

**IEstd.1.921**

A PROJECT REPORT

ON

**CipherWeb**

Submitted By

**Mohd Kalam Aslam Pinjar**

(46421601988)

Towards the Partial Fulfillment of the

Bachelor of Computer Application

Semester VI Examination

**Bachelor of Computer Application**

Kharghar Campus Navi Mumbai

**Tilak Maharashtra Vidyapeeth, Pune**

**[2023 - 2024]**

**TILAK MAHARASHTRA VIDYAPEETH, PUNE**

**(Deemed Under Section3 of UGC Act 1956 Vide Notification**

**No.F.9-19/85**-**U3 dated 24th April 1987 By the Government of India.)**

**VIDYAPEETHBHAVAN, GULTEKDI, PUNE**-**411 037**

**DEPARTMENT OF COMPUTER SCIENCE**



Estd.1921

**CERTIFICATE**

This is to certify that the project entitled, **CipherWeb,** is the bonafide work of **Mr. Mohd** **Kalam Aslam Pinjar** bearing Seat. No: **5732** submitted in partial fulfilment of the requirements for the award of the degree of Bachelor of Computer Application, **PRN 46421601988** under Tilak Maharashtra Vidyapeeth, in the year 2023-2024.

**Internal Guide Coordinator**

**External Examiner**

**Date: College Seal**

## ACKNOWLEDGEMENT

I would like to express my deepest gratitude to everyone who has supported and guided me throughout the development of this project. First and foremost, I am immensely thankful to myself for the dedication & hard work, and perseverance I have invested in bringing this project to fruition.

I would also like to extend my heartfelt appreciation to my family and friends for their unwavering encouragement, understanding, and patience during the journey of creating this project. Their support has been invaluable in keeping me motivated and focused.

Furthermore, I am grateful to the online communities, forums, and resources that have provided me with invaluable insights, troubleshooting assistance, and inspiration along the way. Your contributions have been instrumental in overcoming challenges and refining the project to its current state.

Lastly, I would like to acknowledge the mentors, educators, and individuals who have shared their expertise, feedback, and constructive criticism, helping me grow and improve as a developer throughout this project.

Thank you all for your invaluable contributions and support. This project would not have been possible without every one of you.

## DECLARATION

I, Mohd Kalam Aslam Pinjar, hereby declare that the project titled CipherWeb, submitted as part of BCA Cyber Security is my original work. All the information, data, code, and content included in this project are authentic and have been generated by me during my studies.

I further declare that any external sources, including but not limited to books, articles, websites, and software, used in the development of this project have been properly cited and acknowledged in the bibliography section. Any contributions from individuals or organizations have been duly recognized in the acknowledgements.

I affirm that this project has not been submitted for assessment in any other course or academic program. Any similarities found with other works are purely coincidental, and any instances of collaboration or assistance from fellow students or colleagues have been appropriately acknowledged.

I understand that any attempt to misrepresent the originality or authenticity of this project may result in disciplinary action by the policies of TILAK MAHARASHTRA VIDYAPEETH.

Date -

**Name and Signature of the Student**

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **SR. No** | **Index** | **Page No.** |
| **Chapter 1** | **Introduction** | **1** |
| 1.1 | Background | 2 |
| 1.2 | Objectives | 3 |
| 1.3 | Purpose | 4 |
|  |  |  |
| **Chapter 2** | **System Analysis** | **5** |
| 2.1 | Existing System | 7 |
| 2.2 | Proposed System | 11 |
| 2.3 | Requirement Analysis | 14 |
| 2.4 | Hardware Requirements | 16 |
| 2.5 | Software Requirements | 18 |
| 2.6 | Justification Of Selection Of Technology | 20 |
|  |  |  |
| **Chapter 3** | **System Design** | **24** |
| 3.1 | Module Design | 25 |
| 3.2 | Data Dictionary | 26 |
| 3.3 | ER Diagrams | 27 |
| 3.4 | DFD/UML Diagrams | 28 |
|  |  |  |
| **Chapter 4** | **Implementation and Testing** | **31** |
| 4.1 | Code | 33 |
| 4.2 | Testing Approach | 49 |
| 4.2.1 | Unit Testing | 50 |
| 4.2.2 | Integration Testing | 51 |
|  |  |  |
| **Chapter 5** | **Results and Discussions** | **52** |
| 5.1 | System Planning (Gantt Chart) | **59** |
| **Chapter 6** | **Conclusion and Future Work** | **60** |
|  |  |  |
| **Chapter 7** | **References** | **63** |

## Chapter-1 Introduction

## 

In an era defined by digital connectivity and data exchange, ensuring the confidentiality and integrity of information has become paramount. The increasing prevalence of cyber threats underscores the need for robust cryptographic solutions to safeguard sensitive data from unauthorized access and interception.

In response to this imperative, I have developed CipherWeb, a versatile web platform dedicated to providing users with powerful encryption capabilities. CipherWeb offers a comprehensive suite of 10 cryptographic algorithms, ranging from time-honored methods like Caesar Cipher to sophisticated modern techniques like RSA**.**

Driven by a commitment to accessibility and user-friendliness, CipherWeb integrates seamlessly with Firebase backend services, ensuring scalability, reliability, and data security. Developed with Tailwind CSS and JavaScript Canvas, the platform boasts an intuitive interface and dynamic functionality, empowering users to encrypt and decrypt messages with ease.

In this project, I delve into the architecture, design, and implementation of CipherWeb, exploring its features, functionalities, and the technical intricacies behind its cryptographic algorithms. Through CipherWeb, I aim to democratize the power of encryption, empowering individuals and organizations to protect their digital privacy and security in an increasingly interconnected world.

**Background of the Project**

In today's digit all and scape, the exchange of sensitive information over the internet has become ubiquitous. However, this convenience comes with inherent risks, as malicious actors continuously seek to exploit vulnerabilities and intercept confidential data. Cryptography emerges as a critical defense mechanism, providing a means to secure communication channels and protect sensitive information from unauthorized access.

Against this backdrop, the idea for CipherWeb was conceived. Recognizing the need for accessible and user-friendly cryptographic solutions, CipherWeb aims to empower individuals and organizations with the tools to encrypt and decrypt messages securely. By offering a diverse range of cryptographic algorithms, CipherWeb caters to users with varying security requirements, from casual encryption enthusiasts to cybersecurity professionals.

**Scope of the Project:**

The scope of CipherWeb encompasses the development of a web-based platform that provides users with the ability to perform cryptographic operations seamlessly. Key aspects of the project include:

1. Algorithmic Diversity: CipherWeb offers a selection of 10 cryptographic algorithms, spanning classic ciphers to modern encryption methods. Users can choose the algorithm that best suits their security needs and preferences.
2. User Authentication and Data Security: CipherWeb implements robust user authentication mechanisms to ensure that only authorized users can access its cryptographic functionalities. Additionally, the platform prioritizes data security, employing encryption techniques to safeguard user information and communications.
3. Integration with Firebase Backend Services: To streamline development and enhance scalability, CipherWeb integrates with Firebase backend services. This integration facilitates user authentication, real-time database management, and cloud storage functionalities.
4. User Experience and Accessibility: CipherWeb prioritizes user experience, offering an intuitive interface and interactive features for seamless navigation and usage. The platform is designed to be accessible to users of all levels of technical expertise, from beginners to experienced cryptographers.

**Objective:**

The primary objective of CipherWeb is to provide users with a secure and user-friendly platform for encrypting and decrypting messages. The project aims to achieve the following specific objectives:

1. Algorithmic Diversity: Offer a diverse selection of 10 cryptographic algorithms, ranging from classic ciphers to modern encryption methods, to cater to users with varying security requirements and preferences.
2. User Authentication and Data Security: Implement robust user authentication mechanisms to ensure that only authorized users can access CipherWeb cryptographic functionalities. Prioritize data security by employing encryption techniques to safeguard user information and communications.
3. Integration with Firebase Backend Services: Streamline development and enhance scalability by integrating CipherWeb with Firebase backend services. This integration facilitates user authentication, real-time database management, and cloud storage functionalities, ensuring seamless operation and reliability.
4. User Experience and Accessibility: Prioritize user experience by offering an intuitive interface and interactive features for seamless navigation and usage. Ensure that CipherWeb is accessible to users of all levels of technical expertise, from beginners to experienced cryptographers, thereby democratizing the power of encryption.
5. Educational Resource: Serve as an educational resource for individuals interested in learning about cryptography by providing detailed documentation, explanations, and examples of each cryptographic algorithm implemented in CipherWeb. Empower users to understand and utilize cryptographic techniques effectively for securing their digital communications.

**Purpose:**

The purpose of CipherWeb is to address the growing need for accessible and user-friendly cryptographic solutions in today's digital landscape. With the proliferation of digital communication and the increasing prevalence of cyber threats, there is a heightened demand for tools that enable individuals and organizations to secure their sensitive information effectively.

CipherWeb seeks to fulfil this need by providing a versatile web platform that empowers users with the ability to encrypt and decrypt messages using a diverse range of cryptographic algorithms. By offering a selection of 10 algorithms, spanning from classic ciphers to modern encryption methods, CipherWeb caters to users with varying security requirements and preferences.

The overarching purpose of CipherWeb is to democratize the power of encryption, making it accessible to users of all levels of technical expertise. Whether used for personal communication, professional collaboration, or educational purposes, CipherWeb aims to empower individuals and organizations to protect their digital privacy and security effectively.

### Chapter 2

**System Analysis**

System analysis is a crucial phase in the development of CipherWeb, aimed at understanding the requirements, objectives, and constraints of the project. This chapter delves into the systematic examination of CipherWeb functional and non-functional requirements, user interactions, and system architecture.

* 1. Functional Requirements:
     + Cryptographic Algorithms: Identify and define the 10 cryptographic

algorithms to be implemented in CipherWeb, including their encryption and

decryption processes.

* + - User Authentication: Specify the user authentication mechanisms to ensure

secure access control to CipherWeb functionalities.

* + - Data Management: Determine the requirements for storing and managing

user data, cryptographic keys, and encryption histories.

* + - User Interface: Define the layout, design, and interactive features of the

CipherWeb user interface to facilitate intuitive navigation and usage.

* 1. Non-Functional Requirements:
     + Security: Establish security measures to safeguard user data, communications,

and system integrity against unauthorized access and cyber threats.

* + - Scalability: Ensure that CipherWeb can accommodate a growing user base and

evolving security requirements without compromising performance or

reliability.

* + - Usability: Strive for an intuitive and user-friendly interface, allowing users of

all levels of technical expertise to utilize CipherWeb cryptographic

functionalities effectively.

* + - Performance: Define performance metrics and objectives to ensure that CipherWeb

operates efficiently and responds promptly to user interactions.

* 1. User Interactions:
     + User Registration and Authentication: Analyze the user authentication process,

including registration, login, and password recovery mechanisms.

* + - Encryption and Decryption: Examine the user workflow for encrypting and

decrypting messages using CipherWeb cryptographic algorithms.

* + - Data Management: Assess how users interact with CipherWeb data management

features, including storing and retrieving cryptographic keys and encryption

histories.

* 1. System Architecture:
     + Client-Server Model: Define the architecture of CipherWeb as a client-server

application, with the frontend and backend components interacting to provide

cryptographic functionalities.

* + - Firebase Integration: Outline the integration of CipherWeb with Firebase backend

services, including authentication, real-time database management, and cloud

storage.

Through systematic analysis, CipherWeb functional and non-functional requirements, user interactions, and system architecture are delineated, laying the groundwork for the subsequent phases of development and implementation.

###### Existing System

Before the development of CipherWeb, it's essential to understand the limitations and challenges posed by existing cryptographic systems. This analysis provides insights into the shortcomings that CipherWeb aims to address.

1. Limited Algorithm Selection: Many existing cryptographic systems offer a limited

selection of algorithms, restricting users' options for securing their data. This

limitation may result in users being unable to choose the most suitable algorithm for

their specific security requirements.

1. Complex User Interfaces: Some cryptographic systems have complex and

unintuitive user interfaces, making it challenging for users, especially those with

limited technical expertise, to navigate and utilize the system effectively.

1. Security Concerns: The security of existing cryptographic systems may be

compromised due to vulnerabilities in the algorithms or implementation flaws. These

vulnerabilities could potentially expose sensitive data to unauthorized access or

interception.

1. Inefficient Data Management: Managing cryptographic keys, user profiles, and

encryption histories in existing systems may be cumber some and inefficient, leading

to delays in operations and potential data loss.

1. Lack of Real-Time Updates: Without real-time updates and synchronization, users

may encounter inconsistencies in their cryptographic histories, making it difficult to

track their past encryption activities accurately.

1. Scalability Challenges: Existing cryptographic systems may face scalability

challenges, particularly concerning accommodating a growing user base or evolving

security requirements. This limitation hinders the system's ability to meet users' needs

effectively.

###### 2.1 Proposed System: CipherWeb

CipherWeb is a revolutionary cryptographic platform designed to address the limitations of existing systems while providing users with a comprehensive suite of cryptographic functionalities. The proposed system offers a range of innovative features aimed at enhancing security, usability, and scalability.

1. Expanded Algorithm Selection:

- CipherWeb offers users a diverse selection of 10 cryptographic algorithms, ranging

from classic ciphers like Caesar Cipher to modern encryption methods like **RSA.** This

expanded selection ensures that users have access to the most suitable algorithm for

their specific security requirements.

1. User-Friendly Interface:

- CipherWeb boasts an intuitive and user-friendly interface designed to streamline

cryptographic operations. With clear navigation and interactive elements, users can

easily encrypt and decrypt messages without the need for extensive technical

knowledge.

1. Robust Security Measures:

- Security is paramount in CipherWeb, with robust encryption techniques and best

practices implemented to safeguard user data and communications from unauthorized

access and interception. Advanced security measures ensure the confidentiality and

integrity of user information.

1. Efficient Data Management:

- CipherWeb incorporates efficient data management capabilities, allowing users to

securely store cryptographic keys, manage their profiles, and track their encryption

histories. Real-time updates and synchronization ensure accuracy and reliability in

data management.

1. Seamless Integration with Firebase:
   * CipherWeb seamlessly integrates with Firebase backend services, including

authentication, real-time database management, and cloud storage. platforms.

1. Scalability and Flexibility:
   * Built on a scalable architecture, CipherWeb can accommodate a growing user base

and evolving security requirements. Whether used by individuals, businesses, or

organizations, CipherWeb offers scalability and flexibility to meet diverse needs.

1. Educational Resource:
   * CipherWeb serves as an educational resource, providing users with detailed

documentation, explanations, and examples of each cryptographic algorithm

implemented. Users can learn about cryptography while utilizing CipherWeb

encryption capabilities.

By combining innovative features, robust security measures, and seamless integration with Firebase backend services, CipherWeb emerges as a groundbreaking cryptographic platform, empowering users to protect their digital privacy and security with confidence and ease.

###### 2.2 Requirement Analysis

Requirement analysis is a critical phase in the development of CipherWeb, focusing on gathering, analyzing, and documenting the functional and non-functional requirements of the system. This process involves collaboration with stakeholders to ensure that CipherWeb meets their needs effectively. The following aspects are considered during requirement analysis:

1. Functional Requirements:
   * Cryptographic Algorithms: Identify the 10 cryptographic algorithms to be

implemented in CipherWeb, including their encryption and decryption processes.

* + User Authentication: Specify the user authentication mechanisms, including

registration, login, and password recovery, to ensure secure access control to

CipherWeb functionalities.

- Data Management: Determine their requirements for storing and managing user

data, cryptographic keys, and encryption histories within CipherWeb.

* + User Interface: Define the layout, design, and interactive features of CipherWeb

user interface to facilitate intuitive navigation and usage.

1. Non-Functional Requirements:
   * Security: Establish security measures to safeguard user data, communications, and

system integrity against unauthorized access and cyber threats.

* + Scalability: Ensure that CipherWeb can accommodate a growing user base and

evolving security requirements without compromising performance or reliability.

* + Usability: Strive for an intuitive and user-friendly interface, allowing users of all

levels of technical expertise to utilize CipherWeb cryptographic

effectively.

* + Performance: Define performance metrics and objectives to ensure that

CipherWeb operates efficiently and responds promptly to user interactions.

1. User Interactions:
   * User Registration and Authentication: Analyze the user authentication process,

including registration, login, and password recovery mechanisms, to ensure a

seamless user experience.

* + Encryption and Decryption: Examine the user workflow for encrypting and

decrypting messages using CipherWeb cryptographic algorithms, identifying any

potential usability challenges.

* + Data Management: Assess how users interact with CipherWeb data management

features, including storing and retrieving cryptographic keys and encryption

histories, to streamline operations and enhance user satisfaction.

1. System Architecture:
   * Client-Server Model: Define the architecture of CipherWeb as a client-server

application, outlining the interaction between the frontend and backend

components.

* + Firebase Integration: Specify the integration of CipherWeb with Firebase backend

services, including authentication, real-time database management, and cloud

storage, to ensure seamless operation and scalability.

Through comprehensive requirement analysis, CipherWeb aims to gather and document the needs and expectations of stakeholders effectively, laying the foundation for the subsequent phases of development and implementation.

* 1. Hardware Requirements

CipherWeb is a web-based cryptographic platform that primarily relies on software components for its operation. However, certain hardware requirements are necessary to support the deployment and usage of CipherWeb effectively. The hardware requirements for CipherWeb are as follows:

1. Client Devices:

- Desktop/Laptop Computers: Users can access CipherWeb using desktop or

laptop computers running modern web browsers such as Google Chrome,

Mozilla Firefox, Safari, or Microsoft Edge.

- Mobile Devices: CipherWeb should also be accessible on mobile devices such as

smartphones and tablets, providing users with flexibility and convenience in

accessing the platform.

1. Internet Connectivity:
   * Stable Internet Connection: Users require a stable Internet connection to access

CipherWeb and perform cryptographic operations seamlessly. A reliable internet

connection ensures smooth communication between the client device and the

CipherWeb server.

1. Server Infrastructure:
   * Web Server: CipherWeb requires a web server to host the frontend components of

the platform, including HTML, CSS, and JavaScript files. Commonly used web

servers such as Apache HTTP Server can fulfil this requirement.

* + Firebase Backend Services: Since CipherWeb integrates with Firebase backend

services for authentication, real-time database management, and cloud storage,

access to Firebase's server infrastructure is necessary for seamless operation.

1. Storage:
   * Firebase Cloud Storage: CipherWeb utilizes Firebase Cloud Storage for securely

storing user profile pictures and media assets. Access to Firebase Cloud Storage infrastructure is required to ensure reliable storage and retrieval of user data.

1. Security Considerations:

- SSL/TLS Certificate: To ensure secure communication between client devices and

the CipherWeb server, an SSL/TLS certificate is recommended. This certificate

encrypts data transmitted over the network, protecting it from interception and

unauthorized access.

- Firewall and Intrusion Detection System (IDS): Implementing firewall and IDS

measures on the server infrastructure help protect against unauthorized access and

cyber threats, ensuring the security of user data stored in CipherWeb.

By meeting these hardware requirements, CipherWeb can be deployed and utilized effectively, providing users with a secure and reliable platform for cryptographic operations.

###### Software Requirements

CipherWeb, as a web-based cryptographic platform, relies on a combination of frontend and backend software components to deliver its functionalities securely and efficiently. The software requirements for CipherWeb are as follows:

1. Frontend Technologies:

**- HTML, CSS,** and JavaScript: These fundamental web development languages are

used to create the structure, style, and interactivity of the user interface.

* Tailwind **CSS: A** utility-first CSS framework used to streamline the design

process and ensure responsiveness across devices, enhancing the visual aesthetics

and user experience of CipherWeb.

* JavaScript Canvas: Utilized for dynamic rendering of cryptographic algorithms,

allowing users to visualize and interact with the encryption and decryption

processes in real time.

1. Backend Technologies:

- Firebase: A comprehensive platform provided by Google, offering backend

services including authentication, real-time database management, and cloud

storage. CipherWeb integrates with Firebase to ensure seamless operation and

scalability.

- Firebase Authentication: Provides secure user authentication mechanisms,

allowing users to register, log in, and access cryptographic functionalities.

- Firebase Real-Time Database: Offers a scalable No SQL cloud database for

storing and synchronizing user data, cryptographic keys, and encryption histories

in real-time.

* Firebase Cloud Storage: Facilitates secure storage and retrieval of user profile

pictures and media assets, enhancing the overall user experience of CipherWeb.

1. Integrated Development Environment(IDE):

* Visual Studio Code: A versatile code editor with built-in support for HTML,

CSS, JavaScript, and various plugins for Firebase integration and development efficiency. Visual Studio Code streamlines the development process of CipherWeb and provides essential tools for debugging and testing.

1. Version Control:

* Git: A distributed version control system used to track changes in the codebase and

collaborate with team members efficiently. Git enables developers to manage code

changes, merge branches, and maintain a history of revisions effectively.

By leveraging these software technologies, CipherWeb ensures a seamless and secure platform for cryptographic operations, meeting the needs of individual users and enterprises alike.

###### 2.4 Justification of Platform

The selection of the Firebase platform for integrating backend services into CipherWeb is based on several compelling justifications:

1. Comprehensive Suite of Services: Firebase offers a comprehensive suite of backend services, including authentication, real-time database management, and cloud storage. This integrated approach streamlines development, reduces complexity, and ensures seamless interoperability between different components of CipherWeb.
2. Scalability and Reliability: Firebase provides scalable and reliable backend infrastructure, capable of handling a growing user base and fluctuating demands without compromising performance or reliability. This scalability ensures that CipherWeb can accommodate increasing usage and evolving requirements over time.
3. Real-Time Data Synchronization: Firebase's real-time database enables seamless data synchronization across multiple clients in real time, ensuring that changes made by one user are immediately reflected to others.
4. Authentication and Security: Firebase Authentication offers robust authentication mechanisms, including email/password authentication, social login options, and federated identity providers.
5. Cloud-Based Storage: Firebase Cloud Storage provides reliable and scalable cloud-based storage for user profile pictures, media assets, and other user-generated content within CipherWeb. This ensures efficient storage and retrieval of data while maintaining data integrity and availability.
6. Integration with JavaScript: Firebase offers seamless integration with JavaScript, the primary language used for frontend development in CipherWeb.

### Chapter 3

### System Design

System design is a critical phase in the development of CipherWeb, focusing on translating the requirements gathered during the analysis phase into a well-structured and efficient system architecture. This chapter provides a detailed overview of the system design for CipherWeb, encompass the frontend and backend components, database structure, security considerations, and integration with Firebase backend services.

* 1. Frontend Design:

The frontend design of CipherWeb focuses on creating an intuitive and user-friendly interface that enables users to access and utilize the platform's cryptographic functionalities seamlessly. Key aspects of the frontend design include:

* + - User Interface (UI): UI is designed to be visually appealing and responsive, ensuring

optimal usability across desktop and mobile devices. Tailwind CSS is utilized to

streamline the design process and ensure consistency in styling.

* + - Interactive Features: The frontend incorporates interactive features using JavaScript

and JavaScript Canvas to enable users to visualize and interact with the encryption

and decryption processes in real-time.

* + - User Authentication The frontend includes user authentication mechanisms, allowing

users to register, log in, and access cryptographic functionalities securely.

* 1. Backend Design:

The backend design of CipherWeb encompasses the server-side components responsible for handling user requests, processing cryptographic operations, and managing data. Key aspects of the backend design include:

* + - Server Architecture: CipherWeb adopts a client-server architecture, where the

frontend communicates with the backend server to perform cryptographic operations

and retrieve data. Node.js is used as the runtime environment for the backend server.

* + - Firebase Integration: CipherWeb integrates with Firebase backend services, including

authentication, real-time database management, and cloud storage, to ensure seamless

operation and scalability. Firebase Authentication handles user authentication, while

Firebase Realtime Database and Firebase Cloud Storage manage user data and media

assets, respectively.

* 1. **Database Design:**

The database design of CipherWeb focuses on defining the structure and organization of data within the Firebase Real Time Database. Key aspects of the database design include:

* + - Data Model: data model is designed to store user profiles, cryptographic keys,

encryption histories, and other relevant data in a structured format within the Firebase

Real-Time Database.

- Real-Time Data Synchronization: Firebase's real-time database enables seamless

data synchronization across multiple clients in real-time, ensuring that changes made

by one user are immediately reflected to others. This capability enhances

collaboration and user engagement within CipherWeb.

* 1. Security Considerations:

Security is a paramount concern in the design of CipherWeb, with robust measures implemented to safeguard user data, communications, and system integrity. Key security considerations include:

* + - Encryption: All sensitive data, including user passwords and cryptographic keys, are

encrypted using strong encryption algorithms to prevent unauthorized access and

interception.

* + - Authentication: Firebase Authentication provides robust authentication mechanisms,

including email/password authentication and social login options, to ensure secure

access control to cryptographic functionalities.

* + - Secure Communication: CipherWeb utilizes SSL/TLS encryption to secure communication between client devices and the backend server, protecting data from interception and unauthorized access.
    - Data Protection: Firebase Cloud Storage employs advanced security features such as

access controls and encryption at rest to protect user profile pictures and media assets

stored within CipherWeb.

* 1. Integration with Firebase: Integration with Firebase backend services is a key component of design, providing scalability, reliability, and seamless operation. Integration points include:
     + Authentication Integration: CipherWeb integrates with Firebase Authentication to

handle user registration, login, and access control, ensuring secure authentication

mechanisms are in place.

- Real-Time Database Integration: Firebase Real-Time Database manages user data,

cryptographic keys, and encryption histories, enabling seamless data synchronization

and collaboration within CipherWeb.

* + - Cloud Storage Integration: Firebase Cloud Storage securely stores user profile

pictures and media assets, ensuring efficient storage and retrieval while maintaining

data integrity and availability.

Through comprehensive system design, CipherWeb aims to provide users with a robust, scalable, and secure cryptographic platform that meets their needs effectively and ensures a seamless user experience.

**3.5 Module Division**

1. Authentication Module:

- Responsible for user authentication mechanisms, including registration, login, and

password recovery.

* + - Integrates with Firebase Authentication to ensure secure access control to

CipherWeb's functionalities.

* + - Provides options for email/password authentication, social login, and federated

identity providers.

1. Cryptographic Algorithms Module:
   * Implements the 10 cryptographic algorithms offered by CipherWeb, including

encryption and decryption processes.

- Each algorithm is encapsulated with in its module, allowing for easy integration

and extensibility.

* + Provides functionalities for users to select the desired algorithm and perform

cryptographic operations.

* + User Profile Module:

- Manages user profile information, including user details, preferences, and settings.

* + Allows users to update their profile information, change passwords, and manage account settings.
  + Integrates with Firebase Authentication to synchronize user profile data securely.

1. Encryption History Module:
   * Tracks and manages the encryption histories of users, recording details such as

timestamp, the algorithm used, and encrypted messages.

- Enables users to view their encryption history, search for specific entries, and

manage historical records.

1. User Interface (UI) Module:

- Designs and implements the user interface of CipherWeb, focusing on visual

aesthetics, responsiveness, and usability.

- Incorporates interactive features using JavaScript and JavaScript Canvas to

Enhance user engagement and experience.

- Ensures compatibility across desktop and mobile devices catering to the diverse

needs of users.

1. Data Management Module:

- Handles the storage and retrieval of user data, cryptographic keys, and encryption

histories within CipherWeb.

* + Integrates with Firebase Realtime Database and Firebase Cloud Storage for

efficient data management and synchronization.

* + Implements data validation and error handling mechanisms to ensure data integrity

and reliability.

1. Security Module:

- Implements security measures to safeguard user data, communications and system

integrity.

* + - Utilizes encryption techniques to protect sensitive data including user passwords

and cryptographic keys.

- Ensures secure communication between client devices and the backend server using

**SSL/TLS** encryption.

1. Integration Module:
   * Facilitates integration with Firebase backend services including authentication,

real-time database management, and cloud storage.

- Handles communication between frontend and backend components ensuring

seamless operation and scalability.

###### Data Dictionary

1. User Profile Data:
   * Fields:
     + Username: The unique identifier for each user.
     + Email: The email address associated with the user's account.
     + Password: The hashed password for user authentication.
     + Profile Picture: The image uploaded by the user to represent their profile.
   * Description: Stores user profile information, including username, email, password, and profile picture.
2. Cryptographic Key Data:
   * Fields:
     + Algorithm: The cryptographic algorithm used for encryption and decryption.
     + Key**:** The cryptographic key generated for the selected algorithm.

- Description: Stores cryptographic keys generated by CipherWeb for encrypting and

decrypting messages.

1. Encryption History Data:
   * Fields:
     + Timestamp: The date and time when the encryption was performed.
     + Algorithm: The cryptographic algorithm used for encryption.
     + Encrypted Message: The cipher text resulting from the encryption process.

- Description: Records the encryption history of users, including details such as

timestamp, algorithm used, and encrypted message.

1. User Session Data:
   * Fields:
     + SessionID: The unique identifier for each user session.
     + UserID: The identifier for the user associated with the session.
     + Last Active: The time stamp indicating the last activity time for the session.

- Description: Tracks active user sessions within CipherWeb, enabling session

management and authentication.

1. Error Log Data:
   * Fields:
     + ErrorID: The unique identifier for each error log entry.
     + Timestamp: The date and time when the error occurred.
     + Error Message: The description of the error encountered.
     + Description: Stores error log entries generated by CipherWeb, including details

such as timestamp and error message for troubleshooting and debugging purposes.

1. System Configuration Data:
   * Fields
     + ConfigurationID: The unique identifier for each system configuration entry.
     + Setting Name: The name of the configuration setting.
     + Setting Value: The value assigned to the configuration setting.

- Description: Stores system configuration settings for CipherWeb, allowing for

customization and adjustment of various system parameters.

1. Session Token Data:
   * Fields:
     + Token**ID:** The unique identifier for each session token.
     + UserID: The identifier for the user associated with the token.
     + Expiry Time: The time stamp indicating the expiration time for the token.

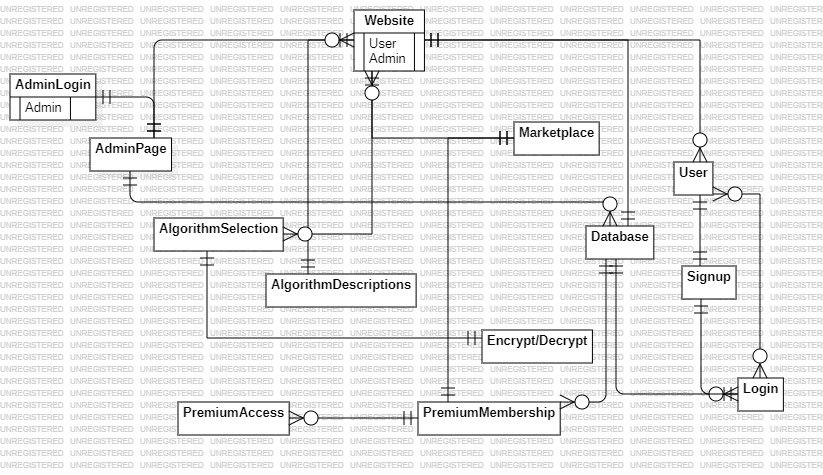
- Description: Stores session tokens generated for user authentication and

authorization within CipherWeb.

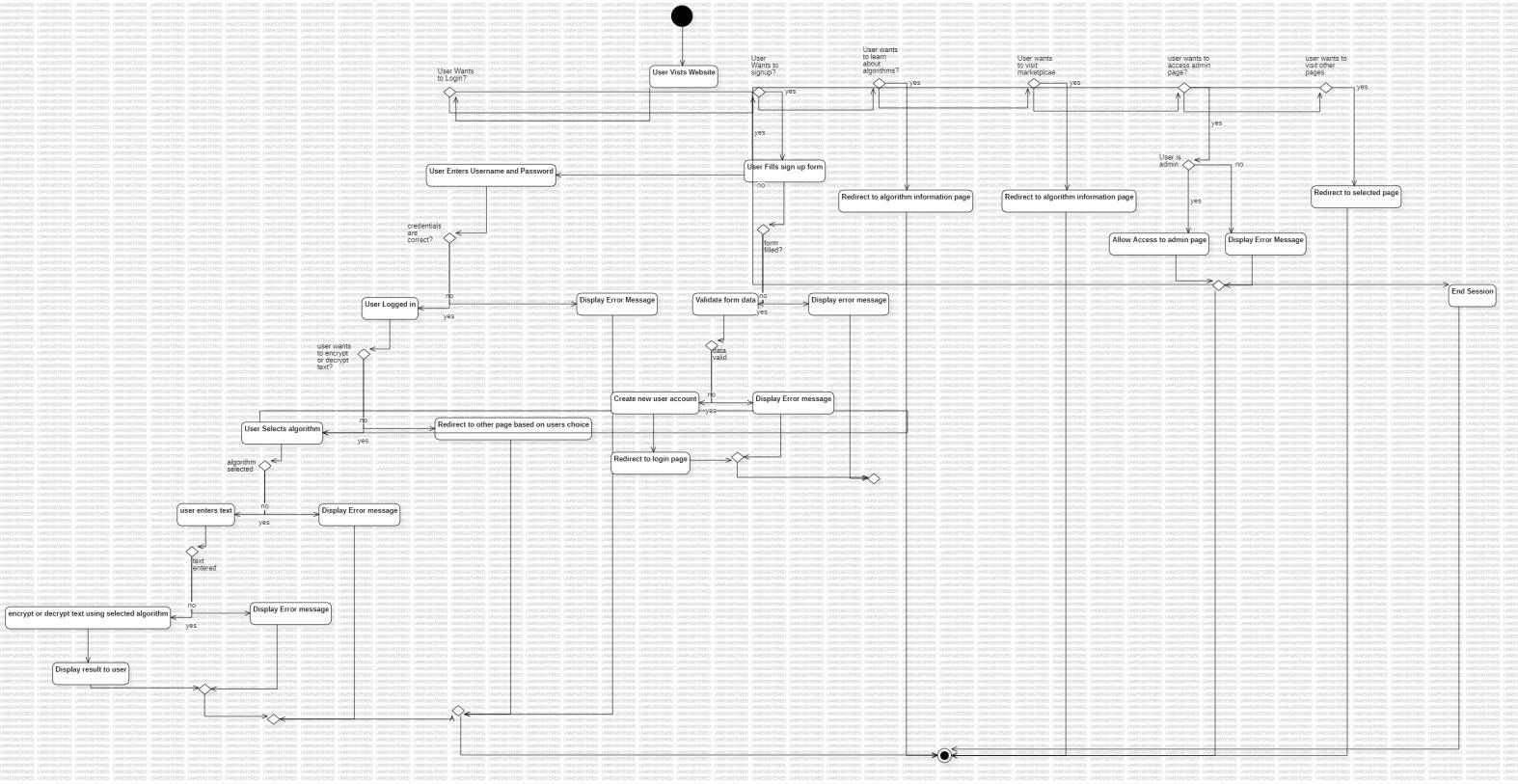
**CHAPTER 3   
SYSTEM DESIGN**

|  |  |
| --- | --- |
| Sr.No | **Diagram** |
| 1. | E-R Diagram |
| 2. | Activity Diagram |
| 3. | Class Diagram |
| 4. | Use Case Diagram |
| 5. | Sequence Diagram |
| 6. | DFD Diagram |

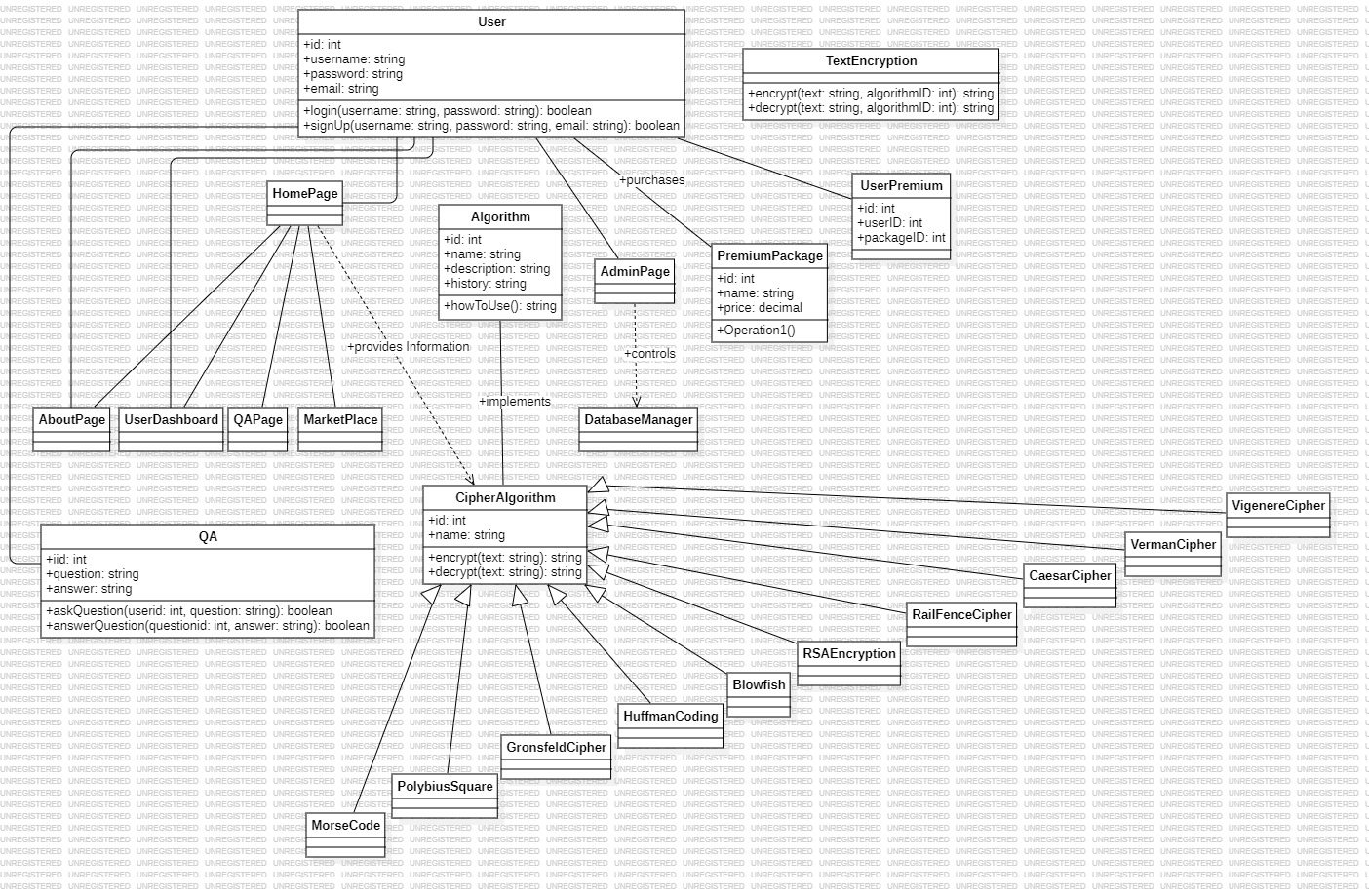
1. E-R Diagrams



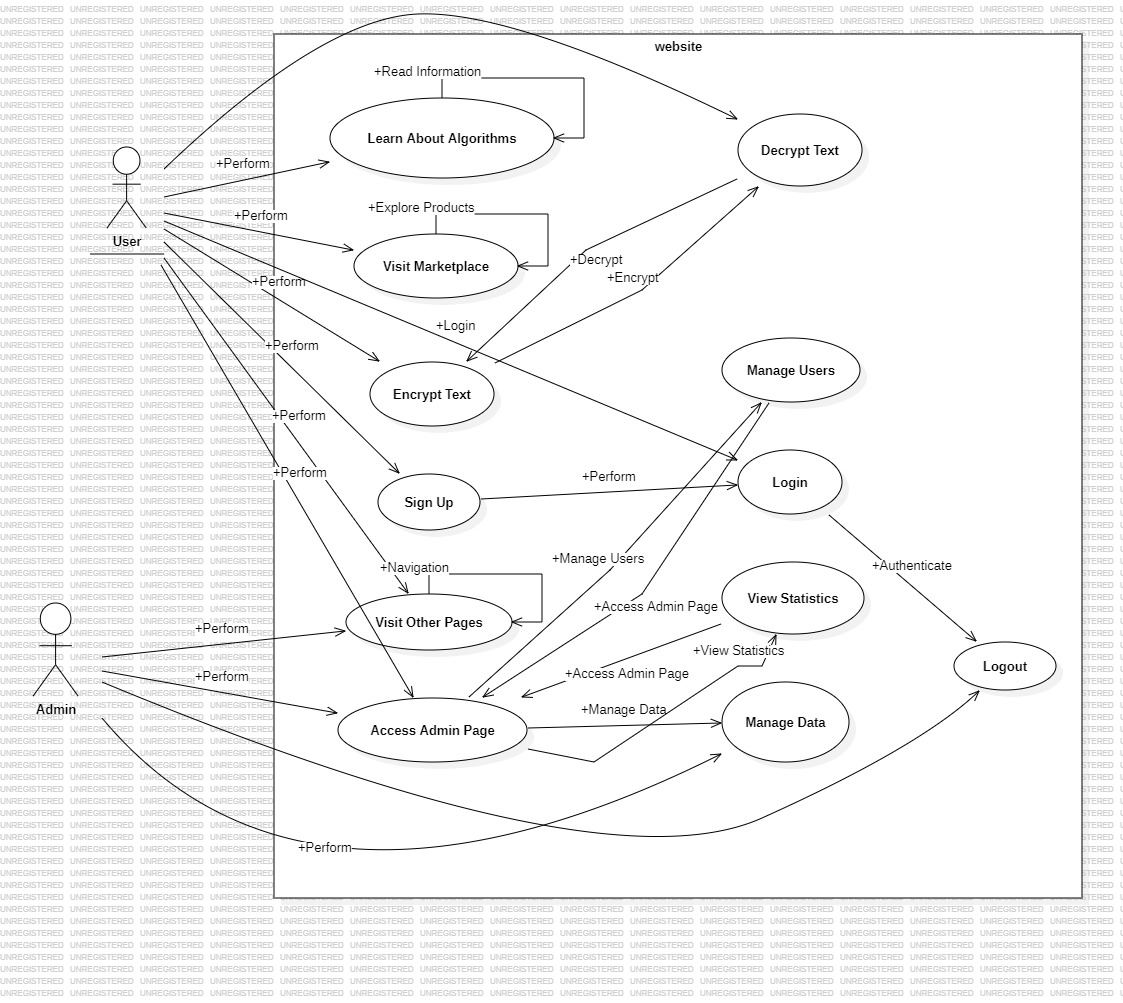
1. Activity Diagram



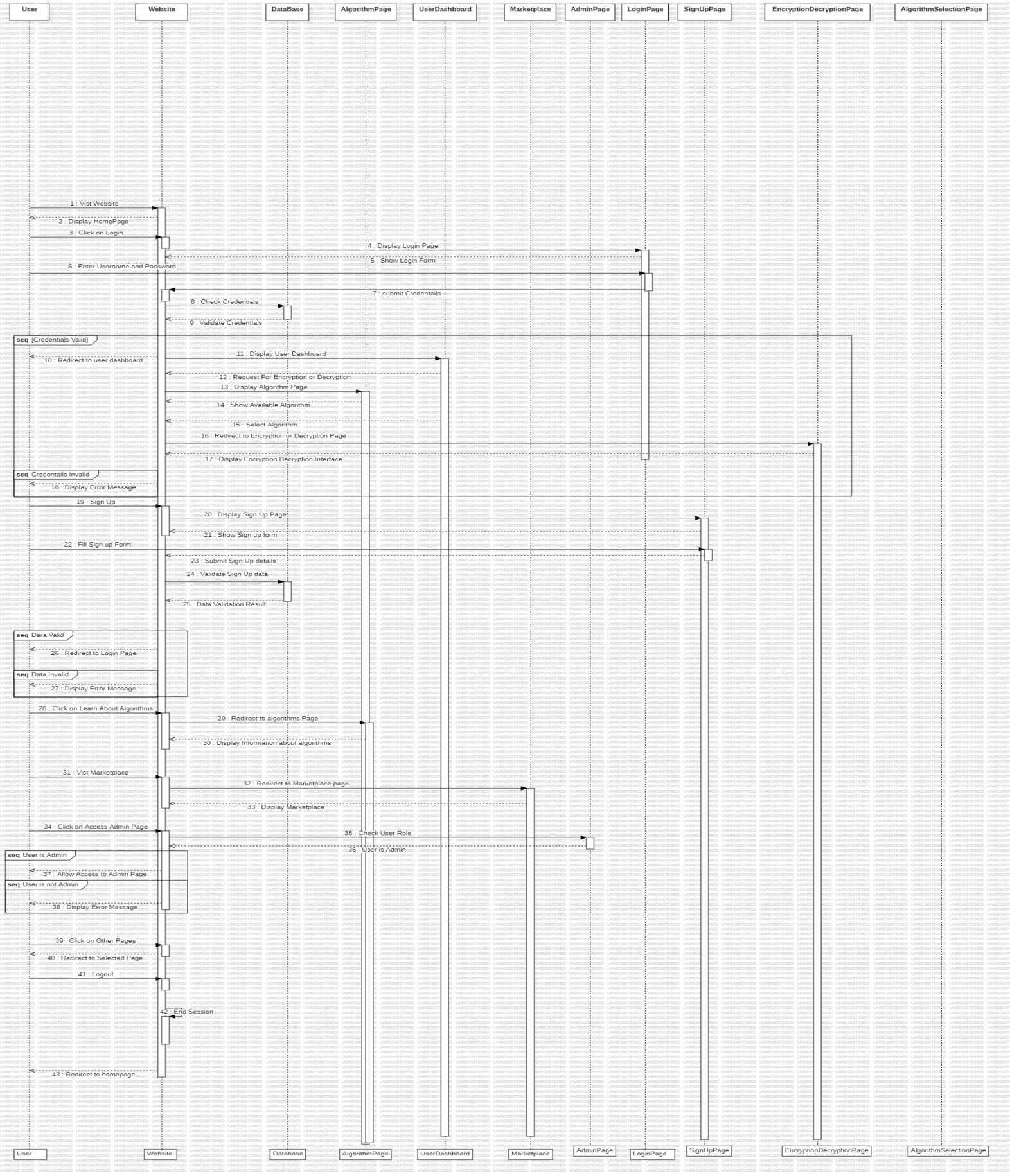
1. Class Diagram



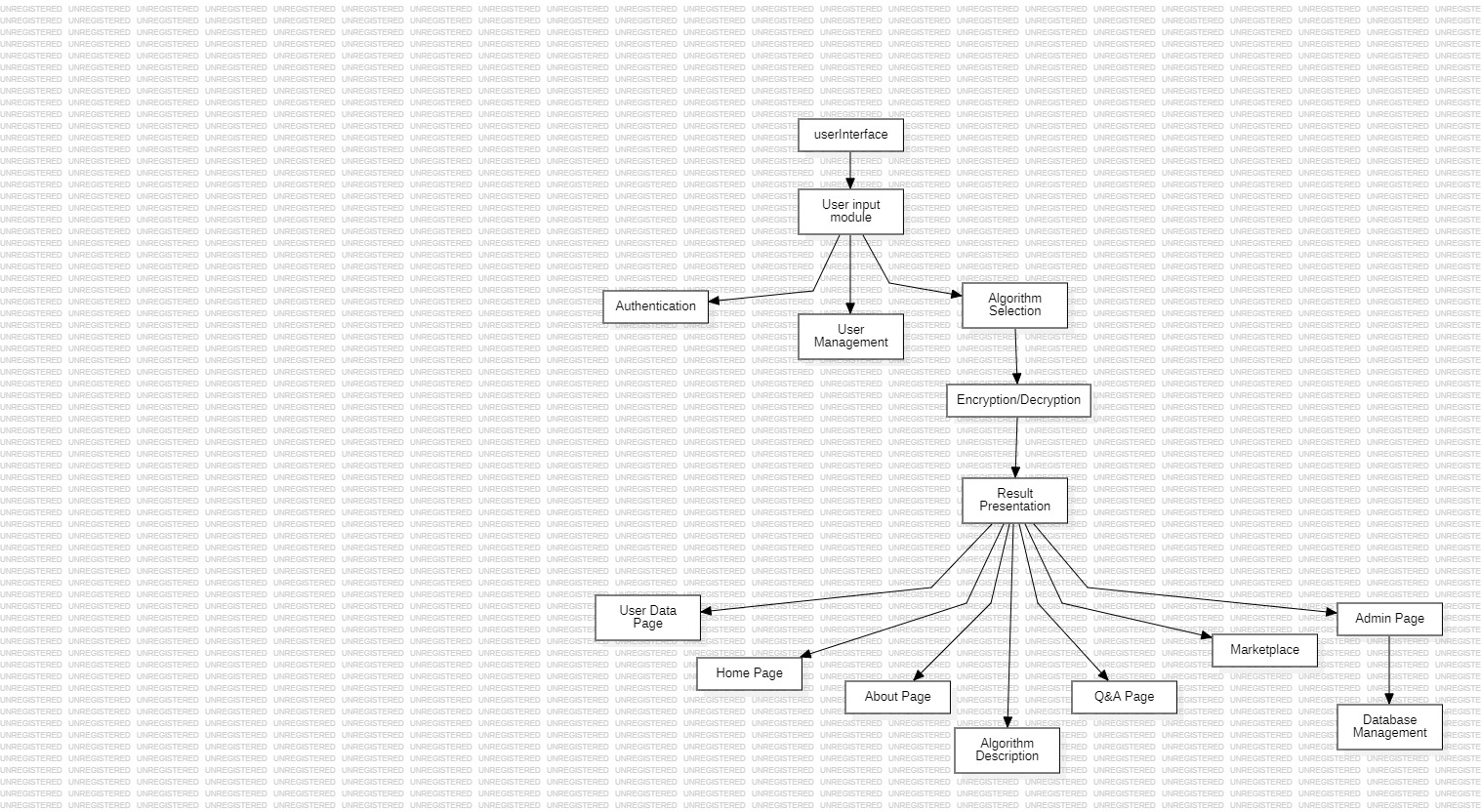
4.Use Case Diagram



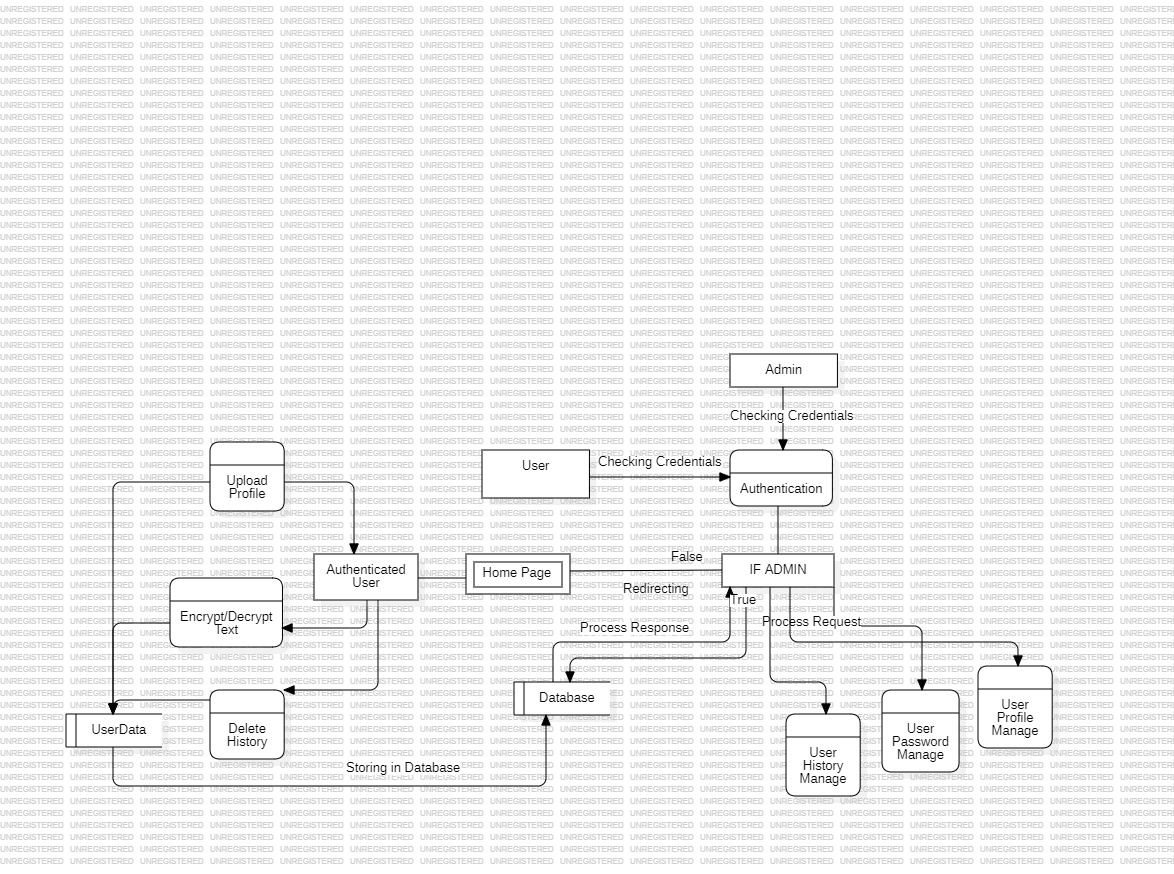
5.Sequence Diagram



##### DFD Diagram(Level-0)



##### DFD Diagram(Level-1)



#### Chapter 4

#### Implementation and Testing

* 1. **Code**

1. Landing Page Code

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>CipherWeb</title>

    <link rel="icon" href="/images/optimized-img/othersideicon.png" type="image/icon type">

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.1/css/all.min.css"

        integrity="sha512-DTOQO9RWCH3ppGqcWaEA1BIZOC6xxalwEsw9c2QQeAIftl+Vegovlnee1c9QX4TctnWMn13TZye+giMm8e2LwA=="

        crossorigin="anonymous" referrerpolicy="no-referrer" />

    <script src="https://cdn.tailwindcss.com"></script>

    <link rel="stylesheet" href="css/formstyle1.css">

    <link rel="stylesheet" href="css/main.css">

</head>

<body id="body">

    <header class="header">

        <nav class="navbar">

            <h2 class="logo">

                <a href="/LandingPage/landing-page.html">

                    <canvas id="canvas1"></canvas>

                </a>

            </h2>

            <input type="checkbox" id="menu-toggle" />

            <label for="menu-toggle" id="hamburger-btn">

                <svg xmlns="http://www.w3.org/2000/svg" height="24" viewBox="0 0 24 24" width="24">

                    <path d="M3 12h18M3 6h18M3 18h18" stroke="currentColor" stroke-width="2" stroke-linecap="round" />

                </svg>

            </label>

            <img id="avatarButton" type="button" data-dropdown-toggle="userDropdown"

                data-dropdown-placement="bottom-start" class="rounded-full w-10 h-10 cursor-pointer ring-[green] ring-2"

                src="https://media.istockphoto.com/id/1130884625/vector/user-member-vector-icon-for-ui-user-interface-or-profile-face-avatar-app-in-circle-design.jpg?s=612x612&w=0&k=20&c=1ky-gNHiS2iyLsUPQkxAtPBWH1BZt0PKBB1WBtxQJRE="

                alt="User dropdown">

            <!-- Dropdown menu -->

            <div class="userDropdownShow" id="userDropdown" class="z-10 shadow rounded-lg">

                <div class="px-4 py-3 text-gray-900 text-sm dark:text-white">

                    <div id="user-email" class="font-medium truncate">User Not Login! </div>

                    <hr color="white">

                </div>

                <ul id="userloggedin" class="py-2 text-gray-700 text-sm dark:text-gray-200"

                    aria-labelledby="avatarButton">

                    <li>

                        <a href="/dashboard.html" class="block px-4 py-2">Dashboard</a>

                        <hr color="white">

                    </li>

                    <li>

                        <a href="/history.html" class="block px-4 py-2">History</a>

                        <hr color="white">

                    </li>

                </ul>

                <div id="signout" class="py-1">

                    <li class="block px-4 py-2 text-sm">Sign

                        out</li>

                </div>

            </div>

            <ul class="links">

                <li><a href="/LandingPage/landing-page.html">Home</a></li>

                <li><a href="/About/about.html">About Us</a></li>

                <li><a href="/Services/services.html">Services</a></li>

                <li><a href="/MarketPlace/marketplace.html">Marketplace</a></li>

                <li><a href="/Contact/contact.html">Contact Us</a></li>

                <li><a href="/Contact/contact.html"></a></li>

                <li id="menu-products">

                    <a href="#">Products</a>

                    <ul id="dropdown-products">

                        <li><a href="/Cryptography-text-all-files/Caesar cipher.html">Caesar-Cipher</a></li>

                        <li><a href="/Cryptography-text-all-files/RSA/rsa.html">RSA-Cipher</a></li>

                        <li><a href="/Cryptography-text-all-files/Morse/morse.html">Morse-Cipher</a></li>

                        <li><a href="/Cryptography-text-all-files/Vernam.html">Vernam-Cipher</a></li>

                        <li><a href="/Cryptography-text-all-files/Vigenère cipher.html">Vigenère-Cipher</a></li>

                        <li><a href="/Cryptography-text-all-files/RailFence Trail.html">RailFence-Cipher</a></li>

                        <li><a href="/Cryptography-text-all-files/Polybius Square/index.html">Polybius

                                Square-Cipher</a>

                        </li>

                        <li><a href="/Cryptography-text-all-files/Huffman/huffman.html">Huffman-Cipher</a></li>

                        <li><a href="/Cryptography-text-all-files/blowfish/BlowfishCipher.html">Blowfish-Cipher</a>

                        </li>

                        <li><a href="/Cryptography-text-all-files/Gronsfeld/Gronsfeld.html">Gronsfeld-Cipher</a>

                        </li>

                    </ul>

                </li>

                <li><a href="/documentation/index.html">Docs</a></li>

                <form action="https://duckduckgo.com/" class="md:w-40 lg:w-66">

                    <span class="relative flex items-center group">

                        <svg aria-hidden="true" viewBox="0 0 20 20"

                            class="group-hover:fill-lime-500 group-focus:fill-lime-500 absolute ml-3 w-4 h-4 fill-lime-400">

                            <path

                                d="M16.293 17.707a1 1 0 0 0 1.414-1.414l-1.414 1.414ZM9 14a5 5 0 0 1-5-5H2a7 7 0 0 0 7 7v-2ZM4 9a5 5 0 0 1 5-5V2a7 7 0 0 0-7 7h2Zm5-5a5 5 0 0 1 5 5h2a7 7 0 0 0-7-7v2Zm8.707 12.293-3.757-3.757-1.414 1.414 3.757 3.757 1.414-1.414ZM14 9a4.98 4.98 0 0 1-1.464 3.536l1.414 1.414A6.98 6.98 0 0 0 16 9h-2Zm-1.464 3.536A4.98 4.98 0 0 1 9 14v2a6.98 6.98 0 0 0 4.95-2.05l-1.414-1.414Z">

                            </path>

                        </svg>

                        <input type="text" name="q" placeholder="Search docs…"

                            class="bg-lime-100 py-2 pr-2 pl-10 border border-lime-100 hover:border-lime-200 focus:border-lime-200 rounded w-full text-lime-800 outline outline-2 outline-offset-2 outline-transparent placeholder-lime-400 focus:outline-lime-600" />

                    </span>

                    <input type="hidden" name="sites" value="spinalcms.com">

                    <input type="submit" value="Search" class="sr-only" />

                </form>

            </ul>

            <div id="nav-login" class="buttons">

                <a href="/Login/Form.html" class="login-btn1 signup">Login In</a>

            </div>

        </nav>

        <div id="toast-signout"

            class="flex items-center bg-white dark:bg-gray-800 shadow p-4 rounded-lg w-full max-w-xs text-gray-500 dark:text-gray-400"

            role="alert">

    </header>

2.**GSAP Animation For Landing Page**

const tl = gsap.timeline({ default: { duration: 1, ease: Expo.easeInOut, } })

tl.to(".header", {

    opacity: 1

}, 'a')

    .from(".navbar", {

        yPercent: -100,

        opacity: 0

    }, 'a').to('.navbar', {

        opacity: 1,

        yPercent: 0,

        stagger: 1,

        duration: 1,

        ease: Expo.easeInOut,

        opacity: 1,

    }, 'a')

tl.from('#hero-sect', {

    opacity: 0,

    xPercent: 100

}, 'b').to('#hero-sect', {

    xPercent: 0,

    opacity: 1,

    stagger: .1,

    duration: 1,

    ease: Power4,

}, 'b')

tl.from("#canvas2", {

    scale: 0,

}, 'c').to("#canvas2", {

    scale: 1,

    duration: 1,

    ease: Power2,

}, 'c')

const tl2 = gsap.timeline({

    scrollTrigger: {

        trigger: "#hero-sect",

        start: "top top",

        end: "bottom top",

        scrub: 2,

    },

})

tl2.to("#canvas2", {

    scale: 5,

    opacity: 0,

    stagger: .03,

    ease: Power4,

})

const tl3 = gsap.timeline({

    scrollTrigger: {

        trigger: ".main",

        start: "top top",

        end: "bottom top",

        pin: true,

        scrub: 2,

    },

})

tl3.to(".main", {

    '--circle': "0%",

    ease: Power2,

}, 'tl3a').to(".video", {

    opacity: 0.3

}, 'tl3a').from("#circularTextCanvas", {

    scale: 0.5

}, "tl3a").to("#circularTextCanvas", {

    scale: 10,

    delay: 3,

    display:"none"

})

const tl4 = gsap.timeline({

    scrollTrigger: {

        trigger: ".container3",

        start: "top top",

        end: "bottom top",

        pin: true,

        scrub: 2,

    },

})

tl4.to(".col3", {

    yPercent: -100,

    opacity: 0

})

const tl5 = gsap.timeline({

    scrollTrigger: {

        trigger: ".container2-bg",

        start: "top 100px",

        end: "-100px top",

        pin: true,

        scrub: 2,

    },

})

tl5.from(".odd", {

    xPercent: -110,

}).to(".odd", {

    xPercent: 0,

    stagger: true

})

tl5.from(".even", {

    xPercent: 110,

}).to(".even", {

    xPercent: 0,

    stagger: true

})

3.**Form Page :**

    <div id="toast-default"

        class="flex items-center bg-white dark:bg-gray-800 shadow p-4 rounded-lg w-full max-w-xs text-gray-500 dark:text-gray-400"

        role="alert">

    </div>

    <div id="container-body">

        <div class="ocean">

            <div class="wave">

            </div>

            <div class="wave">

            </div>

        </div>

        <!-- Log In Form Section -->

        <section>

            <div id="message"></div>

            <div class="container" id="container">

                <div class="form-container sign-up-container">

                    <form action="#">

                        <h1 class="form-head">Sign Up</h1>

                        <div class="social-container">

                            <a id="githubredirect-signup" class="social"><i class="fa-github fab"></i></a>

                            <a id="facebookredirect-signup" class="social"><i class="fa-facebook fab"></i></a>

                            <a id="googleredirect-signup" class="social"><i class="fa-google fab"></i></a>

                        </div>

                        <span>Or use your Email for registration</span>

                        <label>

                            <input id="name" type="text" name="username" placeholder="Name" />

                        </label>

                        <label>

                            <input id="email" type="email" name="email" placeholder="Email" required />

                        </label>

                        <label>

                            <input id="password" type="password" name="password" placeholder="Password" required />

                        </label>

                        <button class="bg-green-800 pt-3 pr-8 pb-3 pl-8 text-white" id="signup-btn" name="signup">Sign

                            Up</button>

                    </form>

                </div>

                <div class="form-container sign-in-container">

                    <form action="#">

                        <h1 class="form-head">Log In</h1>

                        <div class="social-container">

                            <a id="githubredirect-signin" class="social"><i class="fa-github fab"></i></a>

                            <a id="facebookredirect-signin" class="social"><i class="fa-facebook fab"></i></a>

                            <a id="googleredirect-signin" class="social"><i class="fa-google fab"></i></a>

                        </div>

                        <span> Or log In using E-Mail Address</span>

                        <label>

                            <input type="email" id="login\_email" name="login\_email" placeholder="Email" required />

                        </label>

                        <label>

                            <input type="password" id="login\_password" name="login\_password" placeholder="Password"

                                required />

                        </label>

                        <a href="#">Forgot your password?</a>

                        <button name="login" id="signin" class="bg-green-800 pt-3 pr-8 pb-3 pl-8 text-white">Log

                            In</button>

                    </form>

                </div>

                <div class="overlay-container">

                    <div class="overlay">

                        <div class="overlay-left overlay-panel">

                            <h1>Log in</h1>

                            <p>Log In here if you already have an account </p>

                            <button class="mt-5 ghost" id="signIn">Log In</button>

                        </div>

                        <div class="overlay-right overlay-panel">

                            <h1>Create, Account!</h1>

                            <p>Sign up if you still don't have an account ... </p>

                            <button class="mt-5 ghost" id="signUp">Sign Up</button>

                        </div>

                    </div>

                </div>

            </div>

        </section>

    </div>

4.**DashBoard Page :**

   <div class="container">

        <h1>Dashboard</h1>

        <li>

            <img id="avatar" type="button" data-dropdown-toggle="userDropdown" data-dropdown-placement="bottom-start"

                class="rounded-full w-20 h-20 cursor-pointer ring-[green] ring-2"

                src="https://media.istockphoto.com/id/1130884625/vector/user-member-vector-icon-for-ui-user-interface-or-profile-face-avatar-app-in-circle-design.jpg?s=612x612&w=0&k=20&c=1ky-gNHiS2iyLsUPQkxAtPBWH1BZt0PKBB1WBtxQJRE="

                alt="User dropdown">

            <label for="file" class="block relative m-2 px-4 py-2">

                <h2>Change Profile Pic</h2>

                <input class="cursor-pointer" accept="image/png, image/jpeg" type="file" id="photo" /></br>

            </label>

            <div class="flex justify-center m-2">

                <button id="btn-profile-dashboard"

                    class="block relative bg-white hover:bg-[#5ffc16] rounded text-black transition-all duration-300">Upload</button>

            </div>

            <hr color="white">

        </li>

        <div class="userdetailsdashboard">

            <li id="email-dashboard">Email: Dummy@gmail.com</li>

            <textarea name="userpersonal" class="mt-2 p-3 text-black" id="" cols="40" rows="2"

                placeholder="Personal Information"></textarea>

        </div>

        <div class="controls-dashboard">

            <a href="/history.html"><button

                    class="block relative bg-white hover:bg-[#5ffc16] px-4 py-2 rounded text-black transition-all duration-300">Histroy

                    Tab</button></a>

        </div>

    </div>

**Backend**

1. **Firebase Authentication For Anonymous Sign in & Login :**
2. const signupBtn = document.getElementById("signup-btn")
3. signupBtn.addEventListener("click", (e) => {
4. e.preventDefault()
5. const name = document.getElementById("name").value
6. const email = document.getElementById("email").value
7. const password = document.getElementById("password").value
8. const toastMsg = document.getElementById("toast-default")
9. createUserWithEmailAndPassword(auth, email, password)
10. .then((userCredential) => {
11. // Signed up
12. const user = userCredential.user;
13. // ...
14. toastMsg.innerHTML = `<div
15. class="inline-flex flex-shrink-0 justify-center items-center bg-blue-100 dark:bg-blue-800 rounded-lg w-8 h-8 text-blue-500 dark:text-blue-200">
16. <svg class="w-4 h-4" aria-hidden="true" xmlns="http://www.w3.org/2000/svg" fill="none" viewBox="0 0 18 20">
17. <path stroke="currentColor" stroke-linecap="round" stroke-linejoin="round" stroke-width="2"
18. d="M15.147 15.085a7.159 7.159 0 0 1-6.189 3.307A6.713 6.713 0 0 1 3.1 15.444c-2.679-4.513.287-8.737.888-9.548A4.373 4.373 0 0 0 5 1.608c1.287.953 6.445 3.218 5.537 10.5 1.5-1.122 2.706-3.01 2.853-6.14 1.433 1.049 3.993 5.395 1.757 9.117Z" />
19. </svg>
20. <span class="sr-only">Fire icon</span>
21. </div>
22. <div class="font-normal text-sm ms-3">User Created !</div>
23. <button type="button"
24. class="inline-flex justify-center items-center bg-white hover:bg-gray-100 dark:hover:bg-gray-700 dark:bg-gray-800 -mx-1.5 -my-1.5 p-1.5 rounded-lg w-8 h-8 text-gray-400 hover:text-gray-900 dark:hover:text-white ms-auto focus:ring-2 focus:ring-gray-300 dark:text-gray-500"
25. data-dismiss-target="#toast-default" aria-label="Close">
26. <span class="sr-only">Close</span>
27. <svg class="w-3 h-3" aria-hidden="true" xmlns="http://www.w3.org/2000/svg" fill="none" viewBox="0 0 14 14">
28. <path stroke="currentColor" stroke-linecap="round" stroke-linejoin="round" stroke-width="2"
29. d="m1 1 6 6m0 0 6 6M7 7l6-6M7 7l-6 6" />
30. </svg>
31. </button>`
32. toastMsg.classList.add("toastanime")
33. setTimeout(() => {
34. toastMsg.classList.remove("toastanime")
35. }, 3000)
36. set(ref(database, 'user/' + name), {
37. name: name,
38. email: email,
39. password: "Hidden"
40. })
41. setTimeout(() => {
42. window.location.href = "/LandingPage/landing-page.html"
43. }, 4000)
44. })
45. .catch((error) => {
46. const errorCode = error.code;
47. const errorMessage = error.message;
48. // ..
49. toastMsg.innerHTML = `<div
50. class="inline-flex flex-shrink-0 justify-center items-center bg-blue-100 dark:bg-blue-800 rounded-lg w-8 h-8 text-blue-500 dark:text-blue-200">
51. <svg class="w-4 h-4" aria-hidden="true" xmlns="http://www.w3.org/2000/svg" fill="none" viewBox="0 0 18 20">
52. <path stroke="currentColor" stroke-linecap="round" stroke-linejoin="round" stroke-width="2"
53. d="M15.147 15.085a7.159 7.159 0 0 1-6.189 3.307A6.713 6.713 0 0 1 3.1 15.444c-2.679-4.513.287-8.737.888-9.548A4.373 4.373 0 0 0 5 1.608c1.287.953 6.445 3.218 5.537 10.5 1.5-1.122 2.706-3.01 2.853-6.14 1.433 1.049 3.993 5.395 1.757 9.117Z" />
54. </svg>
55. <span class="sr-only">Fire icon</span>
56. </div>
57. <div class="font-normal text-sm ms-3">${errorMessage}</div>
58. <button type="button"
59. class="inline-flex justify-center items-center bg-white hover:bg-gray-100 dark:hover:bg-gray-700 dark:bg-gray-800 -mx-1.5 -my-1.5 p-1.5 rounded-lg w-8 h-8 text-gray-400 hover:text-gray-900 dark:hover:text-white ms-auto focus:ring-2 focus:ring-gray-300 dark:text-gray-500"
60. data-dismiss-target="#toast-default" aria-label="Close">
61. <span class="sr-only">Close</span>
62. <svg class="w-3 h-3" aria-hidden="true" xmlns="http://www.w3.org/2000/svg" fill="none" viewBox="0 0 14 14">
63. <path stroke="currentColor" stroke-linecap="round" stroke-linejoin="round" stroke-width="2"
64. d="m1 1 6 6m0 0 6 6M7 7l6-6M7 7l-6 6" />
65. </svg>
66. </button>`
67. toastMsg.classList.add("toastanime")
68. setTimeout(() => {
69. toastMsg.classList.remove("toastanime")
70. }, 3000)
71. });
72. })
73. const signinBtn = document.getElementById("signin")
74. signinBtn.addEventListener("click", (e) => {
75. e.preventDefault()
76. const login\_email = document.getElementById("login\_email").value
77. const login\_password = document.getElementById("login\_password").value
78. const toastMsg = document.getElementById("toast-default")
79. signInWithEmailAndPassword(auth, login\_email, login\_password)
80. .then((userCredential) => {
81. // Signed up
82. const user = userCredential.user;
83. // ...
84. toastMsg.innerHTML = `<div
85. class="inline-flex flex-shrink-0 justify-center items-center bg-blue-100 dark:bg-blue-800 rounded-lg w-8 h-8 text-blue-500 dark:text-blue-200">
86. <svg class="w-4 h-4" aria-hidden="true" xmlns="http://www.w3.org/2000/svg" fill="none" viewBox="0 0 18 20">
87. <path stroke="currentColor" stroke-linecap="round" stroke-linejoin="round" stroke-width="2"
88. d="M15.147 15.085a7.159 7.159 0 0 1-6.189 3.307A6.713 6.713 0 0 1 3.1 15.444c-2.679-4.513.287-8.737.888-9.548A4.373 4.373 0 0 0 5 1.608c1.287.953 6.445 3.218 5.537 10.5 1.5-1.122 2.706-3.01 2.853-6.14 1.433 1.049 3.993 5.395 1.757 9.117Z" />
89. </svg>
90. <span class="sr-only">Fire icon</span>
91. </div>
92. <div class="font-normal text-sm ms-3">User Logged In !</div>
93. <button type="button"
94. class="inline-flex justify-center items-center bg-white hover:bg-gray-100 dark:hover:bg-gray-700 dark:bg-gray-800 -mx-1.5 -my-1.5 p-1.5 rounded-lg w-8 h-8 text-gray-400 hover:text-gray-900 dark:hover:text-white ms-auto focus:ring-2 focus:ring-gray-300 dark:text-gray-500"
95. data-dismiss-target="#toast-default" aria-label="Close">
96. <span class="sr-only">Close</span>
97. <svg class="w-3 h-3" aria-hidden="true" xmlns="http://www.w3.org/2000/svg" fill="none" viewBox="0 0 14 14">
98. <path stroke="currentColor" stroke-linecap="round" stroke-linejoin="round" stroke-width="2"
99. d="m1 1 6 6m0 0 6 6M7 7l6-6M7 7l-6 6" />
100. </svg>
101. </button>`
102. toastMsg.classList.add("toastanime")
103. setTimeout(() => {
104. toastMsg.classList.remove("toastanime")
105. }, 3000)
106. setTimeout(() => {
107. window.location.href = "/LandingPage/landing-page.html"
108. }, 4000)
109. })
110. .catch((error) => {
111. const errorCode = error.code;
112. const errorMessage = error.message;
113. toastMsg.innerHTML = `<div
114. class="inline-flex flex-shrink-0 justify-center items-center bg-blue-100 dark:bg-blue-800 rounded-lg w-8 h-8 text-blue-500 dark:text-blue-200">
115. <svg class="w-4 h-4" aria-hidden="true" xmlns="http://www.w3.org/2000/svg" fill="none" viewBox="0 0 18 20">
116. <path stroke="currentColor" stroke-linecap="round" stroke-linejoin="round" stroke-width="2"
117. d="M15.147 15.085a7.159 7.159 0 0 1-6.189 3.307A6.713 6.713 0 0 1 3.1 15.444c-2.679-4.513.287-8.737.888-9.548A4.373 4.373 0 0 0 5 1.608c1.287.953 6.445 3.218 5.537 10.5 1.5-1.122 2.706-3.01 2.853-6.14 1.433 1.049 3.993 5.395 1.757 9.117Z" />
118. </svg>
119. <span class="sr-only">Fire icon</span>
120. </div>
121. <div class="font-normal text-sm ms-3">${errorMessage}</div>
122. <button type="button"
123. class="inline-flex justify-center items-center bg-white hover:bg-gray-100 dark:hover:bg-gray-700 dark:bg-gray-800 -mx-1.5 -my-1.5 p-1.5 rounded-lg w-8 h-8 text-gray-400 hover:text-gray-900 dark:hover:text-white ms-auto focus:ring-2 focus:ring-gray-300 dark:text-gray-500"
124. data-dismiss-target="#toast-default" aria-label="Close">
125. <span class="sr-only">Close</span>
126. <svg class="w-3 h-3" aria-hidden="true" xmlns="http://www.w3.org/2000/svg" fill="none" viewBox="0 0 14 14">
127. <path stroke="currentColor" stroke-linecap="round" stroke-linejoin="round" stroke-width="2"
128. d="m1 1 6 6m0 0 6 6M7 7l6-6M7 7l-6 6" />
129. </svg>
130. </button>`
131. toastMsg.classList.add("toastanime")
132. setTimeout(() => {
133. toastMsg.classList.remove("toastanime")
134. }, 3000)
135. });
136. })
137. **Firebase Authentication with Google Authentication**

const googleredirect = document.getElementById("googleredirect-signin")

const googleredirectSignup = document.getElementById("googleredirect-signup")

googleredirect.addEventListener('click', (e) => {

    signInWithRedirect(auth, provider);

    getRedirectResult(auth)

        .then((result) => {

            // This gives you a Google Access Token. You can use it to access Google APIs.

            const credential = GoogleAuthProvider.credentialFromResult(result);

            const token = credential.accessToken;

            // The signed-in user info.

            const user = result.user;

            // IdP data available using getAdditionalUserInfo(result)

            // ...

            // console.log(user)

            setTimeout(() => {

                window.location.href = "/LandingPage/landing-page.html"

            }, 3000)

        }).catch((error) => {

            // Handle Errors here.

            const errorCode = error.code;

            const errorMessage = error.message;

            // The email of the user's account used.

            const email = error.customData.email;

            // The AuthCredential type that was used.

            const credential = GoogleAuthProvider.credentialFromError(error);

            // ...

            // console.log(errorMessage, email, credential)

        });

})

googleredirectSignup.addEventListener('click', (e) => {

    signInWithRedirect(auth, provider);

    getRedirectResult(auth)

        .then((result) => {

            // This gives you a Google Access Token. You can use it to access Google APIs.

            const credential = GoogleAuthProvider.credentialFromResult(result);

            const token = credential.accessToken;

            // The signed-in user info.

            const user = result.user;

            // IdP data available using getAdditionalUserInfo(result)

            // ...

            // console.log(user)

        }).catch((error) => {

            // Handle Errors here.

            const errorCode = error.code;

            const errorMessage = error.message;

            // The email of the user's account used.

            const email = error.customData.email;

            // The AuthCredential type that was used.

            const credential = GoogleAuthProvider.credentialFromError(error);

            // ...

            // console.log(errorMessage, email, credential)

        });

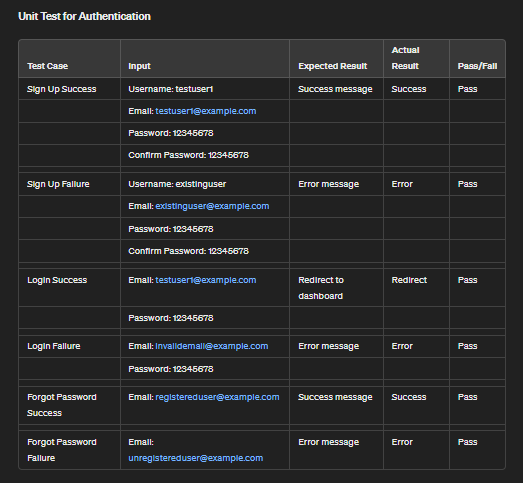
})

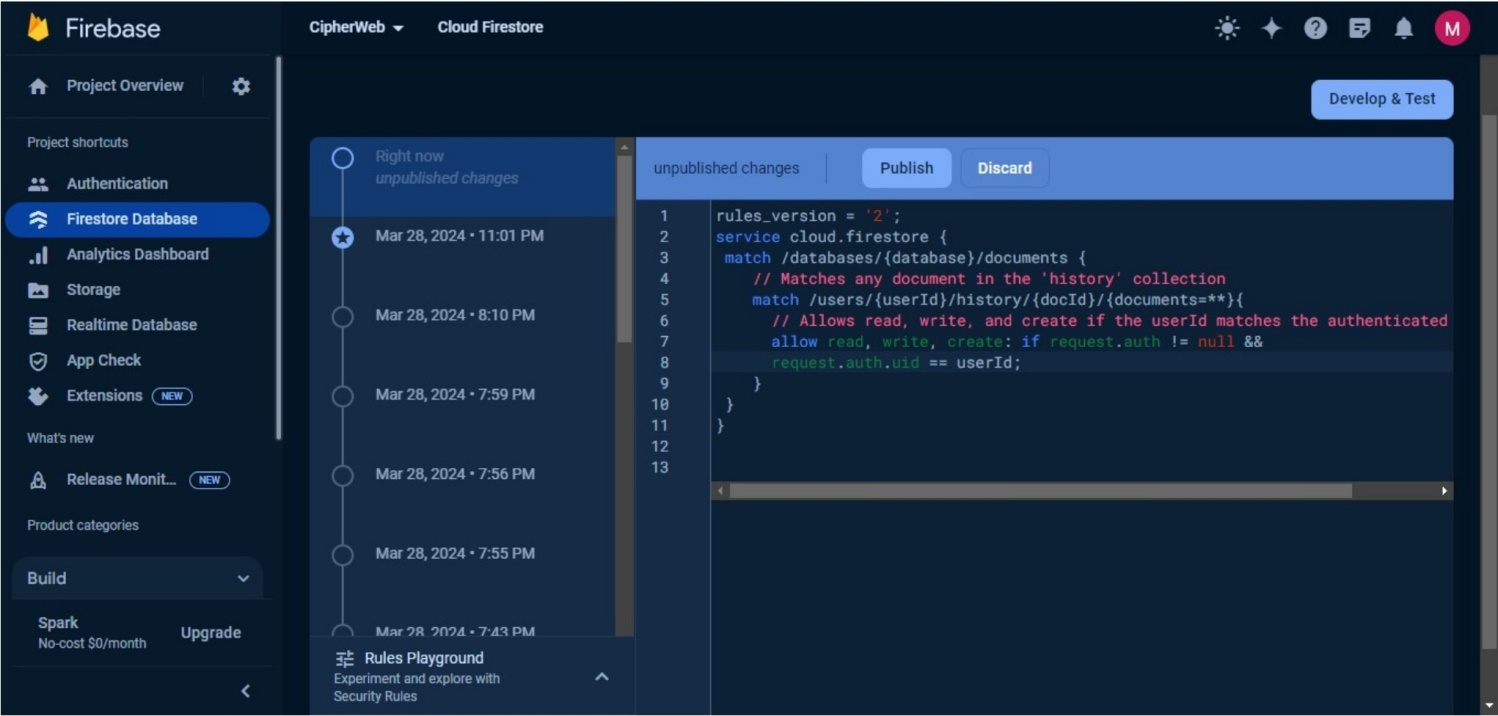
* 1. Testing Approach
     1. **Unit Testing**

**SYSTEM TESTING**

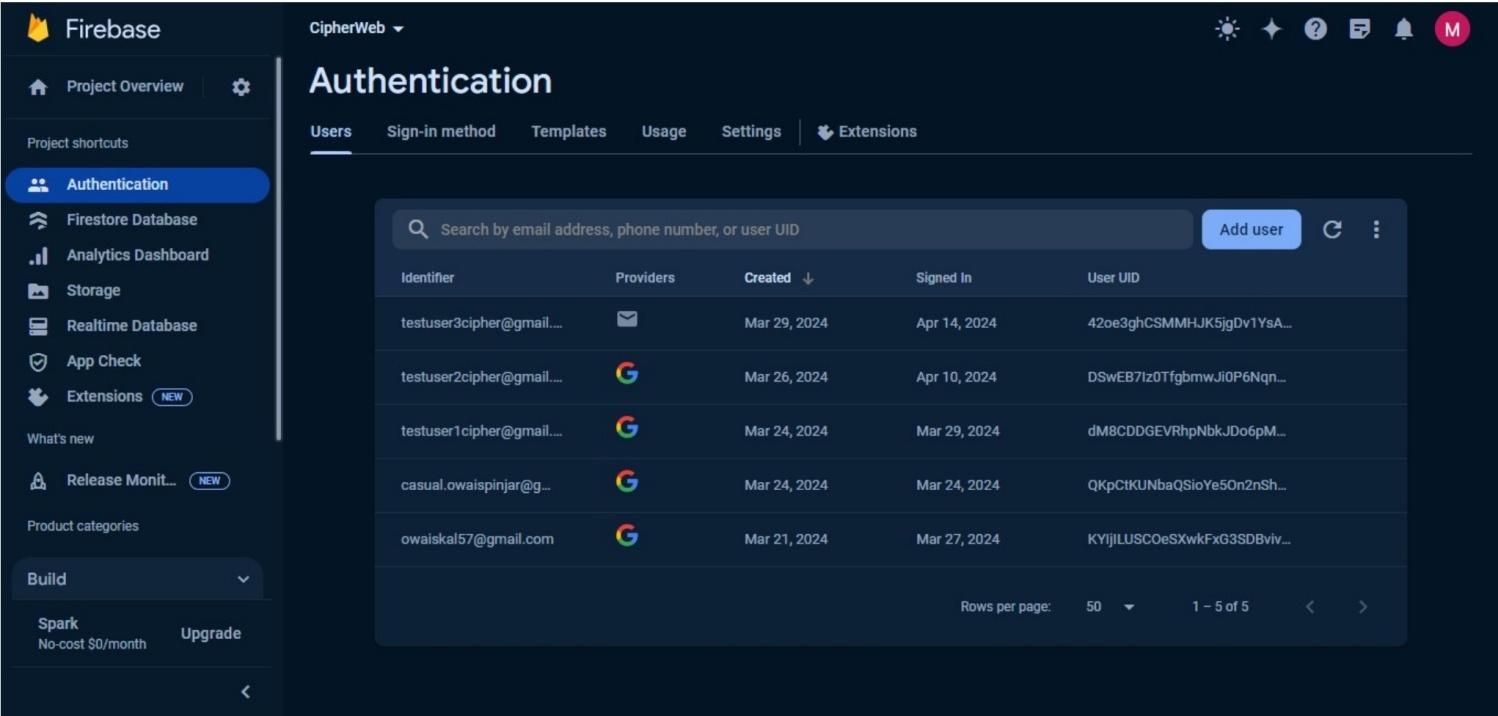
###### Test for User Login:

Testing admin login form: This Form is used for login of user. In this we enter user email and password if both are correct then page will open otherwise if any data is wrong it will redirect to login page and ask for user email and password.

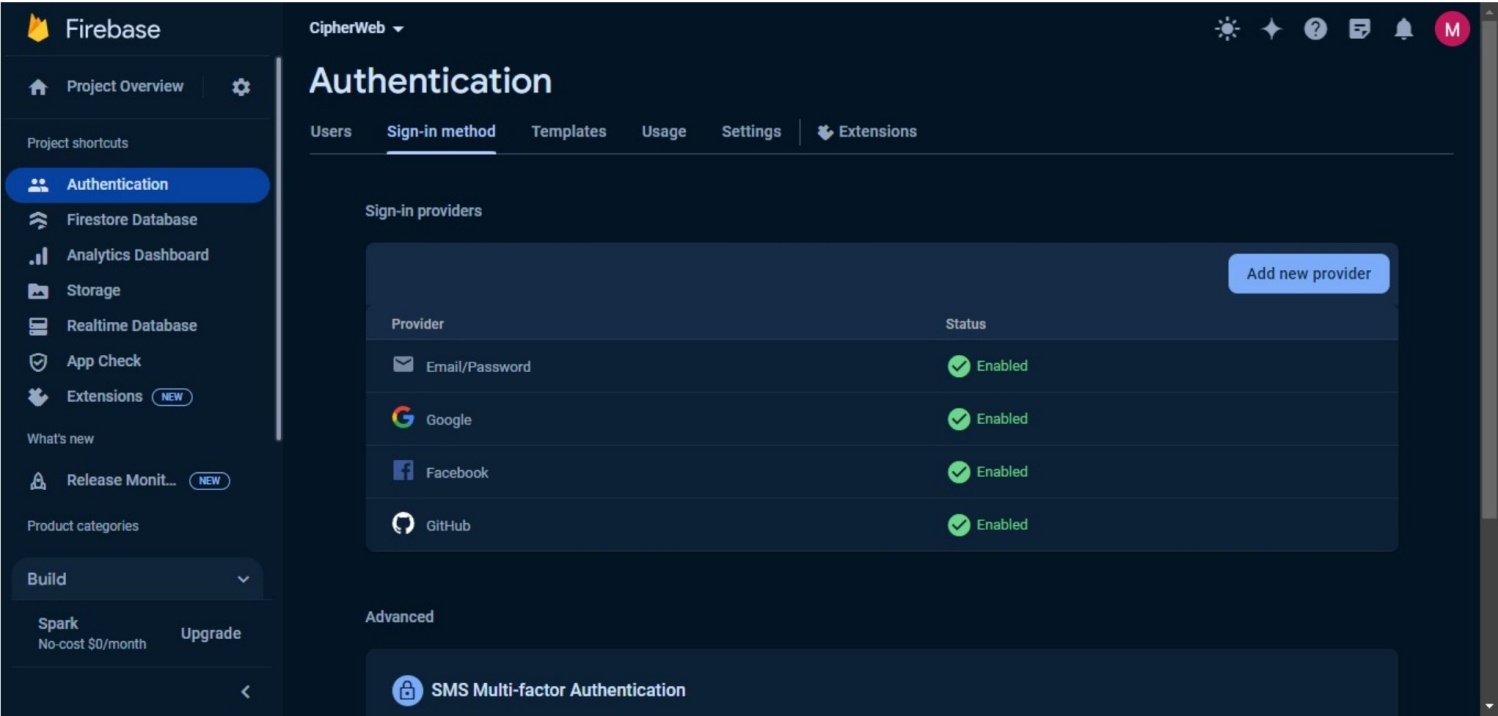


Firebase Authentication Rule

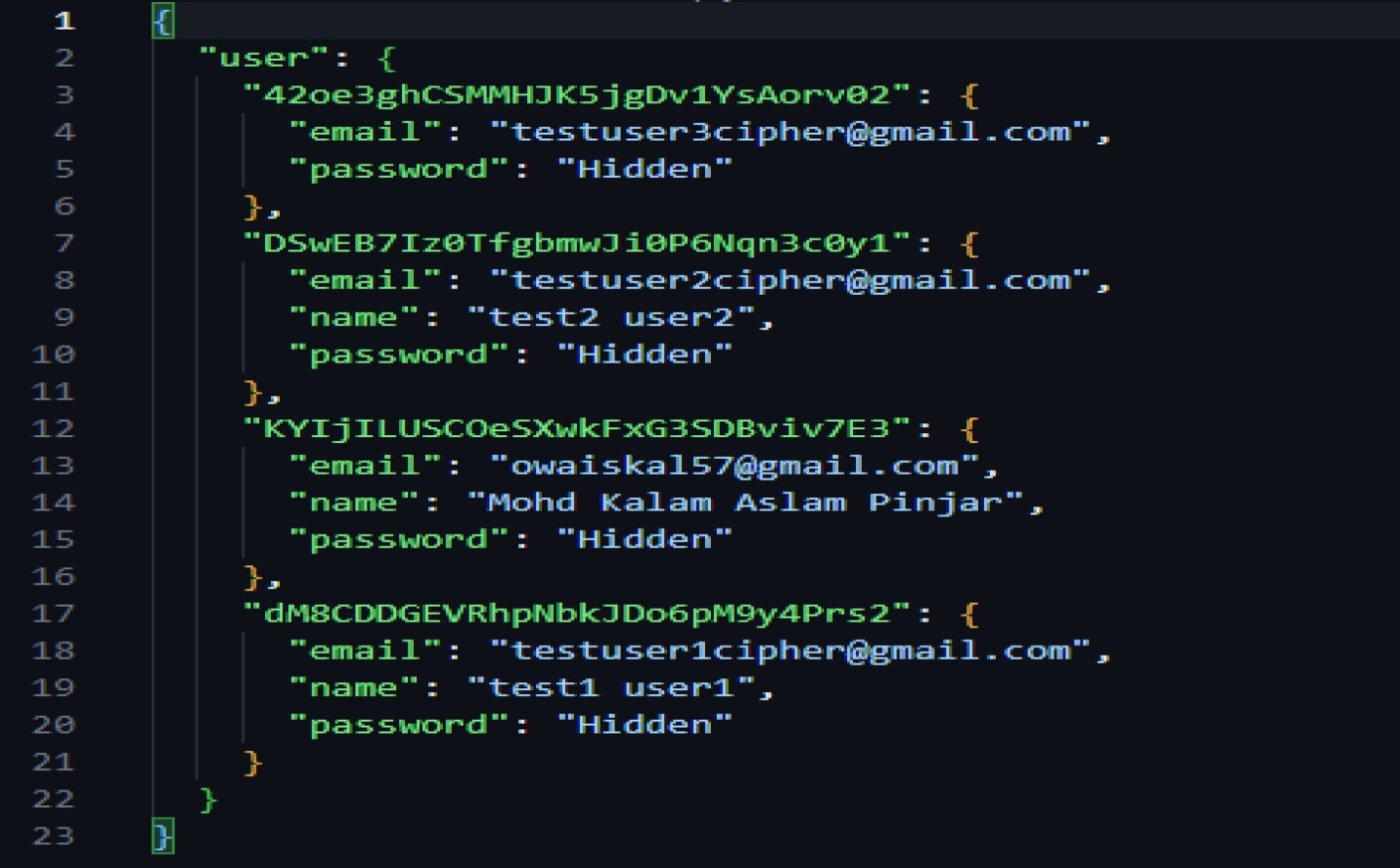
User Table in Firebase Admin Console



Firebase Authentication Provider



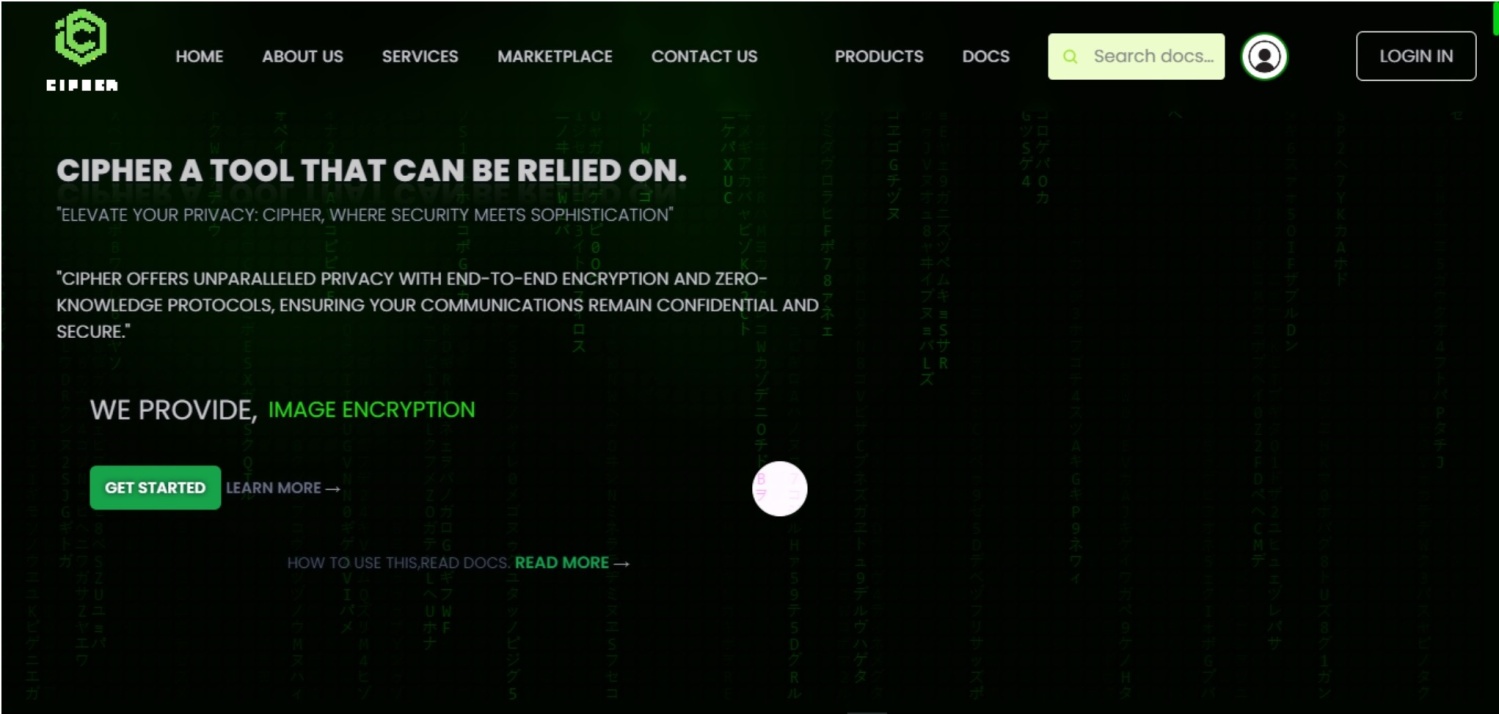
Firebase Database Data(JSON Format)



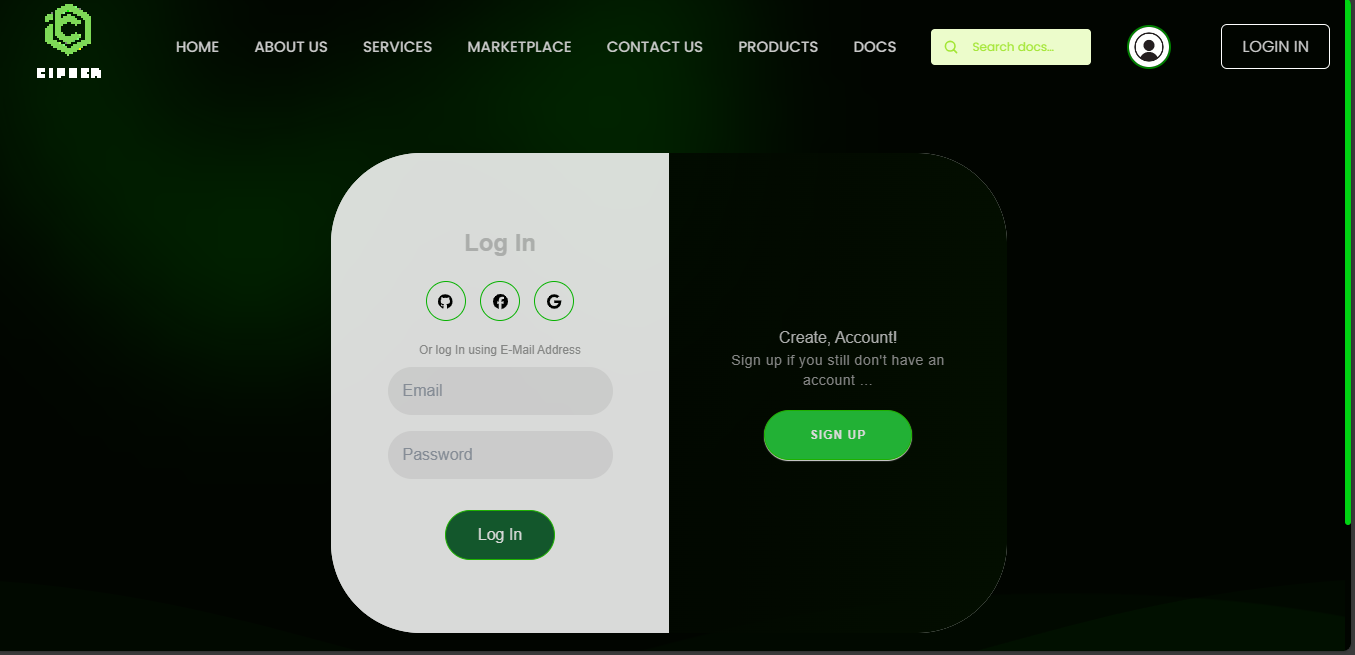
#### Chapter 5

#### Results and Discussions

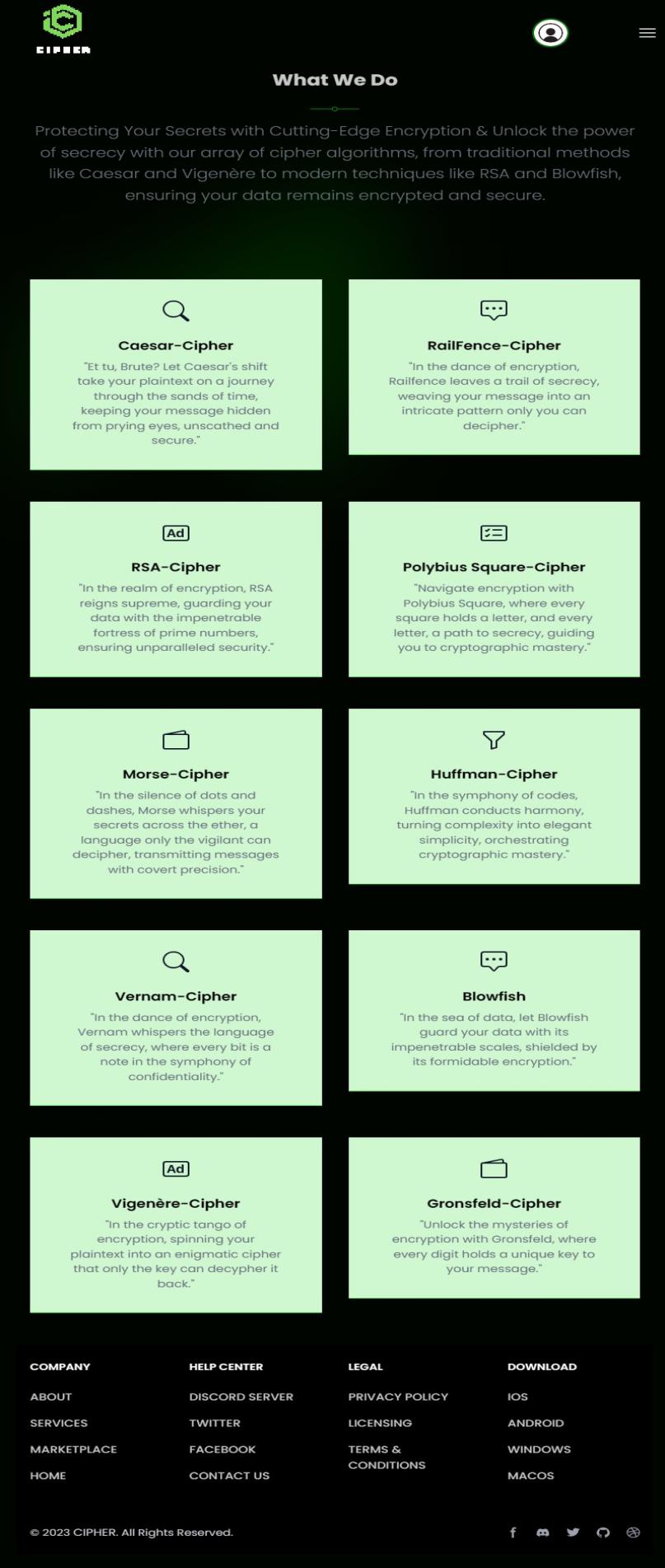
1. Homepage



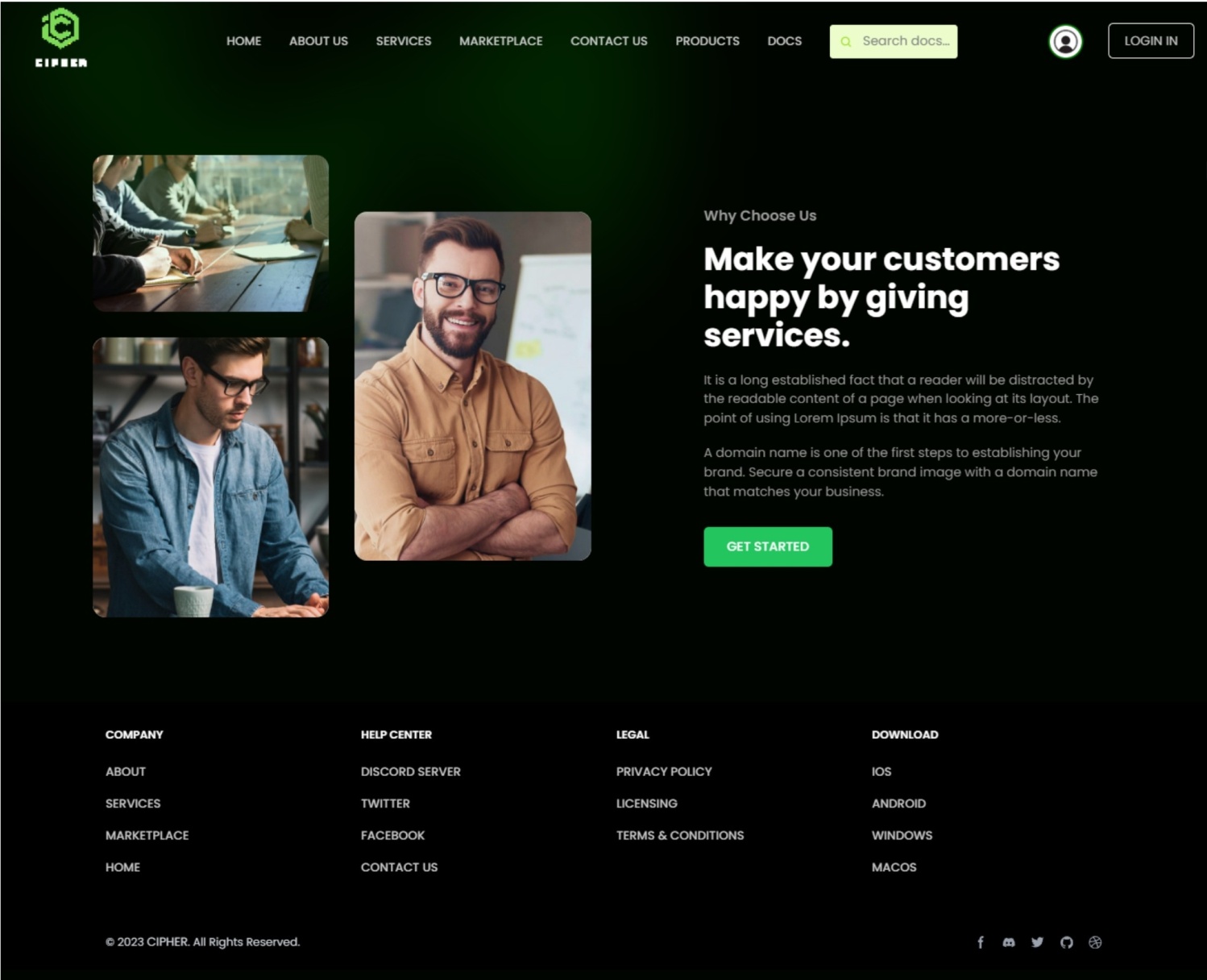
1. **Login & Sign-in Page**



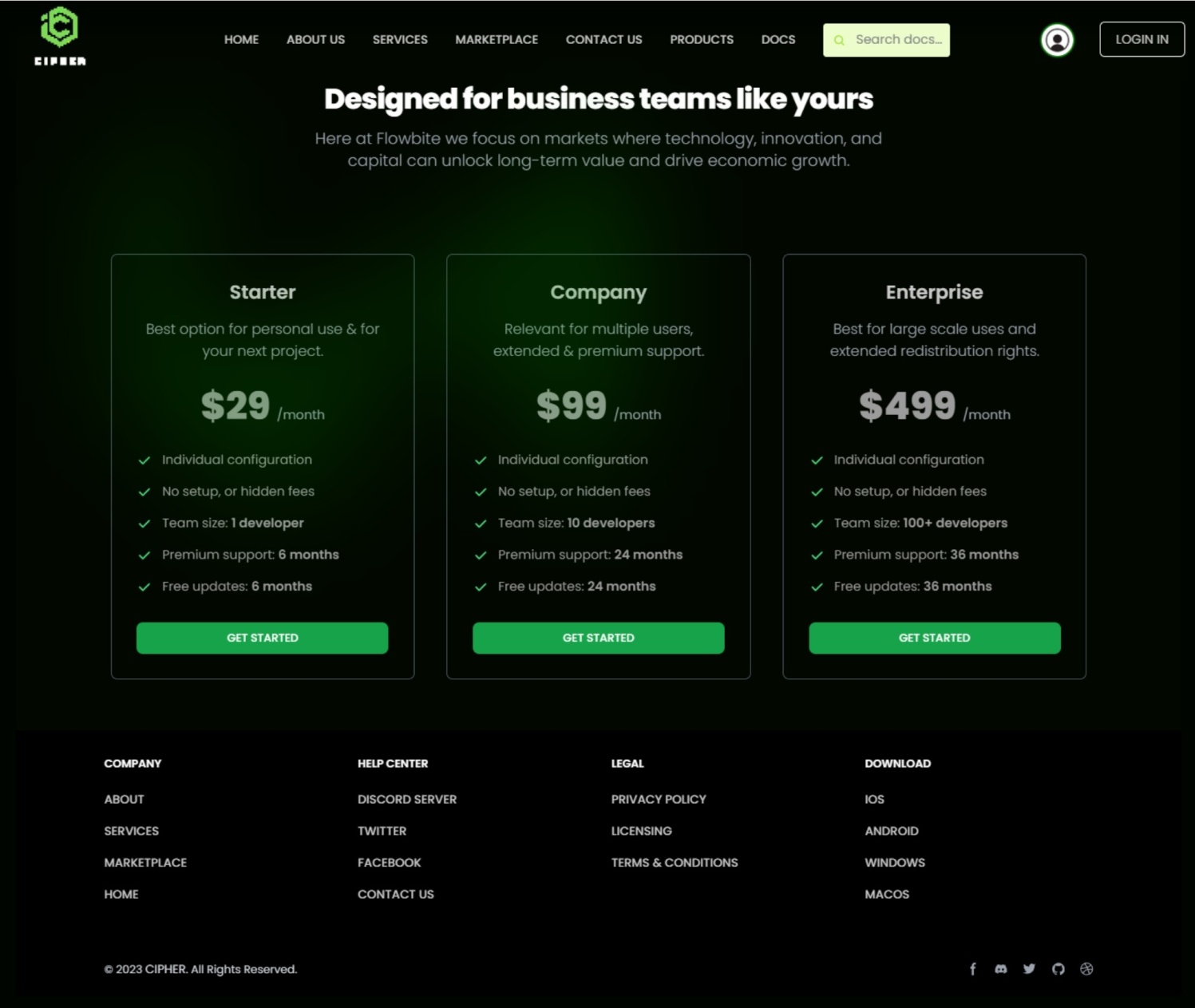
1. **Service Page**



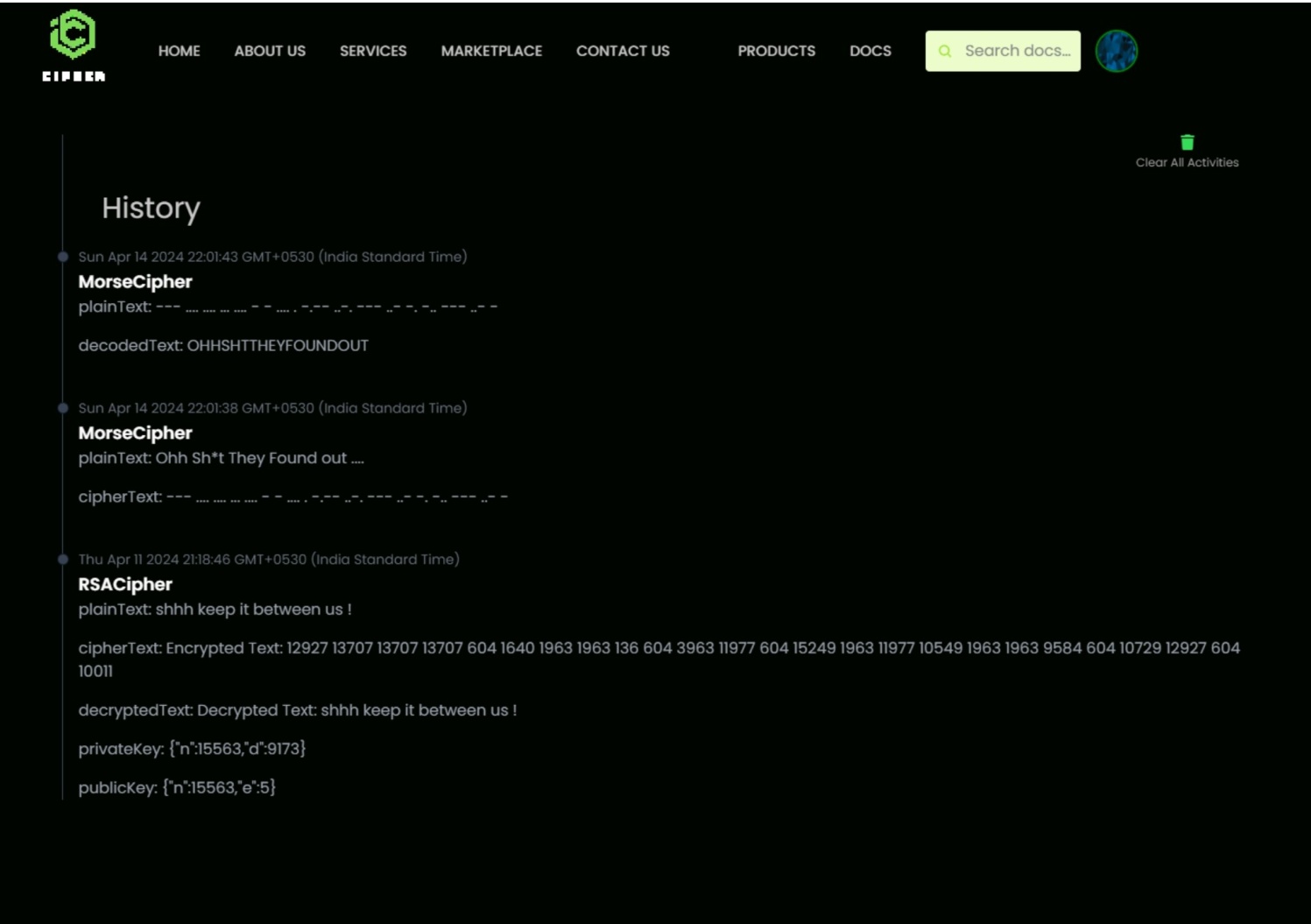
1. About Page



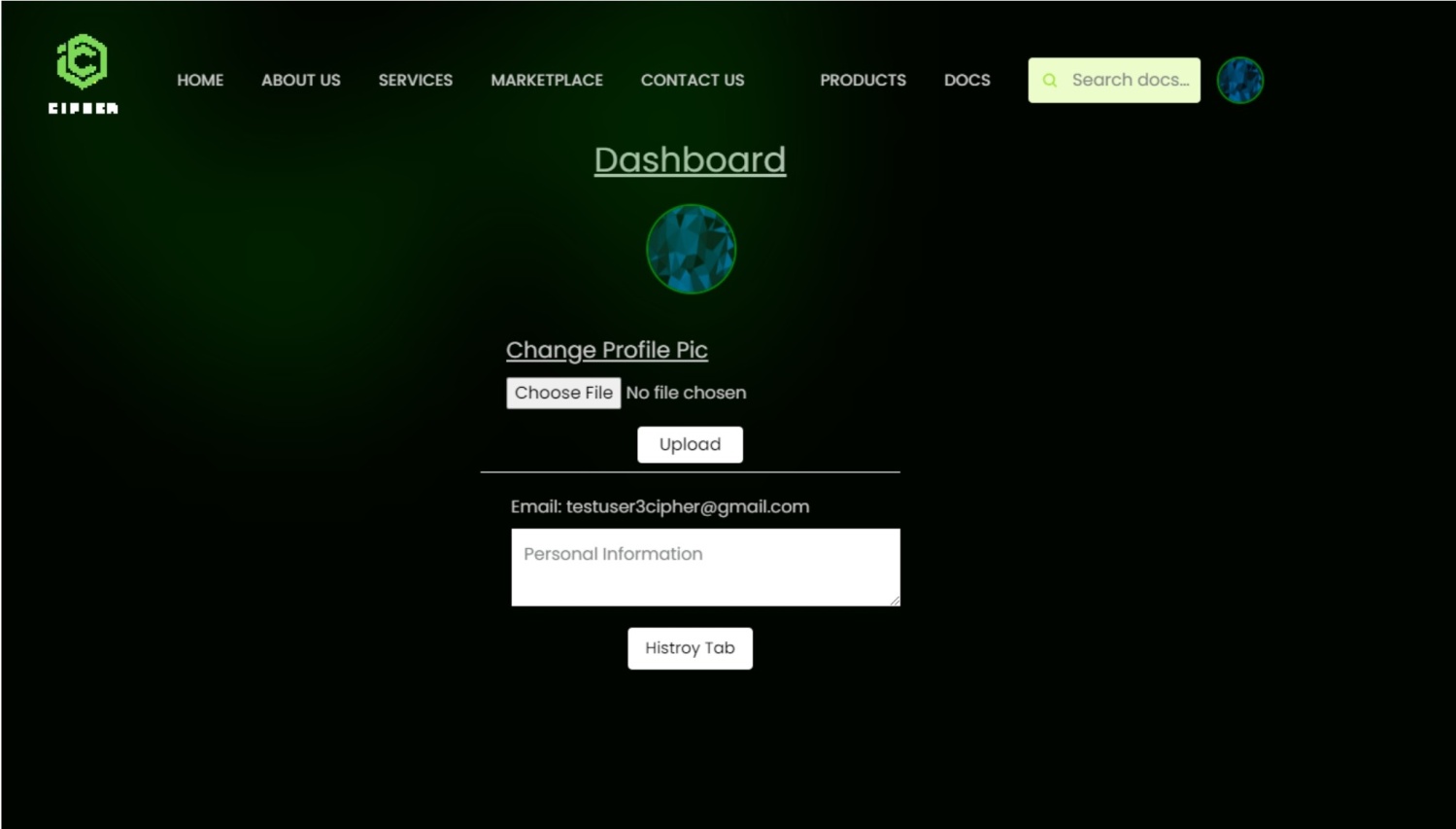
1. **Marketplace Page**



1. **History Page**



1. **Dashboard Page**



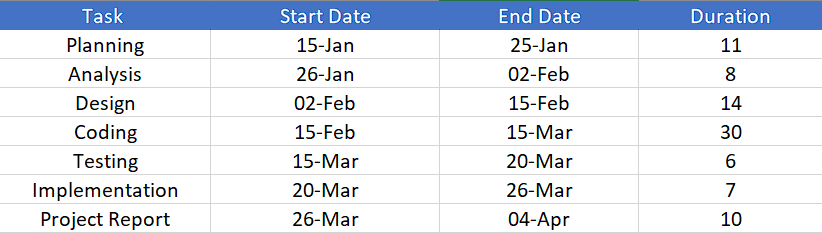
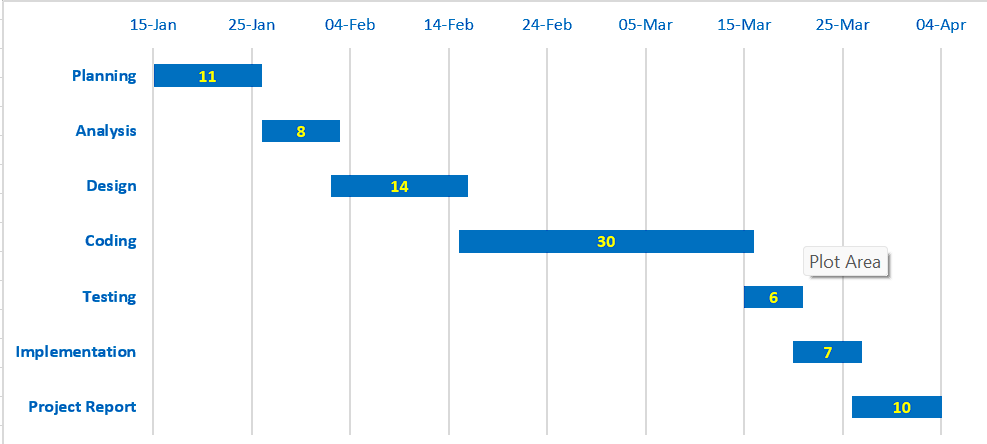
### SYSTEM PLANNING (GRANTT CHART).

System planning involves outlining the project's scope, objectives, tasks, and timelines to ensure efficient execution and successful completion. In the context of a Gantt chart, system planning refers to the process of creating a visual representation of the project timeline, task dependencies, and resource allocation.

The Gantt chart serves as a value able tool in system planning by providing a clear overview of the project schedule and task assignments. It helps project managers and team members understand the sequence of tasks, identify critical milestones, and allocate resources effectively. Additionally, the Gantt chart allows for the identification of potential bottlenecks or delays, enabling proactive mitigation measures to be implemented.

During system planning, key considerations include:

1. Task Identification: Determine the specific tasks required to achieve project objectives, breaking down the project into manageable components.
2. Task Sequencing: Establish the sequence in which tasks need to be completed, considering dependencies and logical order.
3. Resource Allocation: Assign resources, such as personnel, equipment, and budget, to each task based on availability and requirements.
4. Timeline Estimation: Estimate the duration of each task and define start and end dates, considering factors such as complexity, dependencies, and resource constraints.
5. Milestone Definition: Identify key milestones or checkpoints in the project timeline to track progress and ensure alignment with project objectives.
6. Task Dependencies: Determine dependencies between tasks, such as those that must be completed before others can start or those that can be performed concurrently.



## Chapter 6

## Conclusion and Future

## Work

In conclusion, the development of CipherWeb, a web-based cryptographic platform, has been a significant endeavour aimed at providing users with secure and user-friendly cryptographic functionalities. Throughout the development process, several key achievements have been made:

1. Comprehensive Feature Set: CipherWeb offers users a diverse selection of 10 cryptographic algorithms, ranging from classic ciphers to modern encryption methods. Users can securely encrypt and decrypt messages, manage cryptographic keys, and track their encryption histories within the platform.
2. Intuitive User Interface: The user interface of CipherWeb is designed to be intuitive and responsive, ensuring of navigation and usage across desktop and mobile devices. Tailwind CSS and JavaScript Canvas are utilized to enhance the visual aesthetics and interactivity of the platform.
3. Robust Security Measures: Security is a top priority in CipherWeb, with encryption techniques, user authentication mechanisms, and secure communication protocols implemented to safe guard user data and communications from unauthorized access and interception.
4. Seamless Integration with Firebase: CipherWeb seamlessly integrates with Firebase backend services, including authentication, real-time database management, and cloud storage. This integration enhances the reliability, scalability, and performance of the platform, ensuring a seamless user experience.

**Future Work:**

While CipherWeb has achieved significant milestones, there are several areas for future improvement and expansion:

1. Enhanced Algorithm Support: Continuously expand the selection of cryptographic algorithms offered by CipherWeb to include additional algorithms and encryption techniques based on user feedback and emerging cryptographic standards.
2. Advanced Security Features: Implement advanced security features such as multi-factor authentication, end-to-end encryption, and secure key management to further enhance the security posture of CipherWeb and protect user data from evolving threats.
3. Improved User Experience: Continuously refine the user interface and user experience of CipherWeb based on user feedback and usability testing. Incorporate user-friendly features, tutorials, and interactive guides to empower users to utilize cryptographic functionalities effectively.
4. Integration with External Systems: Explore opportunities for integrating CipherWeb with external systems and platforms, such as messaging applications, email clients, and collaboration tools, to expand its utility and reach a broader user base.
5. Research and Education: Serve as a platform for research and education in the field of cryptography by providing resources, tutorials, and educational content on cryptographic algorithms, techniques, and best practices.

In summary, CipherWeb represents a significant step forward in the development of cryptographic tools and platforms, offering users a secure, intuitive, and versatile solution for protecting their digital privacy and security. Through ongoing innovation and collaboration, CipherWeb will continue to evolve and adapt to meet the changing needs and challenges of the digital landscape.

#### Chapter 7

**References**

1. Ferguson, N.Schneier, **B.**&Kohno,T.(2010).Cryptography Engineering: Design Principles and Practical Applications. Wiley.
2. Katz,**J.** & Lindell,Y.(2014).Introduction to Modern Cryptography .CRC Press.
3. Stallings W.(2017).Cryptography and Network Security: Principles and Practice. Pearson.
4. Tailwind CSS Documentation. Retrieved from: https://tailwindcss.com/docs
5. Firebase Documentation Retrieved from: https://firebase.google.com/docs
6. Node.js Documentation. Retrieved from: https://nodejs.org/en/docs/
7. Google Cloud Platform Documentation. Retrieved from: https://cloud.google.com/docs
8. Cryptography Stack Exchange. Retrieved from: https://crypto.stackexchange.com/
9. GitHub Repository for CipherWeb. Available at: [https://github.com/KalamPinjar/CipherWeb]