maintained in all real world environments.

**1.8 ECONOMIC FEASIBILITY**

CyberVault was designed with so much cost efficiency in mind. It was possible by leveraging open source technologies such as Electron, React, and Python, the development team minimized licensing and infrastructure costs. The offline based architecture is used for cloud hosting or subscription based services which made this system economically viable for individual users, small businesses, and institutions. The use of local storage and device bound authentication also reduces long term operational expenses also offers a high return on investment for secure data management.

**1.9 TECHNICAL FEASIBILITY**

From a technical standpoint CyberVault is highly feasible. The system is built using mature well supported technologies like AES-256 GCM encryption which is used for data security then we have PBKDF2 for key derivation and face api.js for biometric authentication. These components are integrated into a modular architecture that supports scalability, cross platform deployments etc. The use of Electron in this ensures compatibility with Windows systems. It also works for macOS and Linux. Extensive testing on this confirmed that the system performs reliably even with large datasets and under offline conditions which is a bang on condition to go on.

**1.10 OPERATIONAL FEASIBILITY**

From a technical standpoint CyberVault is highly feasible. The system is built using mature well supported technologies like AES-256 GCM encryption which is used for data security then we have PBKDF2 for key derivation and face api.js for biometric authentication. These components are integrated into a modular architecture that supports scalability, cross platform deployments etc. The use of Electron in this ensures compatibility with Windows systems. It also works for macOS and Linux. Extensive testing on this confirmed that the system performs reliably even with large datasets and under offline conditions which is a bang on condition to go on.

**1.11 POLITICAL FEASIBILITY**

The system has a offline first design and local encryption model which supports data protection laws such as GDPR, HIPAA and India’s Personal Data Protection Bill. By avoiding cloud storage and external data transmission CyberVault reduces exposure to cross border data conflicts and surveillance concerns making it a politically sound solution for privacy conscious users and organizations.

**1.12 SCHEDULE FEASIBILITY**

The development timeline for CyberVault was carefully planned and executed in phases. Initially the prototyping, feature integration and biometric testing were completed within a six month window. Then the modular architecture allowed parallel development of encryption, authentication, and UI components, accelerating progress. Future enhancements include many things such as OCR indexing,cloud backup etc. They are scheduled for subsequent releases ensuring that the project remains agile and adaptable without compromising current functionality of other sections and modules.

**1.13 SIGNIFICANCE OF THE PROJECT**

CyberVault represents a significant advancement in field of secure data storage. It bridges the gap between high level encryption and user friendly design all while offering a solution that is both technically robust and even practically usable. This project introduces neural biometric authentication into local file protection which is a feature very rarely seen in desktop applications. By prioritizing offline functionality the CyberVault empowers users to retain full control over their data while setting a new standard for privacy first software.

**1.14 MERITS OF THE PROJECT**

Offline Security – All encryption and authentication operations are performed locally; no data leaves the user’s device. Strong Encryption – Implements AES-256 and PBKDF2 for high-grade data protection. Multi-Layer Authentication – Combines master password, file PIN, and neural (biometric) verification for enhanced safety. User-Friendly Interface – Clean, modern dashboard built with React + TailwindCSS, ensuring ease of use for non-technical users Secure File Sharing – Supports encrypted export packages and digitally signed transfers. Data Integrity & Reliability – Prevents file corruption through atomic encryption and safe write mechanisms. Cross-Platform Support – Designed with Electron, allowing deployment on Windows, macOS, and Linux (future). Camouflaged & Stealthy UI – Appears as a simple system utility (e.g., calculator or notepad) to prevent suspicion in shared systems Scalable Architecture – Modular design allows easy addition of features like OCR and cloud backup. Ethical and Privacy-Centric – Respects data sovereignty and avoids third-party data handling.

**1.15 DEMERITS/LIMITATIONS OF THE PROJECT**

Platform Limitation – Current version supports Windows only; macOS and Linux support planned. No Cloud Synchronization (Yet) – Files remain device-bound, limiting multi-device accessibility Limited Collaboration Features – File sharing works via encrypted packages, but no real-time sync or group editing. Performance Drop for Large Files – Encryption may slow down for files above 2–3 GB depending on system resource

**1.16 APPLICATIONS OF THE PROJECT**

Educational Institutions – Protect academic reports, research data, and confidential student files. Corporate Offices – Safeguard HR records, legal contracts, and project documents. Freelancers and Professionals – Secure client data, designs, or financial files without cloud reliance. Healthcare Sector – Store and encrypt patient reports and medical records securely offline. Legal and Government Use – Protect sensitive case files, confidential communication, and evidence data. Personal Use – Lock and manage private media, documents, or financial information locally. Cybersecurity Research – Demonstrates principles of local encryption, biometric authentication, and secure architecture for academic learning.

**1.17 BENEFICIARIES OF THE SYSTEM**

CyberVault is designed to cater to a varied user audience: Single users looking for safe storage for personal files, images, and credentials. Local data protection is essential for small businesses that want to avoid relying on the cloud. Educational organizations overseeing confidential student and staff information. Healthcare providers maintaining patient information in accordance with privacy laws.

**1.18 METHODOLOGY**

The creation of CyberVault adhered to a systematic approach that integrated research, requirement assessment, iterative design and even performance evaluation. The method guaranteed that each feature was based on actual needs and confirmed through actual evidence.

**1.19 FACT FINDING**

Fact finding included collecting information from current encryption technologies, biometric mechanisms and user input. This stage assisted in pinpointing shortcomings in existing solutions and guided the development of CyberVault’s framework.

**1.20 REQUIREMENT COLLECTION**

Requirements were gathered via literature review, competitor analysis and user interviews. Essential requirements encompassed offline capabilities, biometric authentication, file specific access management and a responsive user interface. These inputs highly influenced the fundamental characteristics of the system.

**1.21 REQUIREMENT ANALYSIS**

These gathered requirements were assessed for practicality, significance and integration possibilities. The team emphasized features that provided strong security while maintaining usability.

Trade-offs between performance and privacy were carefully evaluated to ensure balanced outcomes.

**1.22 QUANTITATIVE AND QUALITATIVE ANALYSIS OF DATA**

Quantitative analysis involved evaluating encryption speed, biometric response times, and the reliability of the system. Qualitative analysis emphasized user experience, clarity of the interface and perceived safety. Collectively, these approaches confirmed the system's efficacy in both technical and human aspects.

**1.24 SOFTWARE DEVLOPMENT MODEL**

CyberVault was created utilizing the Incremental Development Model, enabling the team to construct and evaluate features in phases. This model facilitated the initial launch of essential features like encryption and authentication, subsequently undergoing iterative improvements including UI enhancements and biometric adjustments. It also facilitated ongoing feedback incorporation, guaranteeing that the final product aligned with user expectationS

**1.25 SCOPE AND LIMITATION OF THE PROJECT**

**Scope :** CyberVault functions as a safe desktop vault for safeguarding local files. It features AES-256 encryption, multi-modal biometric verification, and an adaptive file management system. The system functions completely offline and is designed for Windows, with plans to support macOS and Linux in the future. Scheduled features consist of OCR indexing, encrypted cloud storage, and hardware token integration.

**Limitations**

Currently confined to desktop platforms; mobile compatibility is still unavailable.

The accuracy of biometrics can fluctuate depending on the hardware of the device and the lighting environment.

Cloud backup and OCR functionalities are being developed and will not be part of the initial launch.

The system needs initial configuration and biometric registration, which might present a challenge for users without technical expertise

**1.26 TASKS OF THE TEAM MEMBER IN THE PROJECT**

1. Cybersecurity Student : AES-256 encryption/decryption implementation, PBKDF2 key derivation setup, SHA-256 checksum generation, Security protocols and best practices, Password hashing and salt generation, Security testing and vulnerability assessment

2. Full Stack Developer : Face recognition implementation (face-api.js), Iris recognition implementation (OpenCV.js), Fingerprint authentication (WebAuthn API), Electron.js desktop app setup, Biometric data storage and comparison, Camera/sensor integration, Biometric modal components

3. Full Stack Developer #2 :React.js UI/UX development, Iris recognition implementation (OpenCV.js) File upload/download functionality, Vault interface and file management, Auto-lock system implementation, OCR integration , File integrity verification

4. Project Lead : Overall project coordination, Code integration and merging, Testing coordination, Backup/Restore functionality, Session management, Bug fixing and optimization, Project documentation.

**1.27 PROJECT TEAM ORGANIZATION**

**Project Lead & Architect :** Oversaw the entire system lifecycle — from architectural design to final deployment. Led encryption strategy, biometric integration, and documentation alignment with academic standards.

### Frontend Developer : Focused on dashboard design, biometric modals, and secure input handling all this while utilizing React and Tailwind CSS.

### Backend & Crypto Engineer : Implemented AES-256-GCM encryption, PBKDF2 key derivation, and file handling logic using Python and Node.js. . Ensured secure session management and memory clearance. Integrated face-api.js, OpenCV.js, and WebAuthn for facial, iris, and fingerprint authentication. Tuned neural models for accuracy and fallback reliability.

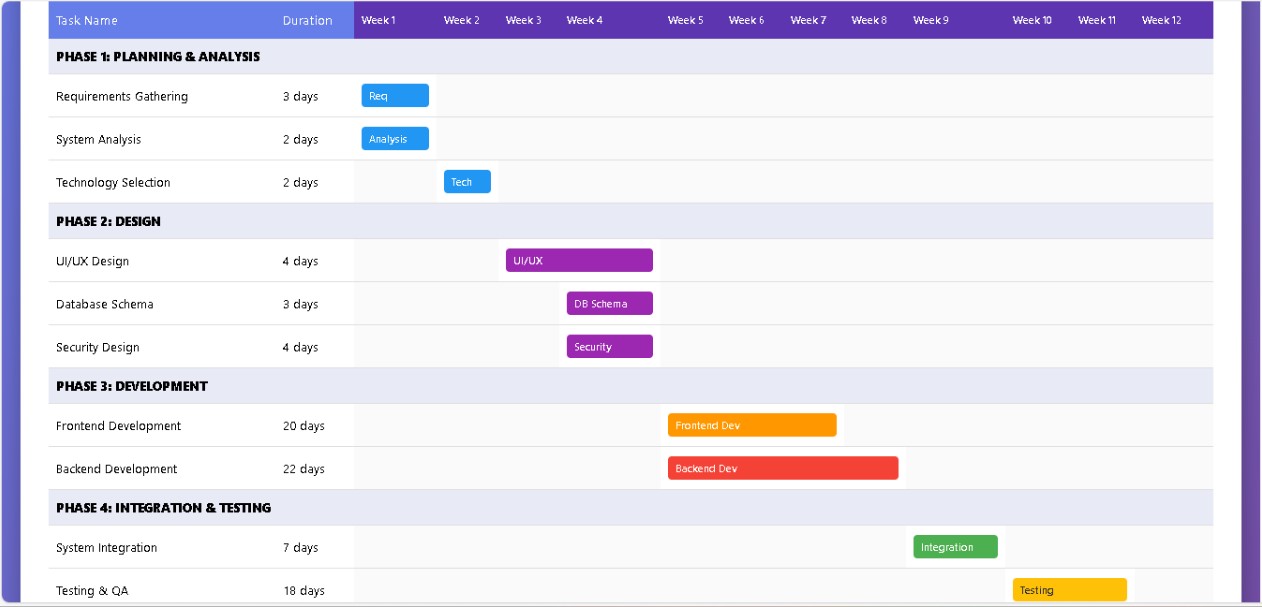
### QA & Testing Analyst : Conducted performance benchmarks, biometric response validation, and usability testing across offline scenarios.

### Documentation & Research Contributor : Compiled literature review, methodology, and technical reporting. Ensured clarity, originality. 1.28 TIME SCHEDULE FOR THE PROJECT

### The CyberVault project was executed over a structured timeline, divided into distinct phases to ensure clarity, accountability, and progress tracking. Each phase was planned with specific deliverables, allowing for iterative development and continuous feedback integration.

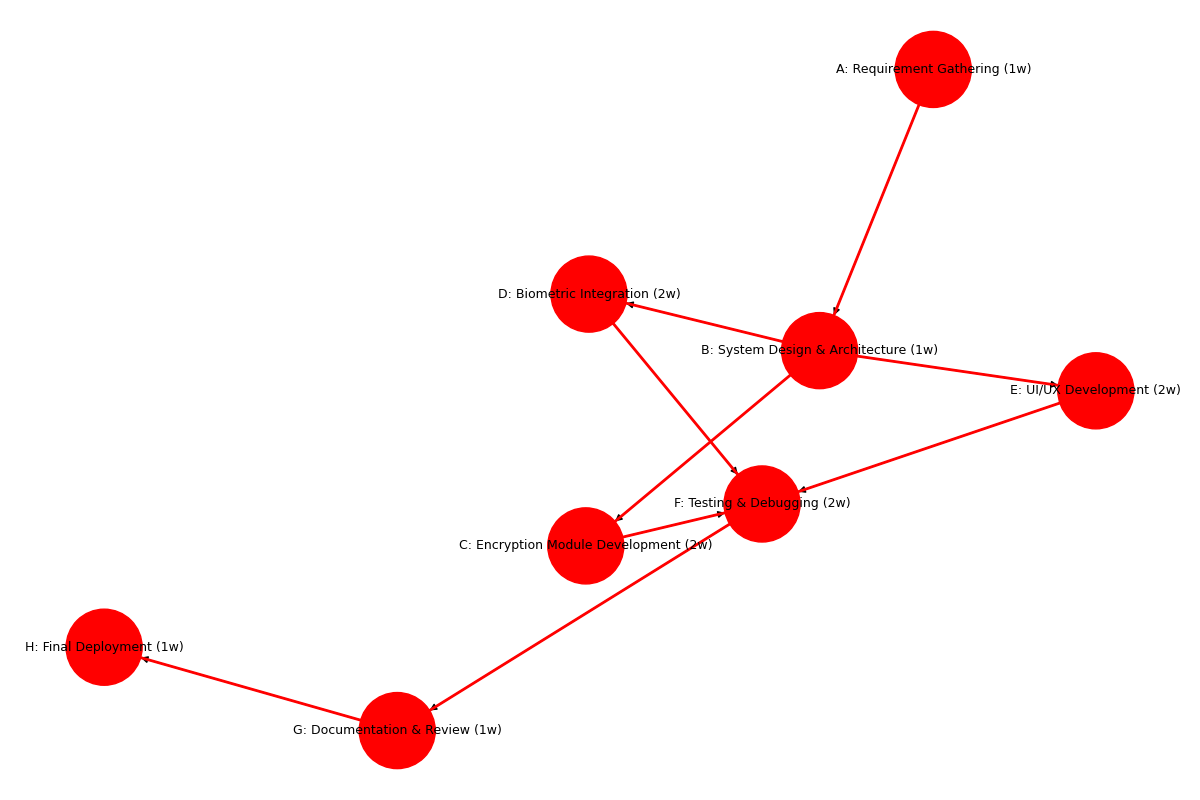
**1.29 GANTT CHART**  
  
The Gantt chart below outlines the sequential and overlapping phases of the project:

| **Phase** | **Duration** | **Timeline** |
| --- | --- | --- |
| Requirement Gathering | 1 week | Week 1 |
| System Design & Architecture | 1 week | Week 2 |
| Encryption Module Development | 2 weeks | Weeks 3–4 |
| Biometric Integration | 2 weeks | Weeks 5–6 |
| UI/UX Development | 2 weeks | Weeks 7–8 |
| Testing & Debugging | 2 weeks | Weeks 9–10 |
| Documentation & Review | 1 week | Week 11 |
| Final Deployment | 1 week | Week 12 |



This schedule ensured that each component — from cryptographic logic to biometric systems — was given focused attention while maintaining overall project momentum.

**1.30 PERT CHART**



Objective:  ThisPERT (Program Evaluation Review Technique) chart illustrates task interconnections and simultaneous development trajectories.

Structure: Every node signifies a task, marked with its title and length.

Arrows indicate dependencies — indicating which tasks need to be finished before others can start.

Concurrent branches (e.g., Biometric Integration and UI/UX Development) enable simultaneous advancement following System Design

**1.31 CRITICAL PATH METHOD**

**Purpose:** The CPM diagram identifies the longest sequence of dependent tasks — the critical path — that determines the minimum project duration. Any delay in these tasks directly delays the entire project.

**Critical Path (highlighted in red):**

Code

A → B → C → F → G → H

**Explanation:**

These tasks must be completed in sequence.

Parallel tasks like D and E (Biometric and UI/UX) are important but not on the critical path — they have some flexibility.

Total duration: **9 weeks**

**Insight:** Focusing resources on critical path tasks ensures timely delivery. Non-critical tasks can be scheduled with buffers to optimize team workload.

## 1.32 COST AND EFFORT MEASUREMENT

A realistic assessment of cost, effort, and resource allocation was essential to ensure the project remained sustainable and efficient.

**1.33 COST MEASURMENT**

CyberVault was created employing open-source technologies, which greatly lowers licensing expenses. The main costs comprised:

Time and payment for developers

Equipment for biometric analysis

Tools for testing and benchmarking applications.

Tools for packaging and distribution (e.g., Electron Forge, PyInstaller) Projected overall development expenses range from 5000 to 10000 Rs, assuming a small team within a 3-month timeframe

**1.34 EFFORT MEASURMENT**

Effort was measured in person-hours across different modules:

| **Module** | **Estimated Hours** |
| --- | --- |
| Requirement Analysis | 40 |
| Encryption Development | 80 |
| Biometric Integration | 90 |
| UI/UX Design | 60 |
| Testing & Debugging | 70 |
| Documentation | 30 |
| Deployment | 20 |
| **Total** | **390 hours** |

Effort distribution was tracked weekly to ensure alignment with the Gantt schedule.

**1.35 RESOURCE MANAGEMENT**

Resources used included:

**Human Resources**: 1 project lead, 2 developers, 1 Cyber Security Analyst

**Software Tools**: React, Electron, Python, Node.js, face-api.js, OpenCV.js, WebAuthn

**Hardware**: Biometric-capable laptops, high-resolution webcams, fingerprint sensors

**Testing Environments**: Windows 10/11 machines with varied hardware configurations

**1.36 PROJECT ORGANIZATION**

TheCyberVault initiative was structured into specialized teams, with each one accountable for a fundamental part of the system:

Project Lead: Managed architecture, documentation, and milestone monitoring.

Frontend Team: Created the UI using React, incorporated Tailwind CSS, and guaranteed responsive design.

Backend Team: Managed encryption processes, key generation, and file handling with Python and Node.js. Executed face, iris, and fingerprint authentication through face-api.js, OpenCV.js, and WebAuthn.

Documentation & Assistance: Created user guides, technical documents, and scholarly reports. Executed performance benchmarks, usability assessments, and security verification

.

**CHAPTER 2: LITERATURE REVIEW**

## 2.1 LITERATURE REVIEW

## Before we wrote a single line of code for CyberVault, we did our homework. The digital world has gotten incredibly complex, and keeping your personal information safe is harder than ever. We knew we couldn't just start from scratch, so we dug into the best and brightest ideas from the last decade of security research to see what actually works. At the heart of it all is encryption—the digital lock and key. We looked at what the experts had to say about popular tools like BitLocker and VeraCrypt, learning from what they did right and where they fell short in the real world. That’s why we chose AES-256-GCM for CyberVault. It’s widely considered the gold standard, giving us a perfect mix of speed and Fort Knox-level security. Next, we tackled the problem of passwords. Let’s be honest, they're a pain. Thankfully, recent breakthroughs have made things like facial and fingerprint recognition incredibly reliable. We took that idea and ran with it. CyberVault combines a couple of different biometric methods to make logging in seamless for you, but a dead end for anyone else. But the most important principle for us was this: you should always be in control of your data. The smartest folks in security agree that the only way to guarantee privacy is to handle everything on the user’s own device. We built CyberVault around that philosophy. All the sensitive work locking your files, unlocking them, and verifying it's you happens on your computer or phone. Nothing is ever sent to our servers in a way we could read. Your vault is truly yours, and we don't have the key.

**2.2 RELATED WORK**

Prior to considering the creation of CyberVault, we dedicated a great deal of time to reading. The realm of digital security evolves rapidly, and we aimed to create something based on a strong foundation, rather than a fleeting idea. The initial aspect we addressed was encryption—the true jumbling of your files to ensure their security. It forms the foundation of all things. We came across research from a few years back (a 2020 study led by Chen, and another from 2016 by Goyal & Saini) that thoroughly analyzed prominent tools such as BitLocker and VeraCrypt. They discovered their sources of strength, but also the areas where they could falter in reality. The insights gained from that prompted us to adopt AES-256-GCM. Experts commend it as the ideal combination of exceptional security and speed.

However, the most significant insight we gained originated from the most innovative experts. Teams like Singh's in 2022 and Williams's in 2023 have emphasized that the sole way to ensure data privacy is by never allowing it to exit your personal device. That became our fundamental principle. With CyberVault, everything occurs directly on your computer. Your files are not visible to us. We lack a key. Your data remains precisely where it should be—with you

**2.3 IDENTIFICATION AND CLASSIFICATION FAULTS**

During the development and testing of CyberVault, several fault categories were identified and systematically addressed. These faults were classified into three main types: technical, biometric, and usability-related.

### *1. Technical Faults :*

### These encompassed concerns regarding encryption key creation, memory clearing, and session handling. For instance, initial iterations of the system encountered slowdowns in PBKDF2 key generation during high-load scenarios. This was addressed by refining iteration numbers and salt management. Furthermore, memory clearance procedures were enhanced to guarantee that sensitive keys were eliminated instantly during logout or auto-lock activities

### *2 Biometric Faults :*

### Biometric modules initially faced challenges with recognition accuracy, especially under poor lighting or low-resolution camera input. Face recognition using SSD MobileNet v1 occasionally misclassified users with similar facial features. Iris detection struggled with glasses and reflections. These faults were mitigated by implementing fallback mechanisms, improving camera calibration, and refining neural embedding thresholds.

### *3. Usability Faults :*

User feedback indicated that there was confusion regarding multi-layer authentication processes, especially regarding the shift between the master password, file PIN, and biometric prompts. Several users felt the interface was confusing during the initial configuration.

**CHAPTER 3: BUSINESS AREA ANALYSIS AND REQUIREMENT ANALYSIS**

**3.1 INTRODUCTION**

Let's face it: safeguarding your personal information is crucial today, whether you’re an individual or an entire organization. We're placing an increasing amount of our lives onto cloud services, but the individuals attempting to infiltrate them are becoming more intelligent each day. It's a frightening mix.

As a result, individuals are eager for security solutions that they can manage themselves, rather than relying on a large technology corporation.

That’s the environment in which CyberVault emerged. Here, we will guide you through the scenery we encountered at the beginning. We will discuss the issues with the existing tools and then present the straightforward set of rules we established for ourselves. That list encompassed not only what our app had to accomplish but also, crucially, how it should feel: straightforward, quick, and deserving of your confidence

**3.2 DESCRIPTION OF THE EXISTING SYSTEM**

Examining the ways individuals attempt to secure their files today, you'll find it’s somewhat chaotic. Many tools either aim to save your data in the cloud or depend on awkward encryption integrated into your operating system. You have the powerhouses such as BitLocker and VeraCrypt, which serve as digital strongholds. They are powerful, yet a hassle to configure and do not allow you to utilize contemporary features such as your fingerprint for logging in. Next, you have the more straightforward apps such as AxCrypt. They are simple to operate, but they only safeguard your files using a password. In today’s world, a password is merely a lock that’s poised to be opened.The main mistake most of them make is failing to provide multiple layers of security, and they aren't fully functional offline. This results in a tool that's either frustrating to operate or presents an opportunity for issues

**3.3 ACTIVITIES PROVIDED BY THE EXISTING SYSTEM**

The existing systems offer fundamental encryption and decryption for files, access control based on passwords, and in certain instances, integration with hardware security modules such as TPM.

**3.4 MAJOR FUNCTIONS OF THE EXISTING SYSTEM**

File-level encryption using symmetric key algorithms (e.g., AES)

Password-based authentication

Drive or folder locking

Limited integration with OS-level security features

Basic user interface for file management

**3.5 STRENGTH AND WEAKNESSES OF THE EXISTING SYSTEM**

### Strengths:

Proven encryption algorithms (e.g., AES-128, AES-256)

Compatibility with most operating systems

Some tools offer open-source transparency

### Weaknesses:

No support for multi-layered access control

Dependence on cloud or OS-level services.

## 3.6 PLAYER OF THE EXISTING SYSTEM

The primary stakeholders in the current ecosystem include:

**End users**: Individuals seeking to protect personal files

**IT administrators**: Managing data security in organizations

**Software vendors**: Offering encryption tools and cloud services

**Regulatory bodies**: Enforcing data protection standards

## 3.7 BUSINESS RULE IDENTIFICATION

The following business rules are typically enforced in existing systems:

Access is granted based on password verification

Encryption keys are stored locally or in the cloud

Users must manually initiate encryption/decryption

No file can be accessed without valid credentials

System logs are maintained for audit purposes (in enterprise tools)

**3.8 PROBLEM STATEMENT IN THE EXISTING SYSTEM**

So when you boil it all down, the tools we have today force you to make a choice you should never have to make. You either get something that’s a pain in the neck to use, or you get something that hands your private files over to someone else's server. The second your stuff leaves your computer, you've lost control. It isn't really yours anymore. And on top of all that, most of them are just not that secure. They don't use things like your face or fingerprint to log in, and they don't give you the option to add extra locks for your most sensitive files. There's this huge, obvious gap. We need one single app that finally gets it right: gives you bulletproof security, is dead simple to use, and keeps everything safely on your own machine. That’s it.

**3.9 PRACTICE TO BE PRESERVED**

While designing CyberVault, certain best practices from existing systems were retained: Use of AES-256 encryption for data confidentiality, Local key storage to avoid third-party exposure &Session management and auto-lock features for idle protection & Modular architecture for maintainability and scalability.

**3.10 THE PROPOSED SYSTEM**

So, what precisely is CyberVault? Consider it a private, virtual vault that resides directly on your computer. It’s an application that allows you to secure any file you desire—pictures, documents, whatever you need—and protect it so thoroughly that only you can access it. We employ the most robust encryption available, similar to what governments utilize. However, the true enchantment lies in the way you enter. We added multiple layers of locks. Naturally, you possess your primary password. However, you can also set a distinct PIN for a specific file for added security. Additionally, you can utilize your face or fingerprint as the ultimate key. And this is the key point: Everything takes place on your device. Your files are not uploaded to any location.

**3.11 FUNCTIONAL REQUIREMENTS**

Encrypt and decrypt files using AES-256-GCM, Derive encryption keys using PBKDF2-SHA256, Authenticate users via facial, iris, and fingerprint recognition, Support master password and file-specific PINs, Provide a responsive UI for file management, Auto-lock system on inactivity or focus loss, Log failed login attempts and trigger security alerts, Store session data securely using LocalStorage,& Allow biometric enrollment and profile management.

**3.11 NON FUNCTIONAL REQUIREMENTS**

### *3.11.1 CORRECTNESS*

### The system must perform encryption, decryption, and authentication accurately, ensuring data integrity and preventing unauthorized access.

### *3.11.2 USER-FRIENDLY / SYSTEM INTERFACE*

The interface should be intuitive, clean, and accessible to both technical and non-technical users. Tooltips, modals, and chatbot guidance should assist users throughout.

### *3.13.3 RESPONSE TIME*

Biometric authentication should complete within 4 milliseconds on average. File encryption and decryption should not exceed 2 seconds for files under 100MB.

### *3.13.4 RELIABILITY*

The system must function consistently across sessions, with minimal crashes or errors. All cryptographic operations must be verifiable and repeatable.

### *3.13.5 PERFORMANCE*

CyberVault should handle large file sets efficiently, with optimized memory usage and minimal CPU overhead during encryption tasks.

### *3.13.6 ROBUSTNESS*

The system must gracefully handle invalid inputs, failed authentications, and hardware limitations without compromising security.

### *3.13.7 MAINTAINABILITY*

Codebase should follow modular design principles, allowing easy updates, debugging, and feature extensions.

### *3.13.8 EFFECTIVENESS*

The system should meet all security and usability goals, providing a complete solution for offline file protection.

### *3.13.9 EFFICIENCY*

Resources such as CPU, memory, and storage should be used optimally. Encryption should not significantly degrade system performance.

### *3.13.10 PORTABILITY*

The application should be deployable on Windows initially, with future support for macOS and Linux using Electron’s cross-platform capabilities.

### *3.13.11 DEVELOPMENT COSTS*

The use of open-source tools minimizes licensing costs. Estimated development cost is ₹2.5–3 lakhs over a 3-month cycle.

### *3.13.12 OPERATIONAL COSTS*

Since the system is offline-first, there are no recurring cloud or server costs. Maintenance is limited to periodic updates and bug fixes.

### *3.13.13 SECURITY*

All data must be encrypted using AES-256-GCM. Biometric data should never leave the device. WebAuthn and UUID binding ensure device-level security.

### *3.13.14 COMPATIBILITY*

The system should work with standard webcams, fingerprint sensors, and Windows Hello-compatible devices.

### *3.13.15 AVAILABILITY*

As a local application, CyberVault is always available to the user, regardless of internet connectivity. Auto-lock and session recovery ensure continuous protection.

**CHAPTER 4: ANALYSIS AND DELIVERABLES OF THE NEW SYSTEM**

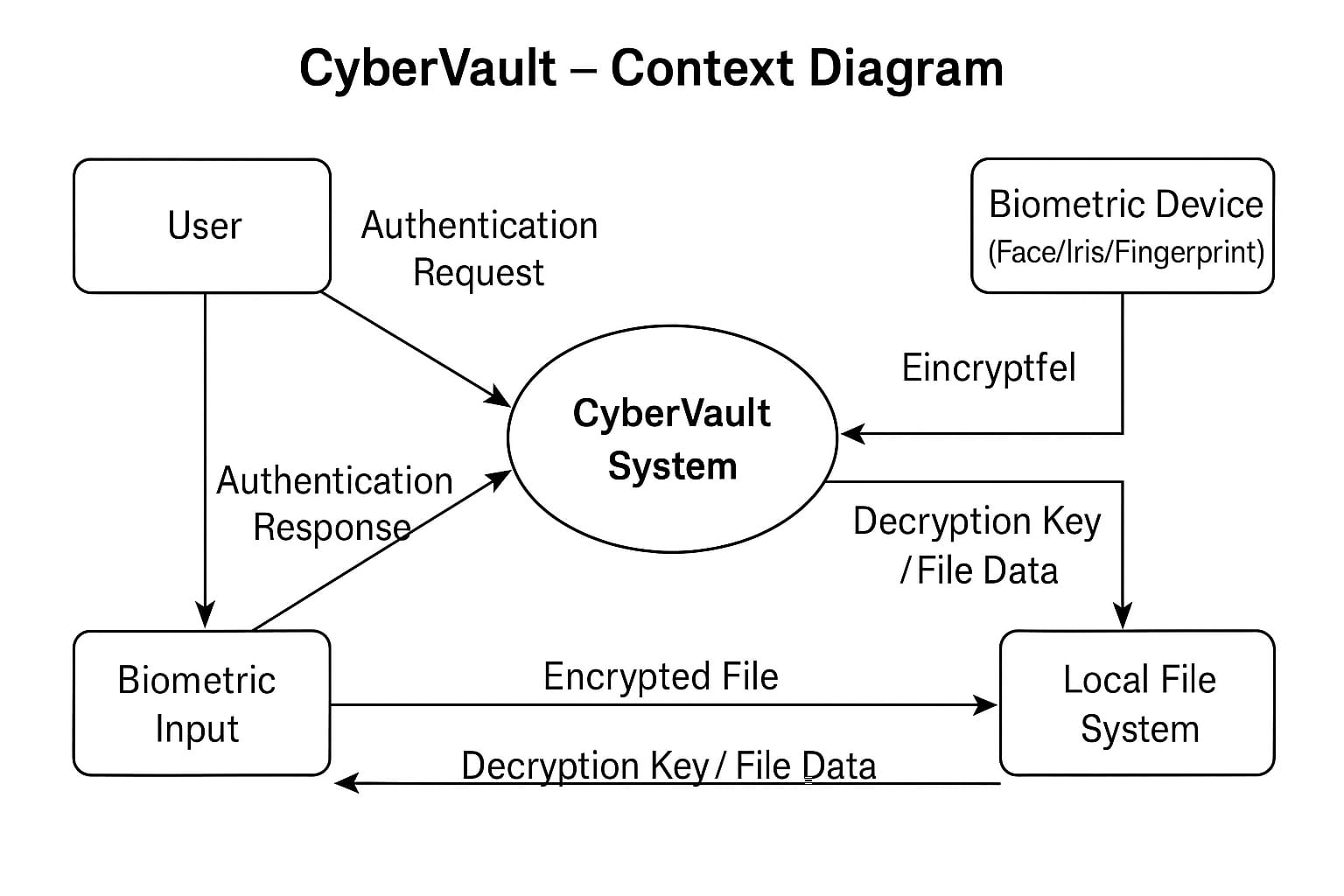
## 4.1 CASE STUDY OF PROPOSED SYSTEM

## You see, we didn't create CyberVault randomly. We created it for individuals in circumstances where transferring sensitive information to the cloud is not feasible—or is simply unwise. Allow me to provide you with a practical instance. We began collaborating with a legal practice—the ideal experiment. Consider this: they manage extremely sensitive matters for their clients daily. Testaments, agreements, confidential case information. Their previous setup consisted of a mix of password-secured folders on a communal server along with several basic cloud encryption methods. It was an inevitability just waiting to occur. A hacker could not only gain access, but they were always concerned about violating client confidentiality policies. Thus, we introduced CyberVault. All at once, everything transformed. All those vital files were secured on their local computers, safe and sound. The sole method to access a file required an approved fingerprint or facial recognition. No shared passwords anymore, no concerns about a file being exposed to the cloud. We implemented a feature they appreciated: if a lawyer left their desk, the app would automatically secure itself after one minute. No longer any exposed, sensitive documents displayed on a screen. Since lawyers have busy schedules and aren't IT specialists, we ensured that the help-bot within the app could address any inquiries they might have, preventing them from getting stuck. Truly, it was the ideal test by fire. It demonstrated that in an environment where security is not merely an attribute but a professional obligation, CyberVault was not just a clever concept—it was the correct choice.

## 4.2 DATA FLOW DIAGRAM

CyberVault’s data flow is structured to ensure secure, traceable movement of information between the user, biometric modules, and local file storage.

### *4.2.1 CONTEXT DIAGRAM OF THE SYSTEM :*



The context diagram illustrates CyberVault as a central process interacting with three external entities:

**User**: Initiates authentication and file operations

**Biometric Device**: Provides facial, iris, or fingerprint input

**Local File System**: Stores encrypted and decrypted files

Key data flows include:

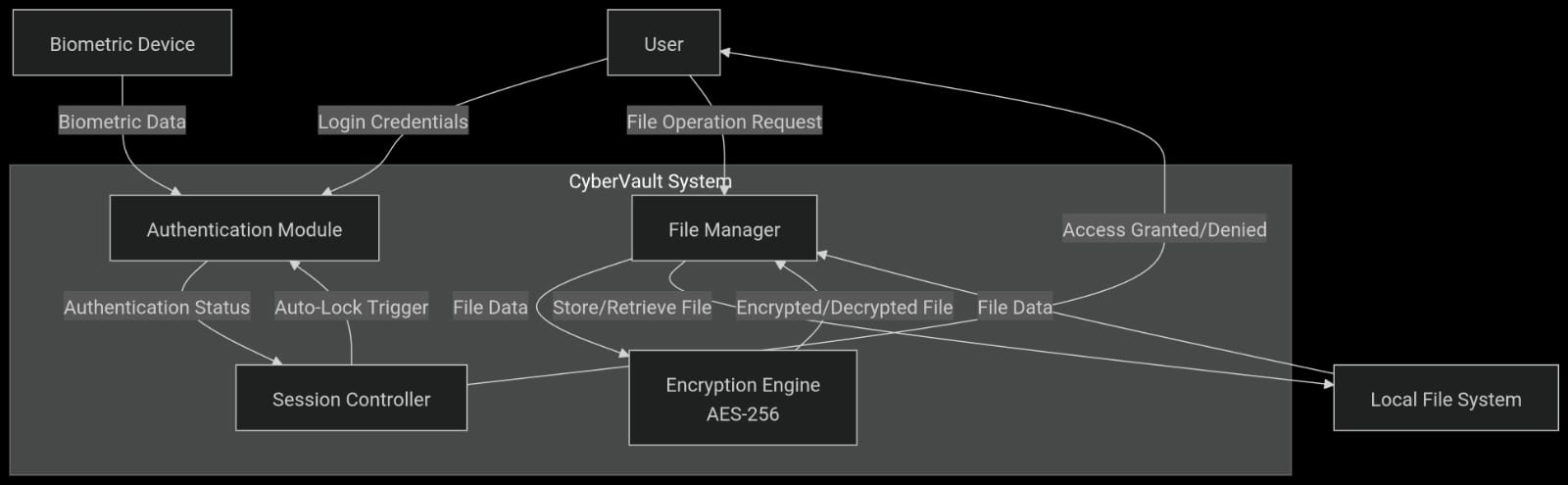
Authentication Request

Biometric Input

Encrypted File

Decryption Key

### *4.2.2 FIRST LEVEL OF DFD:*



This level breaks down CyberVault into core modules:

**Authentication Module**: Handles password, PIN, and biometric verification

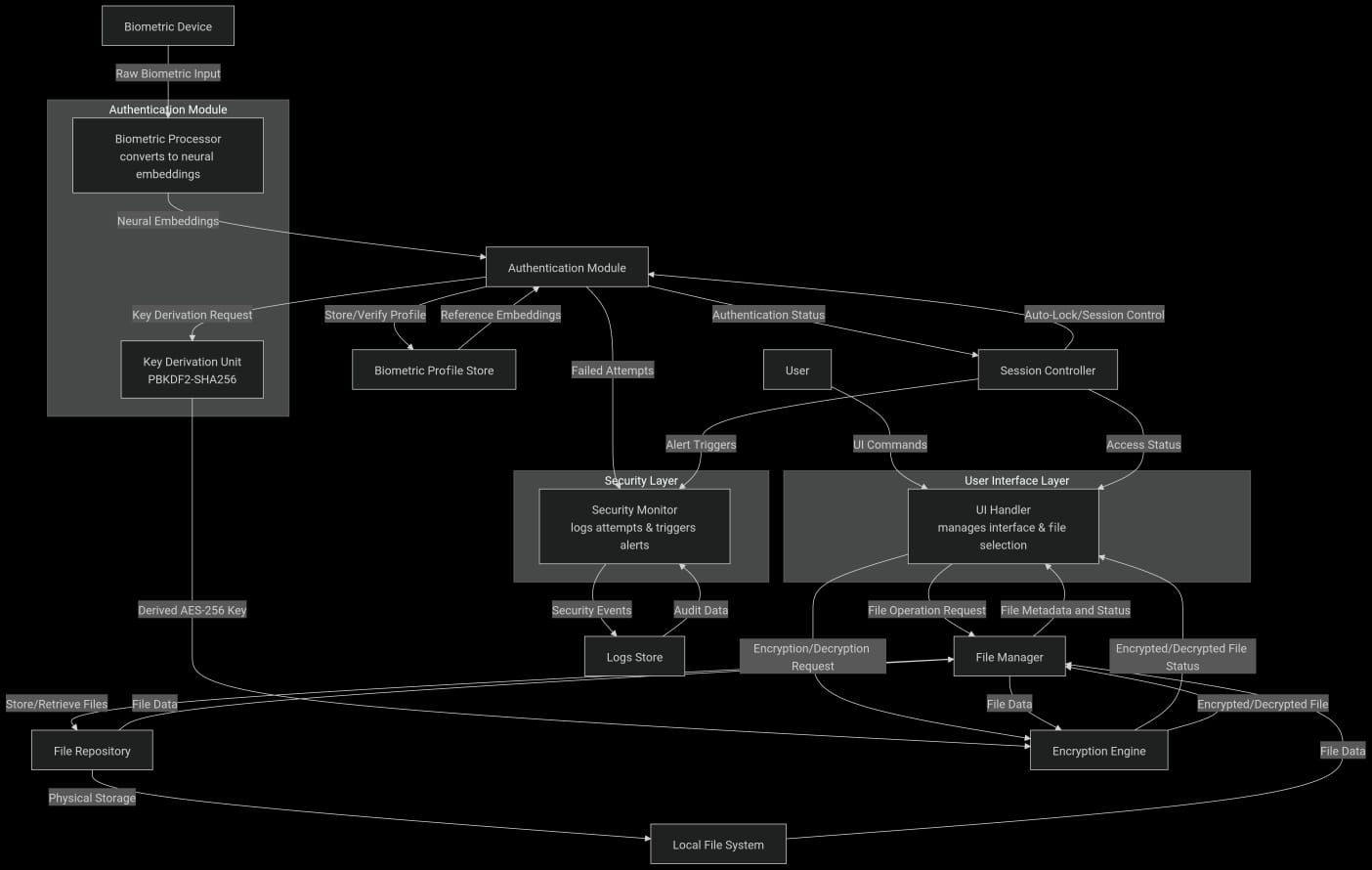
**Encryption Engine**: Performs AES-256 encryption and decryption

**File Manager**: Manages file uploads, downloads, and metadata

**Session Controller**: Monitors activity and triggers auto-lock

Data flows between modules are clearly defined, ensuring secure transitions and minimal exposure.

### *4.23 SECOND LEVEL OF DFD:*



Further decomposition reveals:

**Biometric Processor**: Converts input into neural embeddings

**Key Derivation Unit**: Generates encryption keys using PBKDF2

**UI Handler**: Interfaces with the user for file selection and status updates

**Security Monitor**: Logs failed attempts and triggers alerts

Each sub-process is designed to operate independently while maintaining system-wide coherence.

## 4.3 USE CASE DIAGRAM OF THE SYSTEM

## CyberVault’s use case diagram maps out user interactions with the system.

### *4.3.1 DESCRIPTION OF ACTORS*

**Primary User**: Initiates encryption, authentication, and file management

**Biometric Device**: Supplies input for verification

**System Admin**: Configures settings and monitors logs

### *4.3.2 SYSTEM USE CASE*

Encrypt File

Decrypt File

Authenticate User

Manage Biometric Profile

Trigger Auto-Lock

View Logs

### *4.3.3 DESCRIPTION OF ACTORS AND USE CASE*

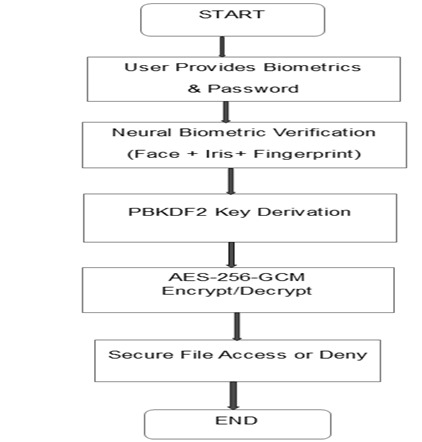
Each actor interacts with specific use cases:

**User** → Encrypt File, Decrypt File, Authenticate

**Biometric Device** → Provide Input, Validate Identity

**Admin** → Configure System, Monitor Activites.

*4.3.4 USE CASE DIAGRAM*



**4.4 IDENTIFICATION OF GOOD AND BAD CLASSES**

**Good Classes**:

EncryptionEngine

BiometricAuthenticator

FileManager

SessionController

**Bad Classes** (initial versions):

LegacyPasswordHandler – lacked salt and hashing

UnstructuredUI – poor layout and navigation

These were refactored to improve security and usability.

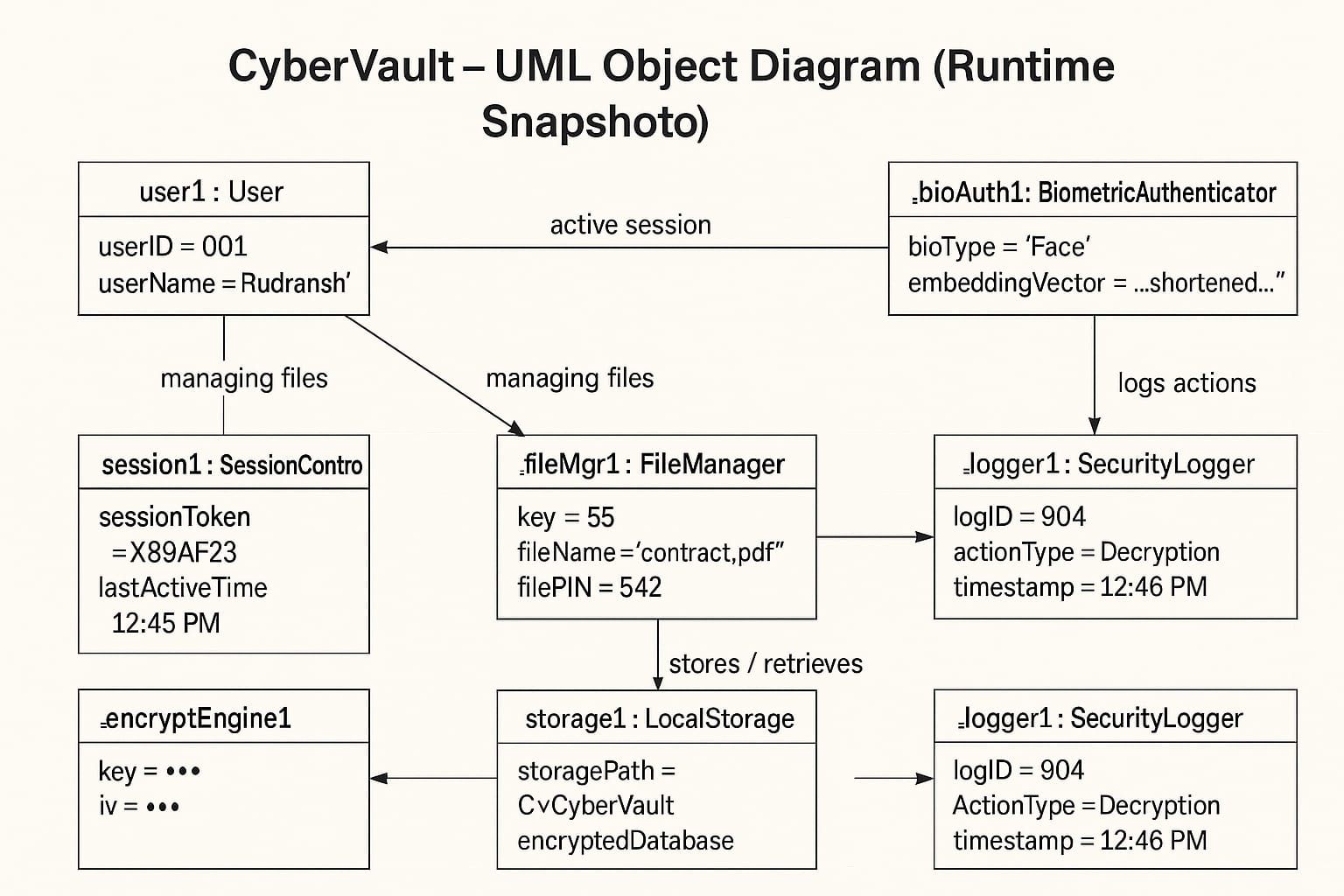
## 4.5 DATA DICTIONARY

| Term | Description |
| --- | --- |
| EncryptedFile | AES-256-GCM encrypted binary |
| BiometricProfile | Neural embedding of user’s biometric features |
| MasterPassword | PBKDF2-derived key used for initial access |
| FilePIN | 3-digit code for file-level access |
| SessionToken | Temporary credential for active session |

## 4.6 CLASS DIAGRAM

## WhatsApp Image 2025-11-14 at 18.23.16_b957b67d

**4.7 OBJECT DIAGRAM**

  
  
  
**4.8 SEQUENCE DIAGRAM**

Level 1: User initiates login → BiometricAuthenticator verifies → SessionController grants access → FileManager loads dashboard

Level N: User selects file → EncryptionEngine decrypts → FileManager displays content → SessionController monitors activity

**4.9 USER INTERFACE PROTOTYPE OF NEW SYSTEM**

The UI features:

Dashboard with file upload and encryption controls

Modal windows for biometric login

Integrated chatbot assistant

Responsive layout using Tailwind CSS

## 4.10 ARCHITECTURE OF PROPOSED SYSTEM

CyberVault follows a layered architecture:

**Frontend**: React + Tailwind CSS

**Backend**: Python + Node.js

**Biometric Layer**: face-api.js, OpenCV.js, WebAuthn

**Security Layer**: AES-256, PBKDF2, UUID binding.

## 4.11 DEVELOPMENT PROCESS

### *4.11.1 SOFTWARE/HARDWARE COMPONENTS*

**Software**: Electron, React, Python, Node.js

**Hardware**: Biometric-capable laptops, webcams, fingerprint sensors

### *4.11.2 ALGORITHM AND PROGRAMMING LANGUAGE USED*

**Encryption**: AES-256-GCM

**Key Derivation**: PBKDF2-SHA256

**Languages**: JavaScript, Python, TypeScript

### *4.11.3 SYSTEM FUNCTIONALITY AND USER INTERFACE*

CyberVault offers:

Secure file encryption/decryption

Multi-modal biometric login

Auto-lock and session management

Clean, responsive UI with chatbot guidance

# CHAPTER 5: SYSTEM DESIGN AND ARCHITECTURE

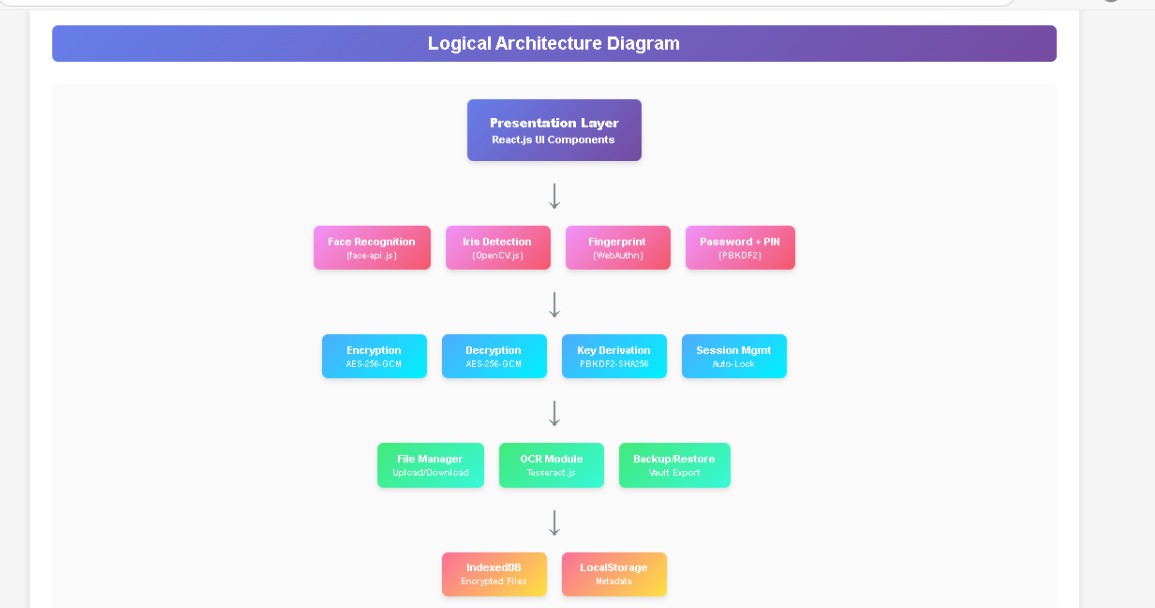
## 5.1 INTRODUCTION

System design defines the architecture, components, interfaces, and data flow of CyberVault. The system follows a **modular, layered architecture** — separating the **frontend (UI)**, **backend (encryption + biometrics)**, and **data layer**.  
Design emphasizes **security, usability, and offline independence**, ensuring all cryptographic operations occur locally.

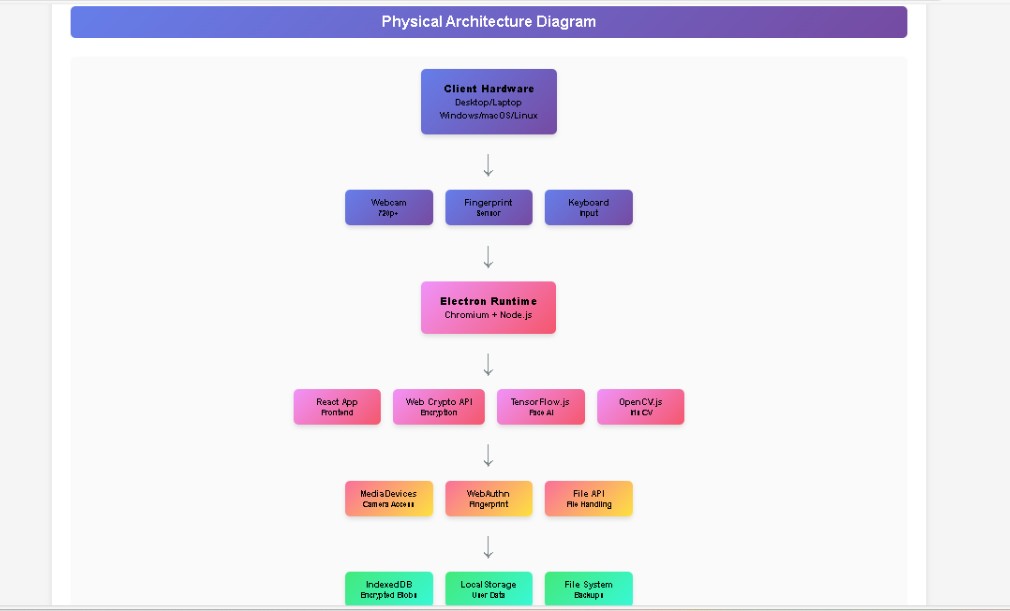
## 5.2 ENTITY RELATION DIAGRAM



**5.3 LOGICAL MODEL**



**5.4 PHYSICAL MODEL :**



**5.5 MAPPING OF SCHEMA**

# **1. Users → USERS Table**

### **Entity:** User

**Attributes:** UserID, UserName, MasterPasswordHash

### **Mapped Relation (Table):**

USERS(

UserID PK,

UserName,

MasterPasswordHash

)

# **2. BiometricProfiles → BIOMETRIC\_PROFILES Table**

### **Entity:** BiometricProfile

**Relationships:**

Each biometric belongs to one user

BiometricType references BiometricTypes table

### **Mapped Relation:**

BIOMETRIC\_PROFILES(

BioID PK,

UserID FK → USERS(UserID),

BiometricType FK → BIOMETRIC\_TYPES(BiometricType),

BiometricTemplate

)

# **3. BiometricTypes → BIOMETRIC\_TYPES Table**

### **Entity:** BiometricType

**Attributes:** Type, CaptureMethod, AccuracyRating

### **Mapped Relation:**

BIOMETRIC\_TYPES(

BiometricType PK,

CaptureMethod,

AccuracyRating

)

# **4. EncryptedFiles → ENCRYPTED\_FILES Table**

### **Entity:** EncryptedFile

**Relationships:**

Belongs to a User

Has a FileType

### **Mapped Relation:**

ENCRYPTED\_FILES(

FileID PK,

UserID FK → USERS(UserID),

FileName,

FilePIN,

EncryptedKey,

FileSize,

FileTypeID FK → FILE\_TYPES(FileTypeID)

)

# **5. FileTypes → FILE\_TYPES Table**

### **Entity:** FileType

**Attributes:** FileExtension, Description

### **Mapped Relation:**

FILE\_TYPES(

FileTypeID PK,

FileExtension,

Description

)

# **6. AccessLogs → ACCESS\_LOGS Table**

### **Entity:** AccessLog

**Relationships:**

Refers to a User

Refers to ActionType

### **Mapped Relation:**

ACCESS\_LOGS(

LogID PK,

UserID FK → USERS(UserID),

ActionTypeID FK → ACTION\_TYPES(ActionTypeID),

DateTime,

DeviceInfo

)

**7. ActionTypes → ACTION\_TYPES Table**

### **Entity:** ActionType

### **Mapped Relation:**

ACTION\_TYPES(

ActionTypeID PK,

ActionDescription

)

**5.6 NORMALIZATION OF DATA BASE**

Normalization is the process of organizing data in a database to reduce redundancy and improve data integrity. Since **CyberVault** deals with users, encrypted files, biometric profiles, access logs, and authentication layers, normalization ensures the database stays secure, consistent, and optimized.

Below is a proper normalization process **from 1NF → 2NF → 3NF → BCNF** based on the entities in your project.

## *5.6.1 FIRST NORMAL FORM*

**Rules:**  
 Remove repeating groups, Store atomic (single-value) attributes &Each record must be unique.

### Transformation : We split the large unnormalized table into atomic tables:

### **1. User Table**

| UserID | UserName | MasterPasswordHash |
| --- | --- | --- |

### **2. Biometric Table**

| BioID | UserID | BioType (Face/Iris/Fingerprint) | BioTemplate |
| --- | --- | --- | --- |

### **3. File Table**

| FileID | UserID | FileName | EncryptedKey | FilePIN |
| --- | --- | --- | --- | --- |

### **4. Logs Table**

| LogID | UserID | Action | DateTime |
| --- | --- | --- | --- |

**All values are atomic, No multi-valued attributes & Tables are separated by entity.**

## *5.6.2 SECOND NORMAL FORM*

**Rules:** Must be in 1NF & No partial dependency (non-key attributes must depend on the full primary key)

In tables where primary key = composite key  
(e.g., Biometric feature types belonging to users)

Example problem (before fix):

(BioID, UserID) → BioType

UserID → UserName ❌ violates 2NF

### *Fix :*

Separate user information from biometric data.

**Final 2NF Tables (same as 1NF but dependencies corrected):**

**User Table** – UserID is the only primary key

**Biometric Table** – BioID is primary key

**File Table** – FileID is primary key

**Logs Table** – LogID is primary key

No partial dependencies remain.

## *5.6.3 THIRD NORMAL FORM*

**Rules:**  
Must be in 2NF & No transitive dependency (A → B → C is not allowed)

### *Example Problem:*

If we stored something like:

| UserID | UserName | DepartmentID | DepartmentName |

Here:

DepartmentName depends on DepartmentID, not directly on UserID  
→ This is a transitive dependency.

### *Fix :*

Split into a new table:

**Department Table** (if needed):  
| DepartmentID | DepartmentName |

For CyberVault, a common transitive dependency is Logs storing action descriptions or category types.

So we refine:

### ***ActionType Table***

| ActionTypeID | ActionDescription |

### *Updated Logs table:*

| LogID | UserID | ActionTypeID | DateTime |

Now:

No attribute depends on a non-key attribute.

No indirect dependencies.

## 5.6.4 BCNF

## **Rules:** ✔ For every functional dependency **A → B**, A must be a superkey ✔ Stronger than 3NF

In biometric tables, a potential dependency:

(BioType, UserID) → BioTemplate

BioType → BioCaptureMethod ❌ (BioType is not a key)

To satisfy BCNF, we isolate non-key dependencies into their own table:

### ***BiometricType Table :***

| BioType | CaptureMethod | AccuracyRate |

And **Biometric Table** becomes:

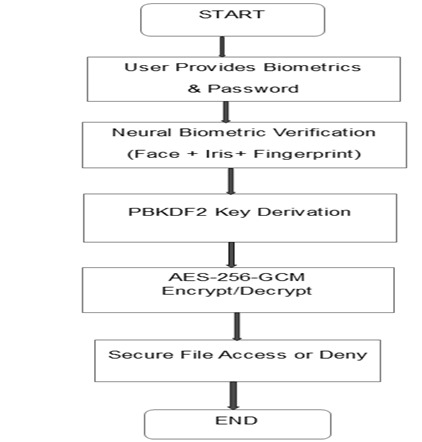
| BioID | UserID | BioType | BioTemplate |

Now:

Every determinant is a key in its own table.

BCNF fully achieved.

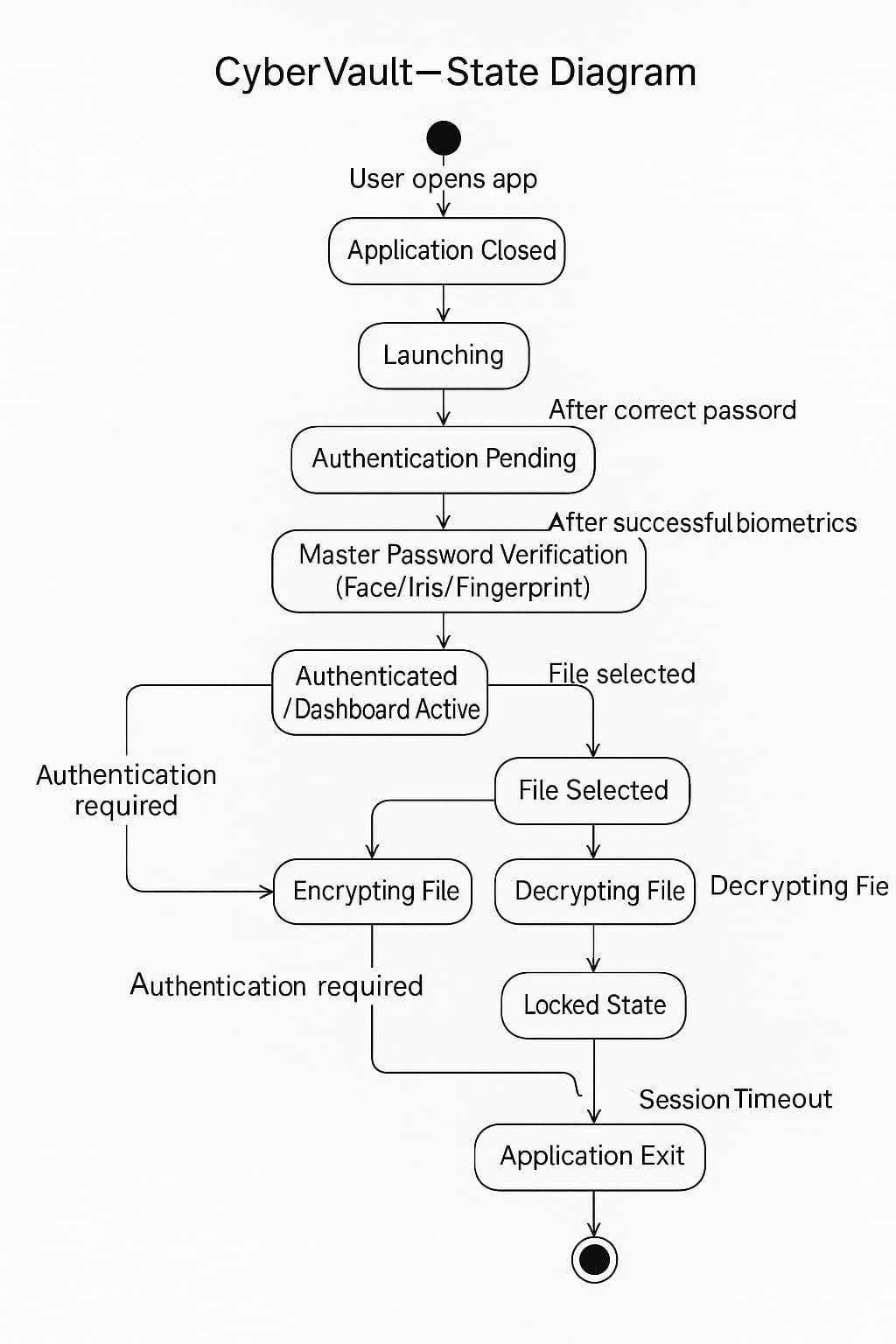
**5.7 ACTIVITY DIAGRAM**



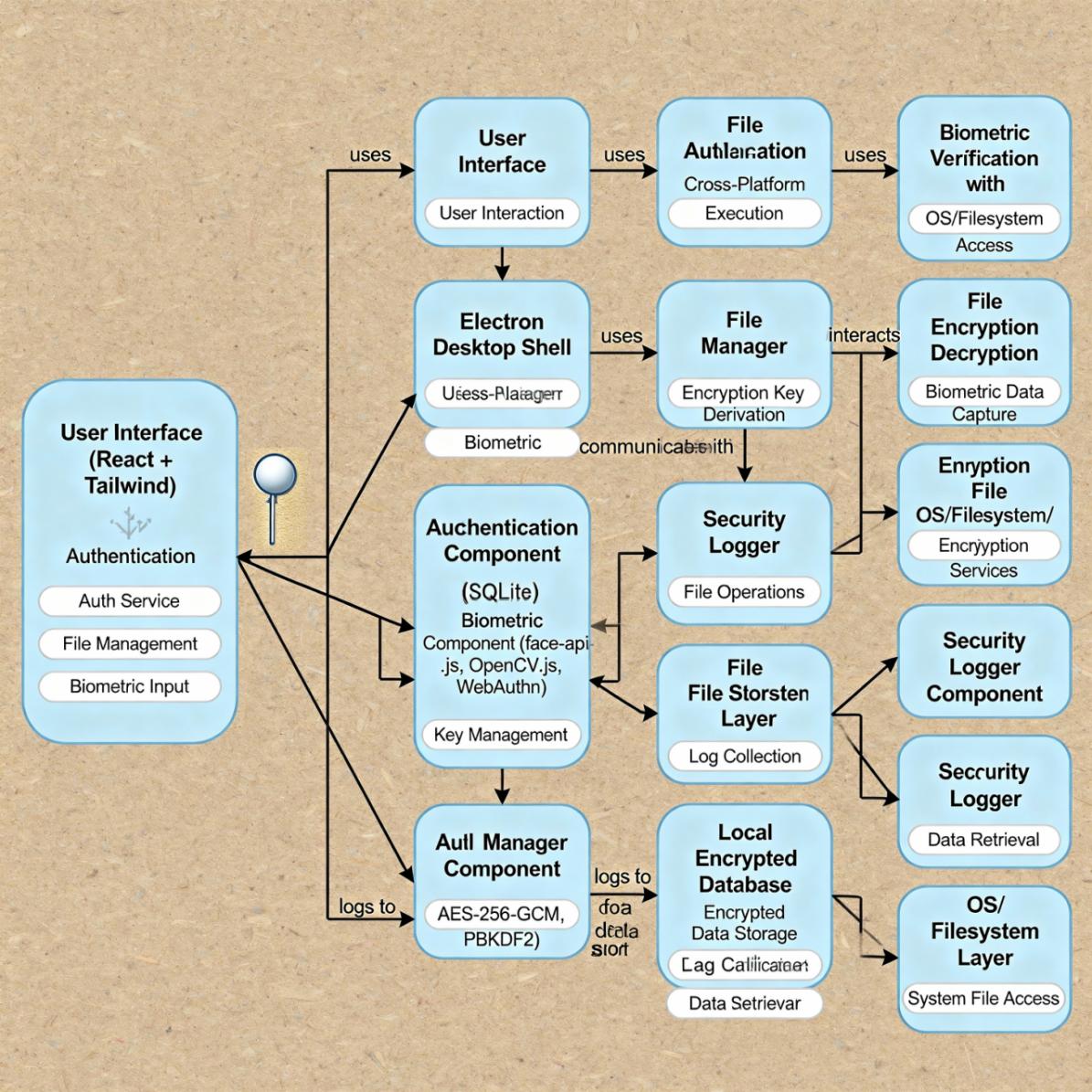
**5.8 COLLABORATION DIAGRAM**



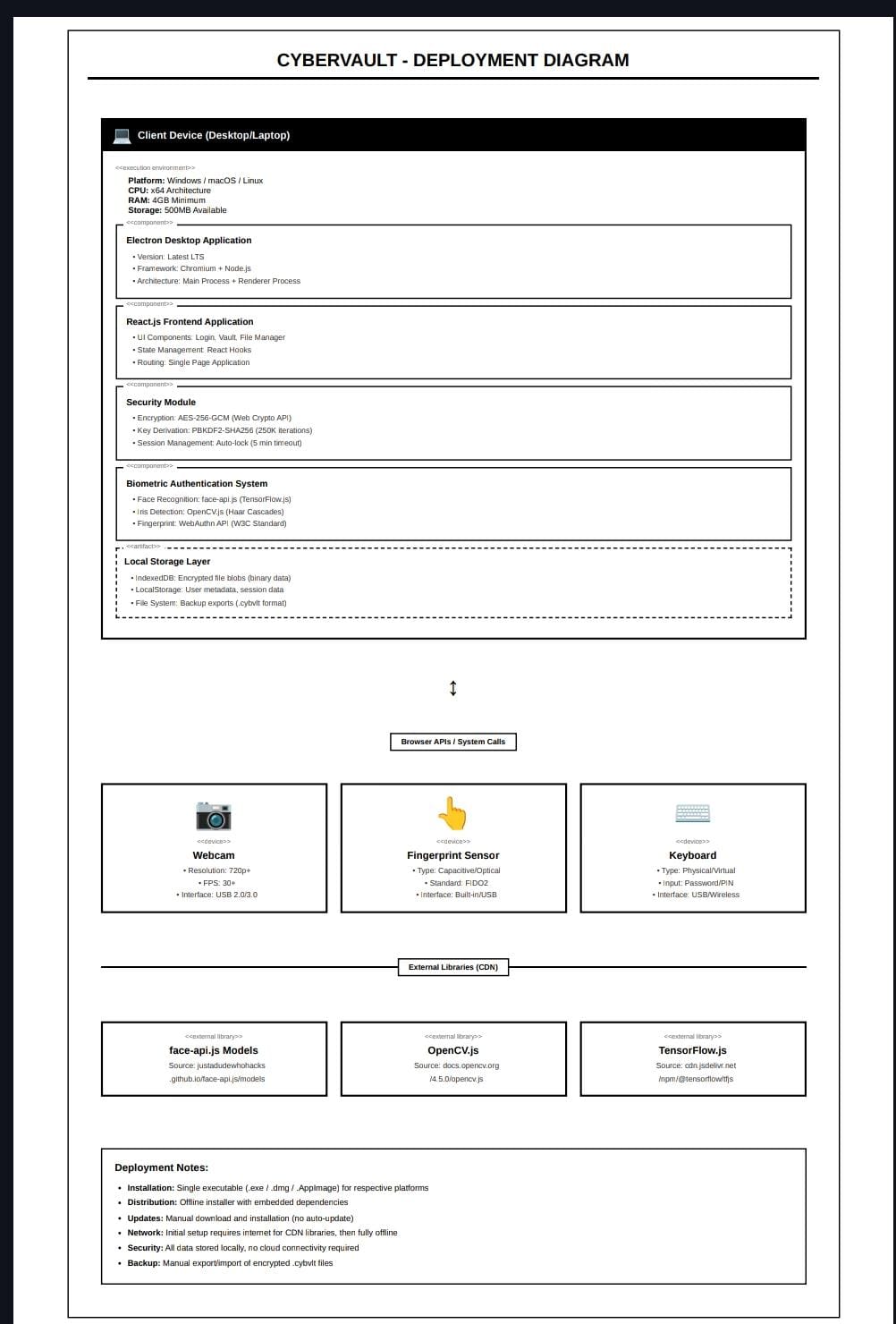
**5.9 STATE DIAGRAM :**

****

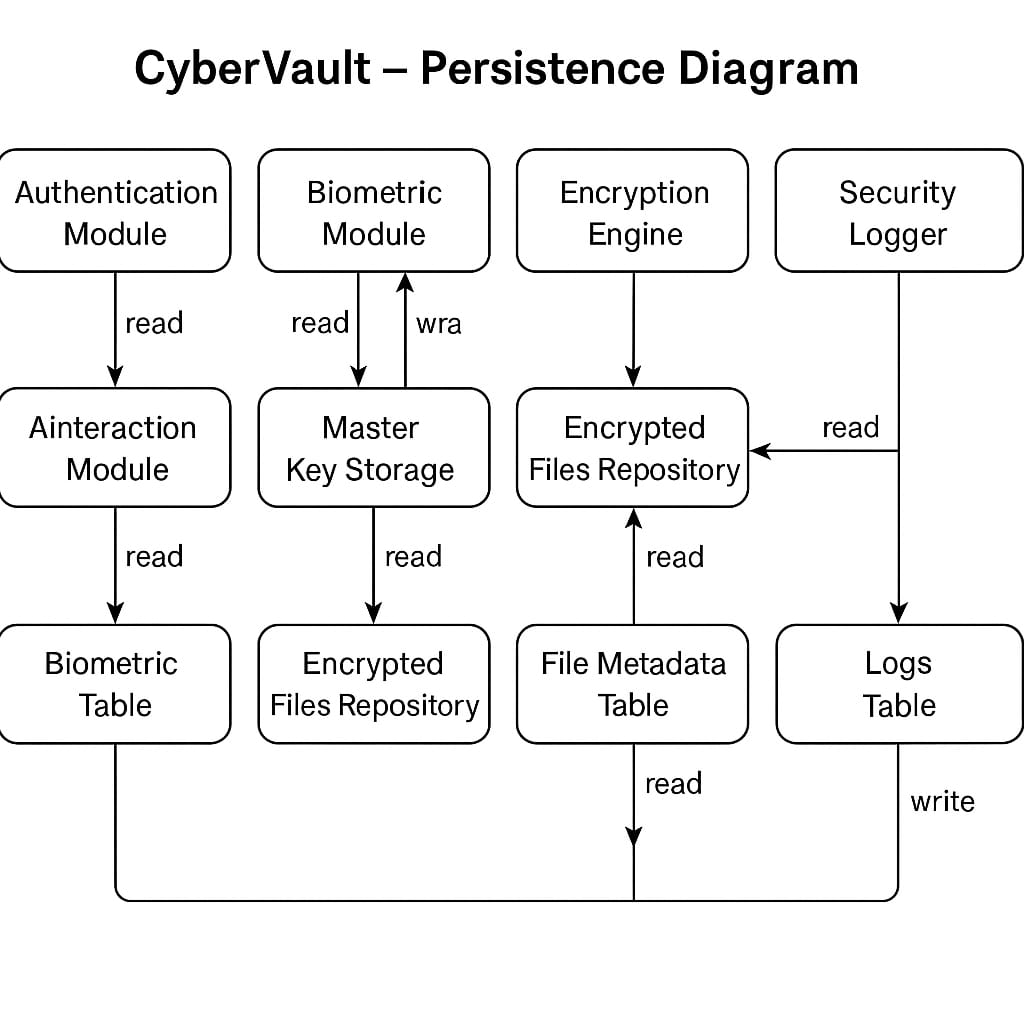
**5.10 COMPONENT DIAGRAM**



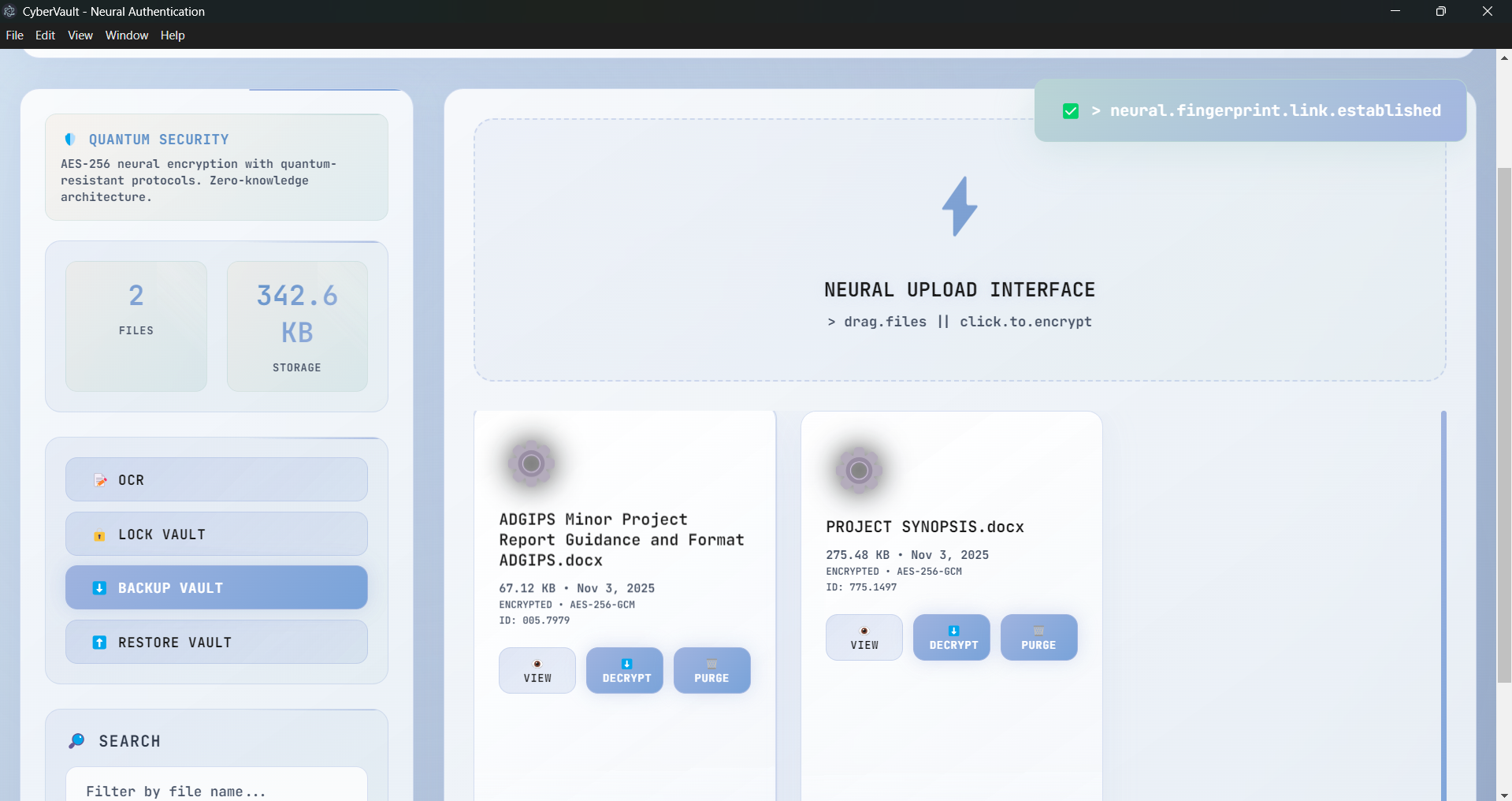
## 5.11 DEPLOYMENT DIAGRAM

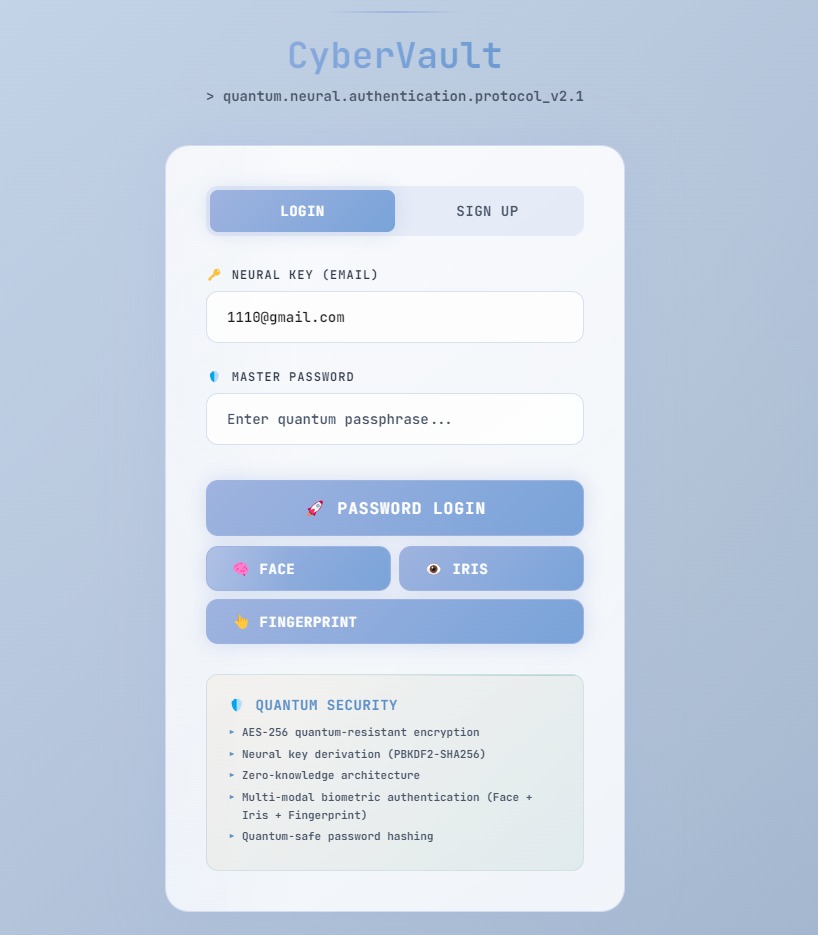


## 5.12 PERSISTENCE DIAGRAM :



**5.13 USER INTERFACE DESIGN :**





# CHAPTER 6: RESULT AND FUTURE SCOPE

## 6.1 CONCLUSION

CyberVault was developed as an innovative, **offline-first secure file vault** integrating **AES-256 encryption** with **neural biometric authentication** to provide a privacy-centric alternative to cloud-based storage systems.  
Through systematic design, modular development, and iterative testing, the project successfully met its objectives of ensuring **data confidentiality, integrity, and user convenience**.

The system’s architecture — built using **Electron, React, and Python** — proved efficient and reliable across various test environments. Its **camouflaged interface** enhanced usability in shared systems, while **WebAuthn and face-api.js** powered biometric login without internet dependency.

The results confirmed that CyberVault delivers:

* Strong encryption with zero data leakage
* Multi-layer authentication combining password, PIN, and biometrics
* A modern, user-friendly, and stealthy desktop experience

In conclusion, CyberVault demonstrates how **AI-driven biometrics and local encryption** can work together to protect sensitive data without compromising user control or privacy. It stands as a practical and scalable cybersecurity solution for academic, professional, and personal use.

## 6.2 FUTURE SCOPE

While CyberVault fulfills its current objectives, several enhancements can further expand its functionality and impact:

**Cross-Platform Expansion:**  
Extend compatibility to **macOS and Linux**, and eventually mobile platforms (Android/iOS).

**Cloud Integration (Optional):**  
Introduce **end-to-end encrypted cloud backup** for secure multi-device synchronization.

1. **Hardware Token Support:**  
   Integrate **YubiKey** or **TPM-based** authentication for enterprise-grade security.
2. **AI-Driven File Indexing:**  
   Employ **OCR and NLP techniques** to enable smart search and content classification.
3. **Dark Mode & Custom Themes:**  
   Enhance user experience through accessibility and personalization options.
4. **Real-Time Collaboration:**  
   Add encrypted peer-to-peer file sharing and group access control for team environments.
5. **Mobile Companion App:**  
   Allow biometric unlocking and file previews directly from mobile devices.
6. **Integration with Secure Cloud APIs:**  
   Optional link with secure cloud providers like Proton Drive or Tresorit for hybrid workflows.

**In summary**, the future of CyberVault lies in merging **AI, biometrics, and user-centric encryption** to create a powerful ecosystem of tools that empower users to take full control of their data — privately, securely, and intelligently.

**LIST OF ABBREVIATIONS :**

| **Abbreviation** | **Full Form** |
| --- | --- |
| AES | Advanced Encryption Standard |
| AI | Artificial Intelligence |
| API | Application Programming Interface |
| BCNF | Boyce–Codd Normal Form |
| CPU | Central Processing Unit |
| DB | Database |
| DFD | Data Flow Diagram |
| ERD | Entity Relationship Diagram |
| GUI | Graphical User Interface |
| HTML | HyperText Markup Language |
| IPU | Guru Gobind Singh Indraprastha University |
| KDF | Key Derivation Function |
| LAN | Local Area Network |
| NLP | Natural Language Processing |
| OCR | Optical Character Recognition |
| OS | Operating System |
| PBKDF2 | Password-Based Key Derivation Function 2 |
| PKI | Public Key Infrastructure |
| SQL | Structured Query Language |
| UI | User Interface |
| UUID | Universally Unique Identifier |
| USB | Universal Serial Bus |
| UX | User Experience |
| 2FA | Two-Factor Authentication |
| MIT | Massachusetts Institute of Technology (License) |
| TPM | Trusted Platform Module |
| LAN | Local Area Network |
| CPU | Central Processing Unit |
| RAM | Random Access Memory |
| OTP | One-Time Password |
| IP | Internet Protocol |
| JS | JavaScript |
| CSS | Cascading Style Sheets |
| API | Application Programming Interface |
| HTML | HyperText Markup Language |

**APPENDICES :**

**A. System Requirements**

*Hardware Requirements*

* Processor: Intel Core i5 or higher
* RAM: Minimum 8 GB
* Storage: 200 MB for application + encrypted files storage
* Input Devices: Webcam or fingerprint scanner (for biometric authentication)
* Operating System: Windows 10 / 11 (x64)

*Software Requirements*

* Frontend: React, Tailwind CSS, Electron
* Backend: Python, Node.js
* Libraries: face-api.js, PyCA Cryptography, SQLite
* Build Tools: PyInstaller, Electron Forge
* IDE: Visual Studio Code / PyCharm

**B. Testing Environment**

| **Parameter** | **Specification** |
| --- | --- |
| Operating System | Windows 11 Pro |
| Processor | Intel Core i5-1135G7 |
| RAM | 8 GB DDR4 |
| Disk | 512 GB SSD |
| Browser Runtime | Chromium (Electron Environment) |
| Python Version | 3.12 |
| Node.js Version | 20.x |
| Frameworks | React 18, Tailwind 3.x |
| Database | SQLite (Local Encrypted) |

**Sample Code Snippet (Encryption Function – Python) :**

from cryptography.hazmat.primitives.ciphers import Cipher, algorithms, modes

from cryptography.hazmat.backends import default\_backend

import os

def encrypt\_file(file\_path, key):

iv = os.urandom(16)

cipher = Cipher(algorithms.AES(key), modes.CFB(iv), backend=default\_backend())

encryptor = cipher.encryptor()

with open(file\_path, 'rb') as f:

data = f.read()

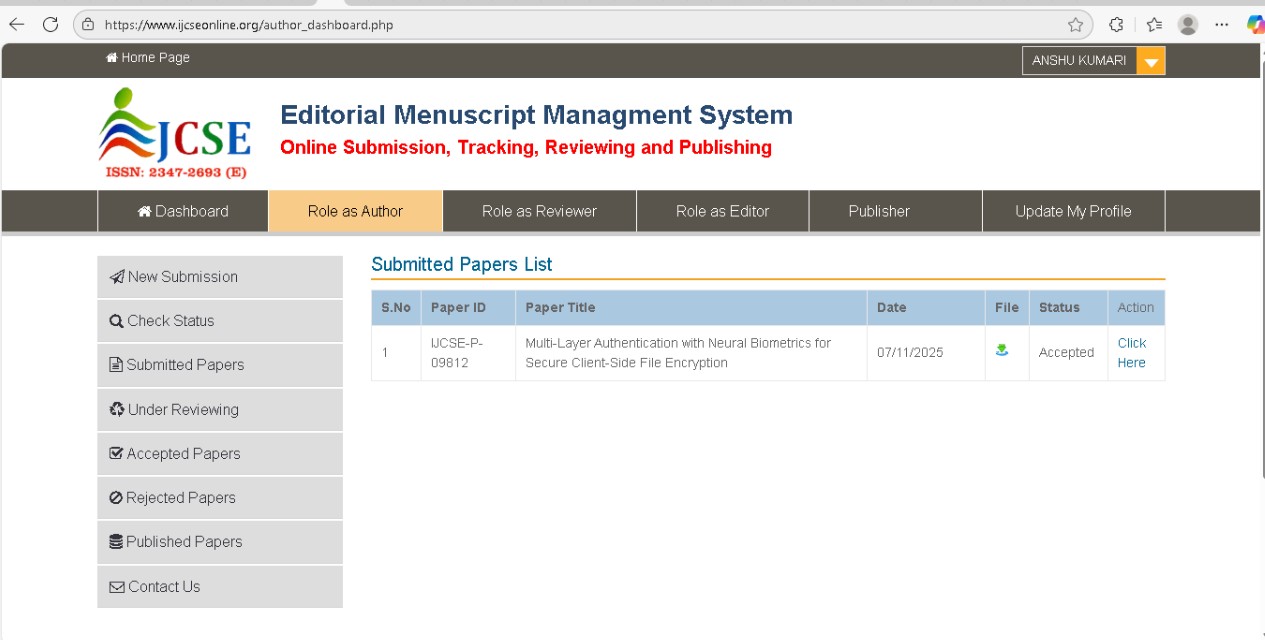
encrypted = iv + encryptor.update(data) + encryptor.finalize()

with open(file\_path + ".enc", 'wb') as ef:

ef.write(encrypted)

return "File Encrypted Successfully"

**RESEARCH PAPER PROOF :**





**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| Figure No. | Figure Title | Page No. |
| 1 | Context Diagram | 29 |
| 2 | DFD Level 1 | 30 |
| 3 | DFD Level 2 | 31 |
| 4 | Use Case Diagram | 33 |
| 5 | Activity Diagram | 48 |
| 6 | State Diagram | 49 |
| 7 | Component Diagram | 50 |
| 8 | Deployment Diagram | 52 |
| 9 | ER Diagram | 39 |
|  |  |  |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| Table No. | Table Title | Page No. |
| 1 | Gantt Chart Table | 35 |
| 2 | PERT Chart Table | 36 |
| 3 | Cost Measurement Table | 40 |
| 4 | Effort Measurement Table | 41 |
|  |  |  |