ENHANCED PRIVACY PRESERVING CONTENT BASED RETRIEVAL IN CLOUD REPOSITORIES

A PROJECT REPORT

Submitted by

S. PARTHASARATHI (REG.NO:821921104049)

R. SARAVANAKUMAR (REG.NO:821921104064)

M. SENTHILKUMARAN (REG.NO:821921104067)

I. VENKATESHWAR (REG.NO:821921104080)

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

ST. JOSEPH'S COLLEGE OF ENGINEERING AND TECHNOLOGY

THANJAVUR

ANNA UNIVERSITY: CHENNAI 600 025

MAY 2025

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project titled **"ENHANCED PRIVACY PRESERVING CONTENT BASED RETRIEVAL IN CLOUD REPOSITORIES"** is the bonafide certificate work of **Mr. PARTHASARATHI.S (821921104049), Mr. SARAVANAKUMAR.R (821921104064), Mr. SENTHILKUMARAN.M (821921104067), Mr. VENKATESHWAR.I (821921104080),** who carried out the project work under my supervision.

SIGNATURE

Mr. P. MUTHAMILSELVAN, M.E.,

HEAD OF THE DEPARTMENT

Department of Computer Science &Engg.

St.Joseph's College of Engg & Tech

Thanjavur

SIGNATURE

Mrs. K. AMUTHA, M.E.,

SUPERVISOR

Department of Computer Science &Engg.

St.Joseph's College of Engg & Tech

Thanjavur

Submitted for the project viva-voce examination held on …….....

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

Everything is possible by way of creator. So first of all we thanks the almighty for giving Grace and Knowledge.

Our sincere thanks to our Honourable Chairman **Rev. Fr. Dr. J. E. ARUL RAJ** for the opportunity and facilities given to us to carry out this project work.

Our sincere thanks to our beloved Administrator **Rev. Sr. P.MARIA ALANGARAM, DMI** for the opportunity and facilities given to us to carry out this project work.

We express our sincere and heartful thanks to our Principal, **Dr. I. NEETHI MANICKAM, M.TECH., Ph.D.,** for his kind support and encouragement.

We would also like to express our gratitude and thanks to **Mr. P. MUTHAMIL SELVAN, M.E.** Head of the Department of Computer Science and Engineering for the valuable guidance and excellent suggestions throughout this project.

We express our thanks to **Mrs. A. FRANCIS THIVYA, M.E.**, project coordinator for his wholehearted encouragement throughout this project.

We like to express our sincere gratitude to our internal project guide **Mrs. K. AMUTHA, M.E.,** for supporting us this project.

We hereby take this opportunity to thank my parents and friends who encouraged and helped me in numerous ways with regard and respect I best of the success of the project to all them.

ABSTRACT

With the exponential growth of data stored in cloud repositories, ensuring both efficient data retrieval and robust privacy protection has become a critical challenge. Traditional content-based retrieval systems often compromise user privacy by exposing sensitive data or search patterns to cloud service providers. This project proposes an enhanced privacy-preserving content-based retrieval framework tailored for cloud repositories. By integrating advanced cryptographic techniques such as homomorphic encryption and secure multi-party computation with efficient indexing and retrieval algorithms, the system allows users to perform content-based queries without revealing the content of the queries or the data itself to the cloud provider. Additionally, the proposed framework supports similarity-based searches, ensuring accurate retrieval of relevant data while maintaining confidentiality. Experimental results demonstrate the system's effectiveness in balancing retrieval efficiency, data security, and computational overhead, making it a viable solution for privacy-sensitive applications such as healthcare, legal archives, and personal data management in cloud environments.

சுருக்கம்

மேகக் களஞ்சியங்களில் சேமிக்கப்படும் தரவுகளின் அதிவேக வளர்ச்சியுடன், திறமையான தரவு மீட்டெடுப்பு மற்றும் வலுவான தனியுரிமை பாதுகாப்பு இரண்டையும் உறுதி செய்வது ஒரு முக்கியமான சவாலாக மாறியுள்ளது. பாரம்பரிய உள்ளடக்க அடிப்படையிலான மீட்டெடுப்பு அமைப்புகள் பெரும்பாலும் மேகக்கணி சேவை வழங்குநர்களுக்கு உணர்திறன் தரவு அல்லது தேடல் முறைகளை வெளிப்படுத்துவதன் மூலம் பயனர் தனியுரிமையை சமரசம் செய்கின்றன. இந்தத் திட்டம் மேகக்கணி களஞ்சியங்களுக்காக வடிவமைக்கப்பட்ட மேம்பட்ட தனியுரிமை-பாதுகாக்கும் உள்ளடக்க அடிப்படையிலான மீட்டெடுப்பு கட்டமைப்பை முன்மொழிகிறது. ஹோமோமார்பிக் குறியாக்கம் மற்றும் பாதுகாப்பான பல-கட்சி கணக்கீடு போன்ற மேம்பட்ட கிரிப்டோகிராஃபிக் நுட்பங்களை திறமையான அட்டவணைப்படுத்தல் மற்றும் மீட்டெடுப்பு வழிமுறைகளுடன் ஒருங்கிணைப்பதன் மூலம், இந்த அமைப்பு பயனர்கள் வினவல்களின் உள்ளடக்கத்தையோ அல்லது தரவையோ கிளவுட் வழங்குநருக்கு வெளிப்படுத்தாமல் உள்ளடக்க அடிப்படையிலான

வினவல்களைச் செய்ய அனுமதிக்கிறது. கூடுதலாக, முன்மொழியப்பட்ட கட்டமைப்பு ஒற்றுமை அடிப்படையிலான தேடல்களை ஆதரிக்கிறது, ரகசியத்தன்மையைப் பேணுகையில் தொடர்புடைய தரவை துல்லியமாக மீட்டெடுப்பதை உறுதி செய்கிறது. மீட்டெடுப்பு செயல்திறன், தரவு பாதுகாப்பு மற்றும் கணக்கீட்டு மேல்நிலை ஆகியவற்றை சமநிலைப்படுத்துவதில் அமைப்பின் செயல்திறனை சோதனை முடிவுகள் நிரூபிக்கின்றன, இது கிளவுட் சூழல்களில் சுகாதாரம், சட்ட காப்பகங்கள் மற்றும் தனிப்பட்ட தரவு மேலாண்மை போன்ற தனியுரிமை-உணர்திறன் பயன்பாடுகளுக்கு ஒரு சாத்தியமான தீர்வாக அமைகிறது.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

With the rapid proliferation of cloud computing technologies, vast volumes of data are now stored and accessed through cloud repositories. These platforms offer scalable storage and computational capabilities, enabling efficient content-based retrieval systems that support diverse applications such as healthcare, finance, and multimedia management. However, the growing reliance on cloud services raises significant concerns regarding data privacy and security, particularly when sensitive information is involved.

Traditional content-based retrieval methods often require data to be processed in plaintext, posing serious risks to confidentiality. While existing privacy-preserving techniques offer partial protection, they frequently compromise on retrieval accuracy, system efficiency, or user experience. Therefore, there is a pressing need for enhanced approaches that can balance robust privacy preservation with effective and accurate content-based retrieval.

This project addresses these challenges by proposing an enhanced privacy-preserving framework that leverages advanced encryption techniques, secure indexing mechanisms, and optimized similarity computation strategies. The proposed system ensures that user data remains confidential while enabling efficient and precise retrieval operations over encrypted data stored in the cloud. This framework aims to bridge the gap between data utility and privacy, paving the way for secure, scalable, and practical cloud-based information retrieval solutions.

CHAPTER 2

LITERATURE SURVEY

2.1 "Enhanced Security for Cloud Storage Using Hybrid Cryptography" by S.S.S. Sharma and S. Kumari [2022]

This paper proposes a hybrid cryptography-based enhanced cloud storage security paradigm that combines the AES encryption and decryption techniques. The proposed paradigm also includes a secure key management method that ensures the confidentiality and integrity of data kept in cloud storage. The study evaluates the proposed architecture's performance and shows how safe cloud storage it can be through testing and simulations.

2.2 "A Hybrid Encryption Scheme for Cloud Storage Security" by M.A. Khan, M.A. Rahman, and M. Islam [2021]

For the security of cloud storage, this paper proposes a hybrid encryption technique that combines symmetric and asymmetric encryption. The suggested method makes use of the RSA algorithms for decryption and encryption, respectively. The research evaluates the performance and security of the proposed system through simulations and testing, demonstrating how effectively it may allay security worries in cloud storage.

2.3 "Secure File Storage and Sharing in Cloud Computing Using Hybrid Cryptography" by P. Kumar and P. Singh [2020]

This research proposes a hybrid cryptography strategy for file sharing and storage in cloud computing. The model uses the encryption techniques of AES to ensure the integrity and secrecy of the data. The study demonstrates the effectiveness and security of the recommended strategy using tests and simulations.

2.4 Rohit Barvekar, Shrajal Behere, Yash Pounikar, and Ms. Anushka Gulhane, "Secure File Storage and Sharing in Cloud Computing using Hybrid Cryptography" (2019)

The recommended security measures that will prevent sensitive data from being used unlawfully will make the system more trustworthy. Keys will be used in the proposed method for both encryption and decryption.

2.5 The work "Secure File Storage and Sharing in Cloud Computing using Hybrid Cryptography" by A. Sharma and K. Sharma. [2018]

This research proposes a hybrid cryptography-based approach for cloud computing and secure file storage that combines AES and ECC encryption. The recommended method ensures confidentiality, integrity, and availability by utilising encryption and decryption methods. The study demonstrates the effectiveness and security of the recommended strategy using tests and simulations.

2.6 System Study

2.6.1 Existing System

Several techniques can enhance privacy-preserving in cloud repositories, including encryption, access control, and privacy-preserving computations. Encryption, like homomorphic encryption, allows data to be processed in a ciphertext state, preventing unauthorized access while enabling computations. Access control mechanisms, such as multi-factor authentication and role-based access control, restrict data access based on user identity and permissions. Privacy-preserving computation, such as privacy-preserving machine learning, enables computations on data while preserving its confidentiality. Additionally, Hash-Solomon coding can be used to divide data and distribute it across multiple locations for enhanced security and anonymity.

Drawbacks of existing systems:

1. Increased Complexity
2. Higher Computational Overhead
3. Potential Performance Degradation
4. Resource Intensive and Costly
5. Security Risks and Vulnerabilities
6. Loss of Certainty and Control
7. Challenges in Interoperability

2.6.2 Proposed System

A proposed system for enhanced privacy-preserving content-based retrieval in cloud repositories aims to provide secure and efficient search of images while protecting the privacy of both the image content and query requests. This can be achieved through techniques like searchable encryption, which allows similarity retrieval on encrypted data, and the use of deep CNN features extracted from images.

Advantages of the proposed system:

- Data Privacy
- Secure Search
- Efficient Retrieval
- Reduced Storage Costs
- Compliance with Regulations

2.6.3 System Specification

Hardware Requirements

Hard disk	:	1 TB
RAM	:	4 GB
Processor	:	Core i3
Monitor	:	15’’Color Monitor

Software Requirements

Front-End	:	HTML, CSS, and JS
Back end	:	Python Flask, MySQL
Operating System	:	Windows 10
IDE	:	Visual Studio Code

CHAPTER 3

SOFTWARE SPECIFICATION

3.1 Front End Tools Description

HTML, CSS, JS

An overview:

- HTML provides the basic structure of sites, which is enhanced and modified by other technologies like CSS and JavaScript.
- CSS is used to control presentation, formatting, and layout.
- JavaScript is used to control the behavior of different elements.

Now, let's go over each one individually to help you understand the roles each plays on a website and then we'll cover how they fit together. Let's start with good of' HTML.

HTML

HTML is at the core of every web page, regardless the complexity of a site or number of technologies involved. It's an essential skill for any web professional. It's the starting point for anyone learning how to create content for the web. And, luckily for us, it's surprisingly easy to learn.

Markup languages work in the same way as you just did when you labeled those content types, except they use code to do it -- specifically, they use HTML tags, also known as "elements." These tags have pretty intuitive names: Header tags, paragraph tags, image tags, and so on.

Every web page is made up of a bunch of these HTML tags denoting each type of content on the page. Each type of content on the page is "wrapped" in, i.e. surrounded by, HTML tags.

For example, the words you're reading right now are part of a paragraph. If I were coding this web page from scratch (instead of using the WYSIWG editor in HubSpot's COS), I would have started this paragraph with an opening paragraph tag: . The "tag" part is denoted by open brackets, and the letter "p" tells the computer that we're opening a paragraph instead of some other type of content.

Once a tag has been opened, all of the content that follows is assumed to be part of that tag until you "close" the tag. When the paragraph ends, I'd put a closing paragraph tag: . Notice that closing tags look exactly the same as opening tags, except there is a forward slash after the left angle bracket. Here's an example:

This is a paragraph.

Using HTML, you can add headings, format paragraphs, control line breaks, make lists, emphasize text, create special characters, insert images, create links, build tables, control some styling, and much more.

To learn more about coding in HTML, I recommend checking out our guide to basic HTML, and using the free classes and resources on codecademy -- but for now, let's move on to CSS.

CSS

CSS stands for Cascading Style Sheets. This programming language dictates how the HTML elements of a website should actually appear on the frontend of the page.

If HTML is the drywall, CSS is the paint.

Whereas HTML was the basic structure of your website, CSS is what gives your entire website its style.

Those slick colors, interesting fonts, and background images? All thanks to CSS. This language affects the entire mood and tone of a web page, making it an incredibly powerful tool -- and an important skill for web developers to learn. It's also what allows websites to adapt to different screen sizes and device types.

To show you what CSS does to a website, look at the following two screenshots. The first screenshot is my colleague's blog post, but shown in Basic HTML, and the second screenshot is that same blog post with HTML and CSS.

JavaScript

JavaScript is a more complicated language than HTML or CSS, and it wasn't released in beta form until 1995. Nowadays, JavaScript is supported by all modern web browsers and is used on almost every site on the web for more powerful and complex functionality.

JavaScript is particularly useful for assigning new identities to existing website elements, according to the decisions the user makes while visiting the page. For example, let's say you're building a landing page with a form you'd like to generates leads from by capturing information about a website visitor. You might have a "string" of JavaScript dedicated to the user's first name. That string might look something like this:

Then, after the website visitor enters his or her first name -- and any other information you require on the landing page -- and submits the form, this action updates the identity of the initially undefined "Firstname" element in your code. Here's how you might thank your website visitor by name in JavaScript:

para.textContent = 'Thanks, ' + First name + "! You can now download your ebook."

In the string of JavaScript above, the "First name" element has been assigned the first name of the website visitor, and will therefore produce his or her actual first name on the frontend of the webpage.

3.2 Back End Tools Description

Python

Python is a powerful and globally used programming language. Python is often used in as a scripting language. JavaScript embedded in a webpage can be used to control how a web browser such as Firefox displays web content, so JavaScript is a good example of a scripting language. Python can be used as a scripting language for various applications, and is ranked in the top 5-10 worldwide in terms of popularity. Python is fun to use. In fact, the origin of the name comes from the television comedy series Monty Python's Flying Circus.

Flask

Flask is a micro web framework written in Python. It is classified as a micro framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where preexisting third-party libraries provide common function

SQLite

SQL is a language to operate databases; it includes database creation, deletion, fetching rows, modifying rows, etc. SQL is an ANSI (American National Standards Institute) standard language, but there are many different versions of the SQL language.

What is SQL?

SQL is Structured Query Language, which is a computer language for storing, manipulating and retrieving data stored in a relational database.

SQL is the standard language for Relational Database System. All the Relational Database Management Systems (RDMS) like MySQL, MS Access, Oracle, Sybase, Informix, Postgres and SQL Server use SQL as their standard database language.

Also, they are using different dialects, such as −

- MS SQL Server using T-SQL,
- Oracle using PL/SQL,
- MS Access version of SQL is called JET SQL (native format) etc.

Why SQL?

SQL is widely popular because it offers the following advantages −

- Allows users to access data in the relational database management systems.
- Allows users to describe the data.
- Allows users to define the data in a database and manipulate that data.
- Allows embedding within other languages using SQL modules, libraries & pre-compilers.
- Allows users to create and drop databases and tables.
- Allows users to create view, stored procedure, functions in a database.

Data Storage and Retrieval:

SQL is used to create, read, update, and delete (CRUD) data in a database. For example, user information, product details, and content for web pages can be stored in a database and accessed using SQL queries.

User Authentication:

Websites often use SQL to manage user accounts. When a user logs in, SQL queries validate their credentials against stored data in the database.

CHAPTER 4

SOFTWARE DEVELOPMENT

4.1 MODULES DESCRIPTION

A privacy-preserving content-based retrieval project in cloud repositories aims to retrieve relevant data from encrypted databases while protecting user privacy. This is achieved through a combination of encryption, indexing, and search techniques that allow the cloud server to perform computations on encrypted data without revealing the underlying content.

4.1.1 Data Owner : The owner uploads the data to a cloud server. The file is divided into octet segments, and utilising multithreading, each segment is encoded concurrently. The encoded file is kept on a cloud server. The cover picture stores the encryption keys. The multiuser environment is known as cloud computing.

4.1.2 Data User : File request from a Cloud User. Upon requesting the file, the user also receives a key via email that includes important details. The file is decoded using a reverse technique.

4.1.3 Admin : After entering the home page, admin acts as the owner of the application and can activate and deactivate users as well as view all uploaded file details and request details. If the username and password are entered correctly, only admin can access the home page; if the details entered are incorrect, admin cannot access the home page.

4.1.4 Cloud : With all registered users' information, owners' uploaded file information, and user-downloaded information, the admin of the cloud module may manage it.

CHAPTER 5

SYSTEM DESIGN AND ANALYSIS

5.1 Architecture Diagram

An architecture diagram is a visual representation of a system's components, their interconnections, and how they work together. It helps to understand the overall structure, functionality, and data flow within a system, making it a valuable tool for communication and collaboration. Architecture diagrams can range in complexity from high-level overviews to detailed technical depictions.

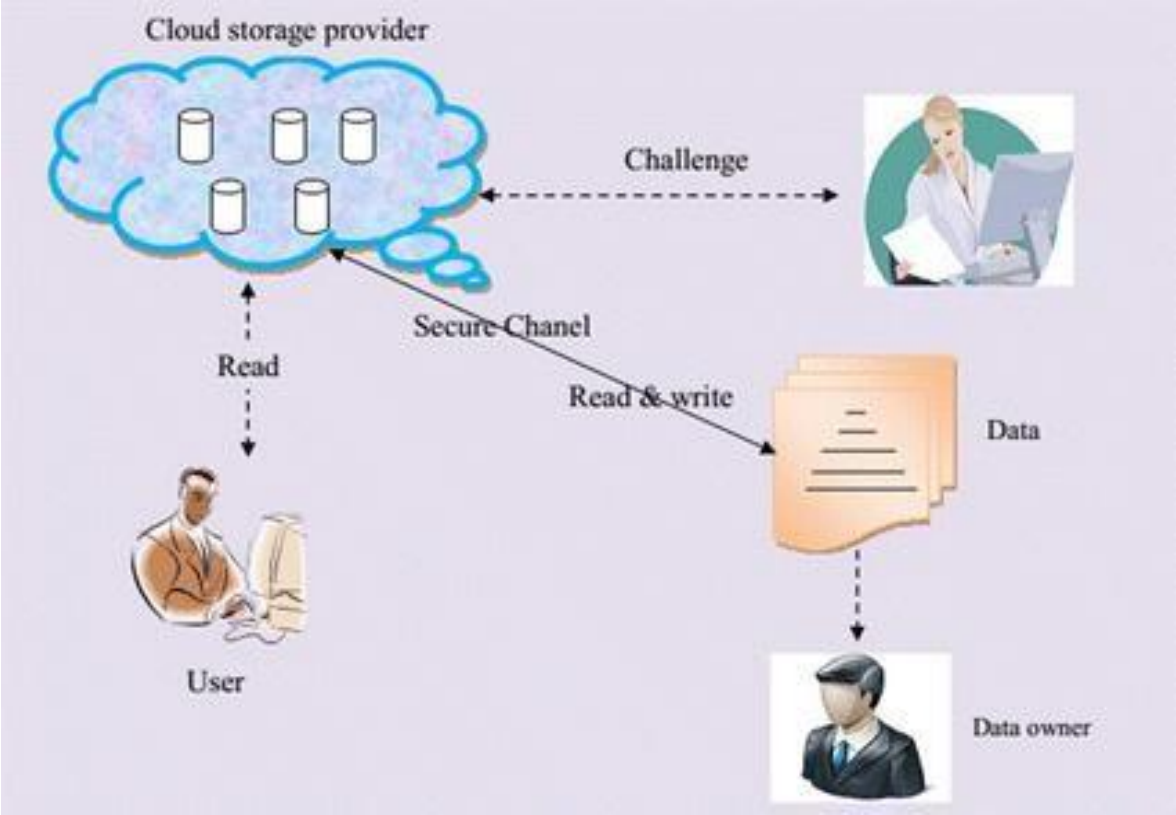


Fig5.1 Architecture Diagram

5.2 Data Flow Diagram

A Data Flow Diagram (DFD) is a graphical representation that shows how data moves through a system or process. It visually illustrates the flow of information, including inputs, outputs, storage, and processing, without detailing the timing or sequence of processes. DFDs are used to understand and document how data is handled within a system, making complex processes easier to visualize and analyze.

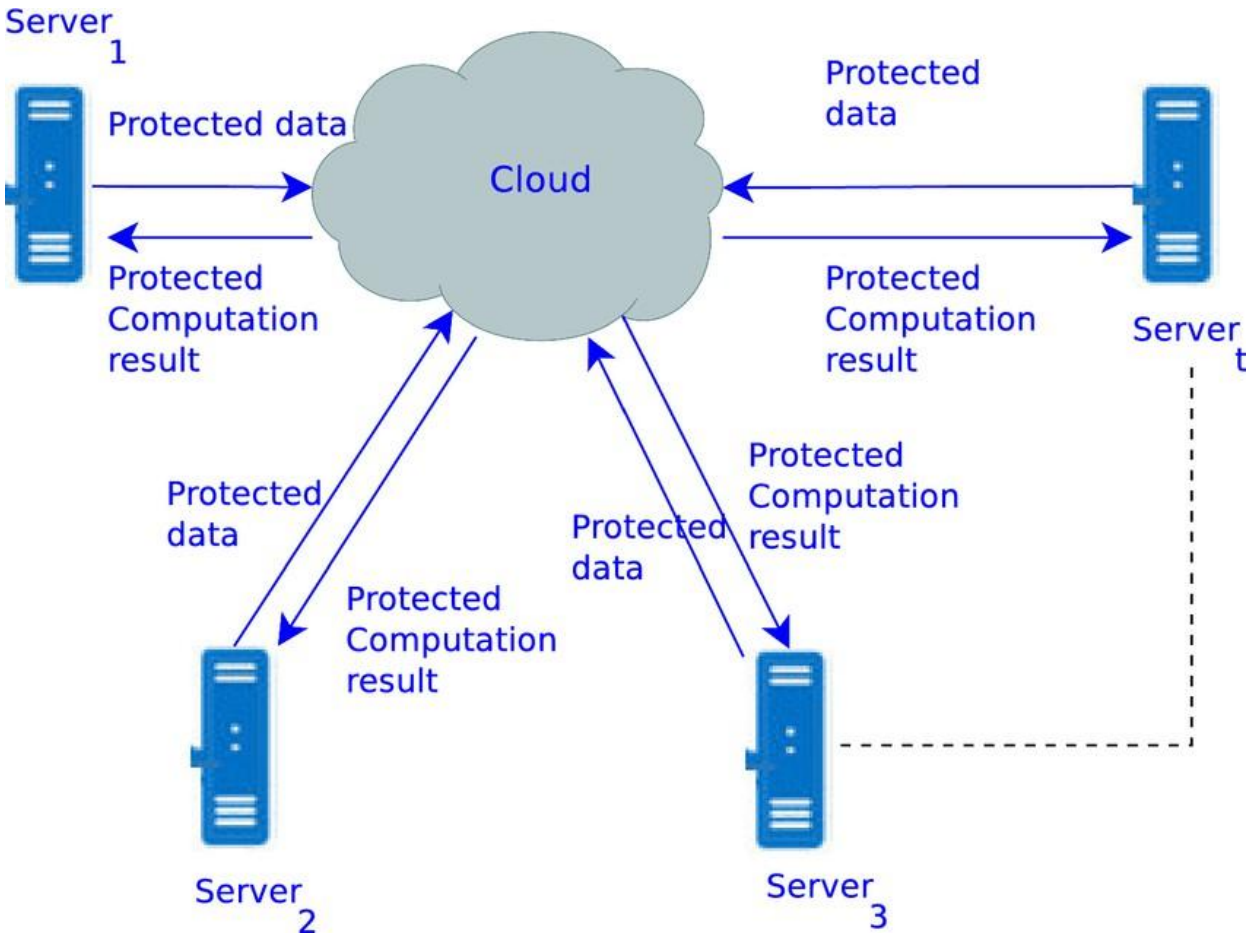


Fig 5.2 Data Flow Diagram

5.3 Use Case Diagram

A use case diagram is a visual representation that shows how users (actors) interact with a system to achieve specific goals. It's a high-level overview of a system's functionality, illustrating the different ways users can interact with it. These diagrams are a standard notation within the Unified Modeling Language (UML).

Enabling users to search for similar images without revealing sensitive information about the original images or the user's search intent to the cloud server. This is achieved through various techniques, including privacy preserving techniques Content-Based Image Retrieval (CBIR), cloud repositories integration and use cases.

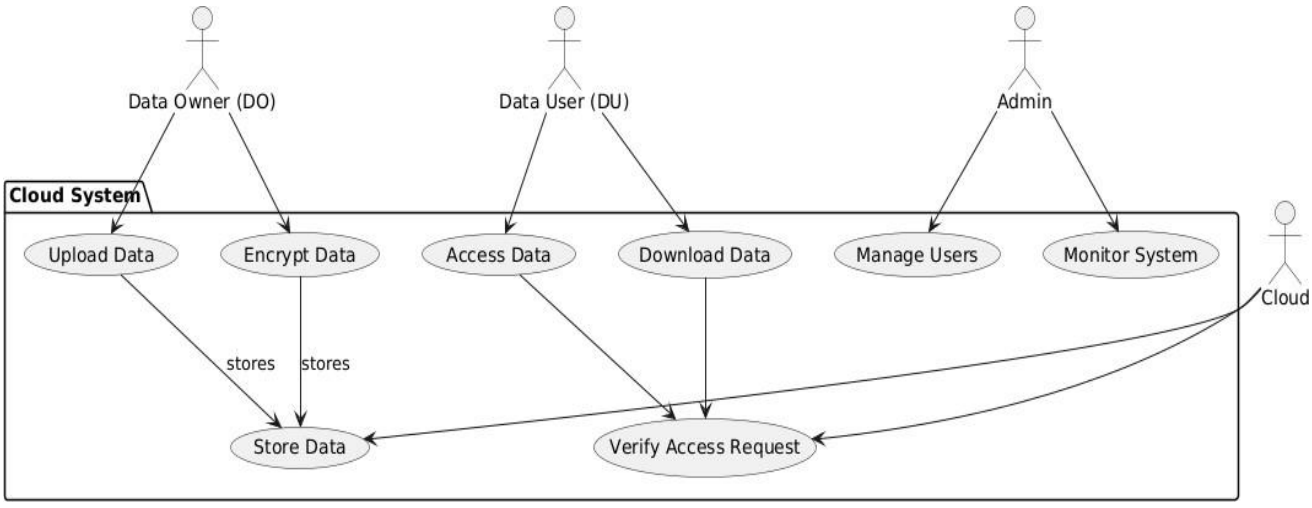


Fig 5.3 Use case Diagram

5.4 Class Diagram

A class diagram is a visual representation of the static structure of a system, particularly in object-oriented programming. It illustrates the classes within a system, their attributes, methods, and relationships with other classes. Think of it as a blueprint of your system, showing the classes and how they interact.

Enhanced privacy-preserving content-based retrieval system in cloud repositories would include classes representing users, images, features, cloud storage, and retrieval methods. Key classes would be User, Image, FeatureExtractor, CloudStorage, RetrievalEngine, and PrivacyMechanism. Relationships would model how these interact, ensuring privacy while enabling efficient retrieval.

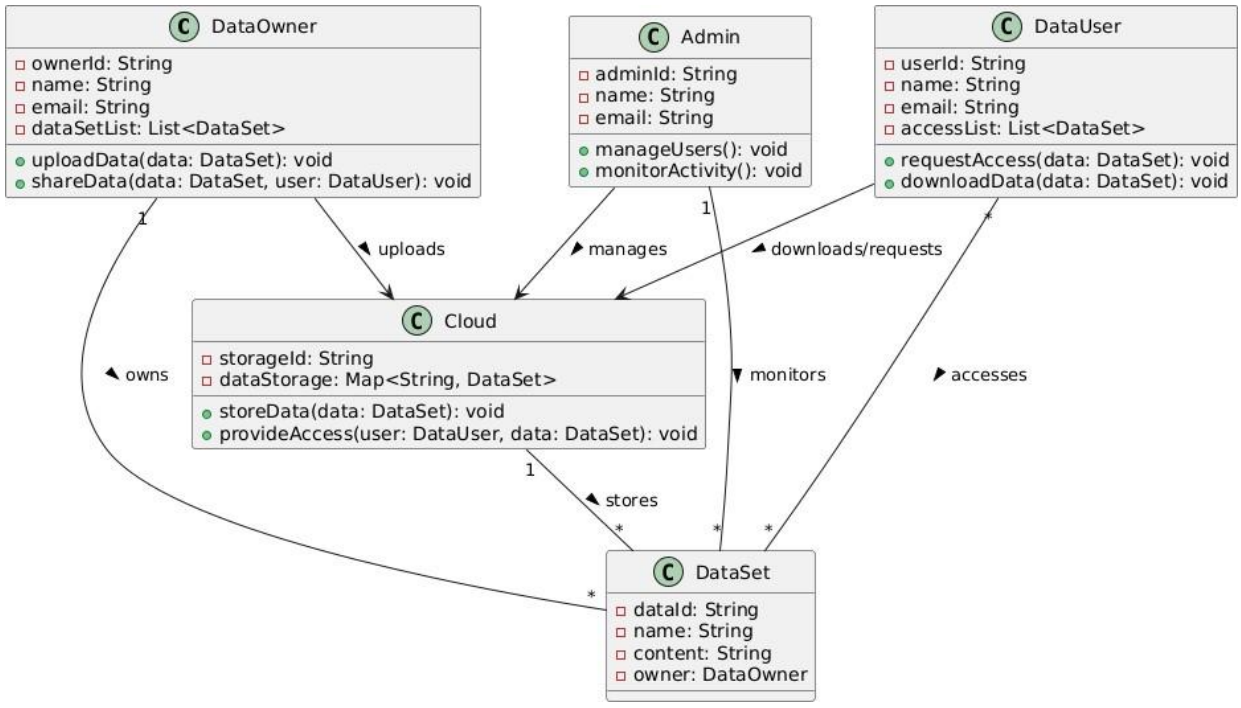


Fig 5.4 Class Diagram

5.5 Activity Diagram

An activity diagram is a type of diagram used in software development and business process modelling to visually represent the flow of actions or steps within a system or process. It's a behavioural diagram that illustrates the sequence of activities, including sequential, conditional, and concurrent flows. Essentially, it's a detailed flowchart that shows how tasks are performed and how control flows from one action to another.

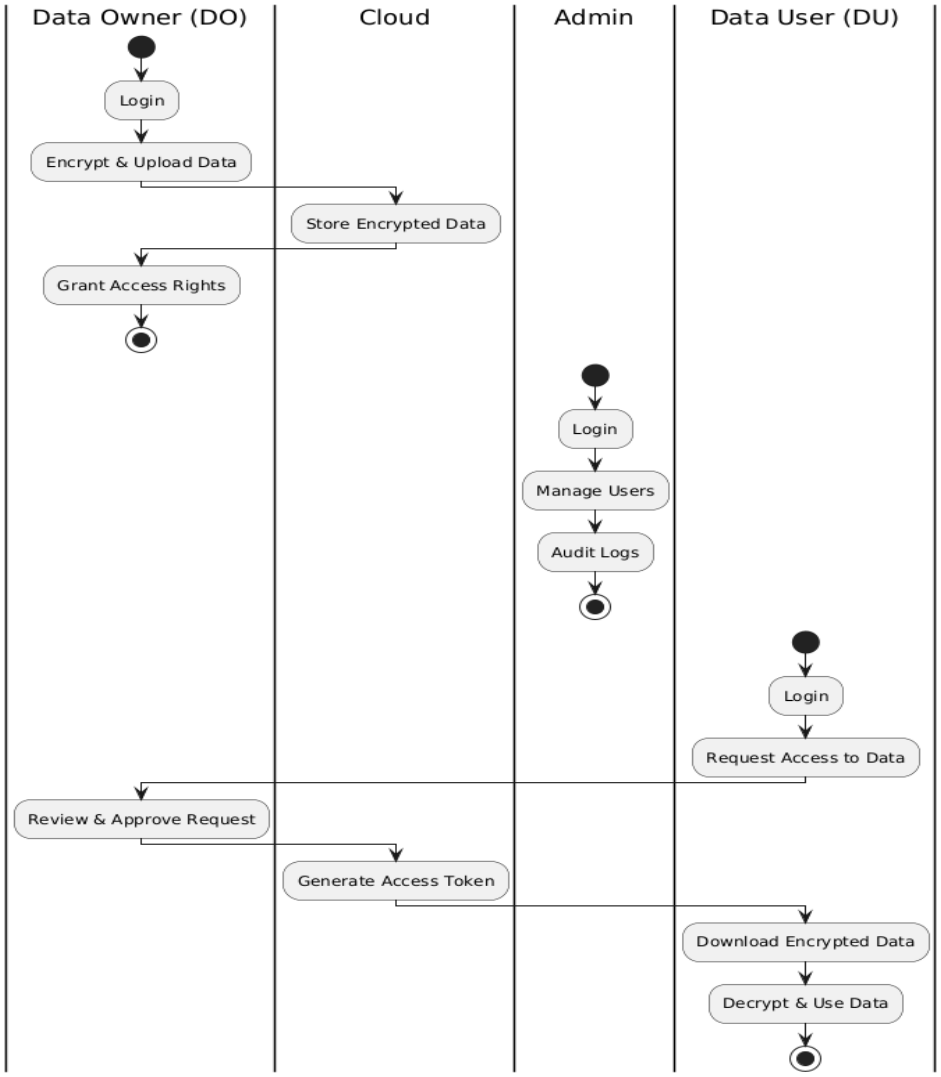


Fig 5.5 Activity Diagram

5.6 Data Base Design

1. Data Storage and Security:

Encryption:

Encrypt image features and potentially even the image data itself to protect against unauthorized access on the cloud server.

Access Control:

Implement robust access control mechanisms to ensure only authorized users can access the encrypted data or query it.

Data Residency:

Consider data residency requirements and choose cloud regions that comply with applicable regulations.

2. Feature Extraction and Indexing:

Feature Extraction:

Use deep learning models or other suitable techniques to extract robust and discriminative image features.

Indexing:

Create an efficient index (e.g., using a BKD-tree or other similarity search index) to enable fast retrieval of similar images based on the extracted features.

3. Similarity Search:

Searchable Encryption:

Consider using homomorphic encryption or other searchable encryption techniques to enable similarity searches on encrypted data without decryption.

Similarity Metrics:

Choose appropriate similarity metrics (e.g., cosine similarity) to quantify the similarity between query images and database images.

4. Privacy Preservation:

Privacy-Preserving Techniques: Employ techniques like differential privacy, data perturbation, or other privacy-preserving methods to minimize the risk of revealing sensitive information during the retrieval process.

Anonymization:

Consider anonymizing user queries and results to further enhance privacy.

5. Database Schema:

Image Metadata:

Store metadata about each image, such as file name, user ID, and encryption keys.

Feature Vectors:

Store the extracted image features in a separate table or as part of the image record.

User Access Control:

Maintain user access control information, including permissions to query and access specific images.

CHAPTER 6

SYSTEM TESTING AND IMPLEMENTATION

6.1 SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types oftest. Each test type addresses a specific testing requirement.

TYPES OF TESTS:

Testing is the process of trying to discover every conceivable fault or weakness in a work product. The different type of testing is given below:

6.1.1 UNIT TESTING:

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration.

This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration.

Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

6.1.2 INTEGRATION TESTING:

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

6.1.3 FUNCTIONAL TEST:

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input :Identified classes of valid input must be accepted.

Invalid Input :Identified classes of invalid input must be rejected.

Functions :Identified functions must be exercised.

Output :Identified classes of application outputs must be exercised.

Systems/ Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

6.1.4 PERFORMANCE TESTING:

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configurationoriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

6.1.5 WHITE BOX TESTING:

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

6.1.6 BLACK BOX TESTING:

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

6.1.7 ACCEPTANCE TESTING:

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered

6.2 INPUT DESIGN AND OUTPUT DESIGN

INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:'

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

OBJECTIVES

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily.

When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2. Select methods for presenting information.

3. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

- ❖ Convey information about past activities, current status or projections of the
- ❖ Future.
- ❖ Signal important events, opportunities, problems, or warnings.
- ❖ Trigger an action. Confirm an action.

Accuracy:

The retrieval system should provide accurate results, even when dealing with encrypted or hashed data.

Efficiency:

The system should be efficient in terms of storage, computation, and retrieval speed.

Security:

The system should be secure against various attacks, including data breaches and unauthorized access.

Scalability:

The system should be able to handle large datasets and user queries.

CHAPTER 7

APPENDIX

7.1 Source Code

```
import os
import base64
import sqlite3
import uuid
from flask import Flask, render_template, request, redirect, url_for, flash, session, send_file
from werkzeug.security import generate_password_hash, check_password_hash
from io import BytesIO
import mimetypes
import io
import base64

app = Flask(__name__)
app.secret_key = 'supersecretkey'

DB_NAME = 'database.db'

def guess_mime_type(filename):
mime_type, _ = mimetypes.guess_type(filename)
return mime_type or 'application/octet-stream'

# --- Database Setup ---
def init_db():
conn = sqlite3.connect(DB_NAME)
c = conn.cursor()
# Users table
c.execute('''
CREATE TABLE IF NOT EXISTS users (
id INTEGER PRIMARY KEY AUTOINCREMENT,
username TEXT UNIQUE NOT NULL,
password TEXT NOT NULL
)
''')
# viewer History
```

```
c.execute('''
CREATE TABLE IF NOT EXISTS views (
id INTEGER PRIMARY KEY AUTOINCREMENT,
type TEXT,
user_id INTEGER,
file_id TEXT,
created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
)
''')
# Files table
c.execute('''
CREATE TABLE IF NOT EXISTS files (
id INTEGER PRIMARY KEY AUTOINCREMENT,
user_id INTEGER,
file_id TEXT,
filename TEXT,
pin TEXT,
segment_number INTEGER,
data TEXT,
FOREIGN KEY(user_id) REFERENCES users(id)
)
''')
conn.commit()
conn.close()

init_db()

# --- Helper Functions ---
def get_user(username):
conn = sqlite3.connect(DB_NAME)
c = conn.cursor()
c.execute('SELECT * FROM users WHERE username = ?', (username,))
user = c.fetchone()
conn.close()
return user

def save_file_segments(user_id, filename, file_data, pin):
file_id = str(uuid.uuid4())
segment_size = 512  # bytes
total_segments = (len(file_data) + segment_size - 1) // segment_size
```

```
conn = sqlite3.connect(DB_NAME)
c = conn.cursor()
for i in range(total_segments):
segment = file_data[i*segment_size:(i+1)*segment_size]
encoded_segment = base64.b64encode(segment).decode('utf-8')
c.execute('''
INSERT INTO files (user_id, file_id, filename, pin, segment_number, data)
VALUES (?, ?, ?, ?, ?, ?)
''', (user_id, file_id, filename, pin, i, encoded_segment))
conn.commit()
conn.close()

return file_id

def get_file_segments(file_id):
conn = sqlite3.connect(DB_NAME)
c = conn.cursor()
c.execute('''
SELECT segment_number, data FROM files
WHERE file_id = ?
ORDER BY segment_number ASC
''', (file_id,))
segments = c.fetchall()
conn.close()
return segments

# --- Routes ---

@app.route('/')
def index():
if 'user_id' in session:
return redirect(url_for('upload'))
return redirect(url_for('login'))

@app.route('/register', methods=['GET', 'POST'])
def register():
if request.method == 'POST':
username = request.form['username']
password = generate_password_hash(request.form['password'])
```

```
conn = sqlite3.connect(DB_NAME)
c = conn.cursor()
try:
c.execute('INSERT INTO users (username, password) VALUES (?, ?)', (username,
password))
conn.commit()
flash('Registration successful. Please login.', 'success')
return redirect(url_for('login'))
except sqlite3.IntegrityError:
flash('Username already exists!', 'danger')
finally:
conn.close()
return render_template('register.html')

@app.route('/login', methods=['GET', 'POST'])
def login():
if request.method == 'POST':
username = request.form['username']
password = request.form['password']
user = get_user(username)

if user and check_password_hash(user[2], password):
session['user_id'] = user[0]
session['username'] = user[1]
return redirect(url_for('my_files'))
else:
flash('Invalid username or password', 'danger')
return render_template('login.html')

@app.route('/logout')
def logout():
session.clear()
return redirect(url_for('login'))

@app.route('/upload', methods=['GET', 'POST'])
def upload():
if 'user_id' not in session:
return redirect(url_for('login'))
```

```
if request.method == 'POST':
file = request.files['file']
pin = request.form['pin']

if file and pin:
file_data = file.read()
file_id = save_file_segments(session['user_id'], file.filename, file_data, pin)
flash(f'File uploaded successfully! Access it here: {request.url_root}file/{file_id}', 'success')
return redirect(url_for('upload'))
else:
flash('File and PIN required.', 'danger')

return render_template('upload.html')

@app.route('/delete-file/<file_id>', methods=['POST'])
def delete_file(file_id):
if 'user_id' not in session:
return redirect(url_for('login'))

conn = sqlite3.connect(DB_NAME)
c = conn.cursor()

# Ensure the file belongs to the logged-in user
c.execute('SELECT 1 FROM files WHERE file_id = ? AND user_id = ?', (file_id,
session['user_id']))
if not c.fetchone():
conn.close()
flash('Unauthorized action.', 'danger')
return redirect(url_for('my_files'))

# Delete all segments of the file
c.execute('DELETE FROM files WHERE file_id = ? AND user_id = ?', (file_id,
session['user_id']))
conn.commit()
conn.close()

flash('File deleted successfully.', 'success')
return redirect(url_for('my_files'))
```

```
@app.route('/my-files')
def my_files():
if 'user_id' not in session:
return redirect(url_for('login'))

conn = sqlite3.connect(DB_NAME)
c = conn.cursor()

# Group by file_id to get one file per uploaded file
c.execute('''
SELECT file_id, filename, MIN(id) as first_id
FROM files
WHERE user_id = ?
GROUP BY file_id
ORDER BY first_id DESC
''', (session['user_id'],))

files = c.fetchall()
conn.close()

return render_template('file_list.html', files=files)

@app.route('/file-history/<file_id>')
def file_history(file_id):
if 'user_id' not in session:
return redirect(url_for('login'))

conn = sqlite3.connect(DB_NAME)
c = conn.cursor()

c.execute('SELECT filename FROM files WHERE file_id = ? LIMIT 1', (file_id,))
file_row = c.fetchone()
file_name = file_row[0] if file_row else 'Unknown File'

# Join views with users table to get username
c.execute('''
SELECT users.username, views.type, views.created_at
FROM views
JOIN users ON users.id = views.user_id
WHERE views.file_id = ?
```

```
ORDER BY views.created_at DESC
'''', (file_id,))

history = c.fetchall()
conn.close()

return render_template('file_history.html', history=history,file_name=file_name)

@app.route('/view_file/<file_id>', methods=['GET', 'POST'])
def view_file(file_id):
if 'user_id' not in session:
return redirect(url_for('login'))

conn = sqlite3.connect(DB_NAME)
c = conn.cursor()

# Check file exists
c.execute('SELECT filename FROM files WHERE file_id = ? LIMIT 1', (file_id,))
file_row = c.fetchone()

if not file_row:
conn.close()
flash('File not found.', 'danger')
return redirect(url_for('my_files'))

filename = file_row[0]
file_data_url = None

if request.method == 'POST':
pin = request.form.get('pin')

c.execute('INSERT INTO views (type,user_id, file_id) VALUES (?, ?,?)', (''File
View'',session['user_id'],file_id ))
conn.commit()

# VERY IMPORTANT - Order by segment_number
c.execute('''
SELECT data
FROM files
WHERE file_id = ? AND pin = ?
```

```
ORDER BY segment_number ASC
'', (file_id, pin))

segments = c.fetchall()
conn.close()

if not segments:
flash('Incorrect PIN or no data found.', 'danger')
return redirect(request.url)

# Build full base64 correctly
full_base64 = ''.join([segment[0] for segment in segments])

mime_type = guess_mime_type(filename)

# Create valid Data URL
file_data_url = f"data:{mime_type};base64,{full_base64}"

else:
conn.close()

return render_template('file_view_pin.html', file_id=file_id, filename=filename,
file_data_url=file_data_url)


@app.route('/file/<file_id>', methods=['GET', 'POST'])
def file(file_id):
if 'user_id' not in session:
return redirect(url_for('login'))

if request.method == 'POST':
pin = request.form['pin']

conn = sqlite3.connect(DB_NAME)
c = conn.cursor()
c.execute('INSERT INTO views (type,user_id, file_id) VALUES (?, ?,?)', ("Packets
View",session['user_id'],file_id ))
conn.commit()

c.execute('SELECT DISTINCT pin FROM files WHERE file_id = ?', (file_id,))
```

```
result = c.fetchone()
conn.close()

if result and result[0] == pin:
segments = get_file_segments(file_id)
return render_template('view_packets.html', segments=segments, file_id=file_id)
else:
flash('Invalid PIN!', 'danger')
return render_template('enter_pin.html', file_id=file_id)

@app.route('/file/<file_id>/download')
def download(file_id):
if 'user_id' not in session:
return redirect(url_for('login'))

segments = get_file_segments(file_id)
file_data = b''.join([base64.b64decode(seg[1]) for seg in segments])

# Get filename
conn = sqlite3.connect(DB_NAME)
c = conn.cursor()

c.execute('INSERT INTO views (type,user_id, file_id) VALUES (?, ?,?)',
("download",session['user_id'],file_id ))
conn.commit()

c.execute('SELECT DISTINCT filename FROM files WHERE file_id = ?', (file_id,))
filename = c.fetchone()[0]
conn.close()

return send_file(BytesIO(file_data), download_name=filename, as_attachment=True)

if __name__ == '__main__':
app.run(debug=True)



{% extends 'layout/Base.html' %}

{% block title %}Enter Pin{% endblock %}
```

```
{% block content %}
<div class="card" style="width: 100%; max-width: 400px;">
<h2 class="text-center mb-4">Enter PIN</h2>
<form method="POST">
<input type="text" name="pin" class="form-control mb-3" placeholder="PIN" required>
<button type="submit" class="btn btn-primary w-100">View Packets</button>
</form>
{% with messages = get_flashed_messages(with_categories=true) %}
{% if messages %}
{% for category, message in messages %}
<div class="alert alert-{{ category }}">{{ message }}</div>
{% endfor %}
{% endif %}
{% endwith %}
</div>
{% endblock %}

{% extends 'layout/Base.html' %}

{% block content %}
<div class="container mt-5">
<h2 class="mb-4">File History</h2>

<div class="card shadow-sm">
<div class="card-body">
<h5 class="card-title mb-4">File: <span class="text-primary">{{ file_name
}}</span></h5>

{% if history %}
<table class="table table-hover">
<thead>
<tr>
<th>User</th>
<th>Action</th>
<th>Timestamp</th>
</tr>
</thead>
<tbody>
{% for record in history %}
```

```
<tr>
<td>{{ record[0] }}</td>
<td>{{ record[1] }}</td>
<td>{{ record[2] }}</td>
</tr>
{% endfor %}
</tbody>
</table>
{% else %}
<p>No views yet for this file.</p>
{% endif %}
</div>
</div>

<div class="mt-3">
<a href="{{ url_for('my_files') }}" class="btn btn-secondary">Back to My Files</a>
</div>
</div>
{% endblock %}


{% extends 'layout/Base.html' %}

{% block title %}My Uploaded Files{% endblock %}

{% block content %}
<div class="container p-4">
<h2 class="text-center mb-4">My Files</h2>

{% if files %}
<div class="row">
{% for file in files %}
<div class="col-md-4 mb-2">
<div class="card shadow-sm h-100">
<div class="card-body flex-column">
<h5 class="card-title">{{ file[1] }}</h5>
<hr>
<div class="row gap-0">
<div class="col-3">
<div class="mt-auto">
```

```
<a href="/file-history/{{ file[0] }}" class="btn btn-secondary text-center w-100"><i
class="bi bi-clock-history"></i></a>
</div>
</div>
<div class="col-9">
<div class="mt-auto">
<a href="/file/{{ file[0] }}" class="btn btn-primary w-100"><i class="bi bi-eye"></i> View
Packets</a>
</div>
</div>
</div>
<hr>
<a href="/view_file/{{ file[0] }}" class="btn btn-primary w-100"><i class="bi bi-eye"></i>
View File</a>
<form action="{{ url_for('delete_file', file_id=file[0]) }}" method="post" class="mt-2">
<button type="submit" class="btn btn-danger btn-sm w-100 py-2" onclick="return
confirm('Are you sure you want to delete this file?')"><i class="bi bi-trash"></i>
Delete</button>
</form>
</div>
</div>
</div>
{% endfor %}
</div>
{% else %}
<div class="alert alert-info text-center">
You have not uploaded any files yet.
</div>
{% endif %}
</div>
{% endblock %}


{% extends 'layout/Base.html' %}

{% block content %}
<div class="container mt-5">
<div class="row justify-content-center">
<div class="col-md-8">
<div class="card shadow-sm">
```

```
<div class="card-body">

<h3 class="card-title mb-4 text-center">View File</h3>

{% if not file_data_url %}
<!-- PIN Form -->
<form method="post">
<div class="mb-3">
<label class="form-label">Filename:</label>
<input type="text" class="form-control" value="{{ filename }}" readonly>
</div>

<div class="mb-3">
<label for="pin" class="form-label">PIN</label>
<input type="password" name="pin" id="pin" class="form-control" required>
</div>

<button type="submit" class="btn btn-primary w-100">View File</button>
</form>
{% else %}
<!-- File Preview -->
<h5 class="mb-3">Filename: <span class="text-primary">{{ filename }}</span></h5>

{% if file_data_url.startswith('data:application/pdf') %}
<!-- PDF Viewer -->
<iframe src="{{ file_data_url }}" width="100%" height="600px" style="border:
none;"></iframe>
{% elif file_data_url.startswith('data:image') %}
<!-- Image Viewer -->
<img src="{{ file_data_url }}" alt="Image" class="img-fluid rounded shadow">
{% elif file_data_url.startswith('data:video') %}
<!-- Video Viewer -->
<video controls class="w-100 rounded shadow">
<source src="{{ file_data_url }}">
Your browser does not support the video tag.
</video>
{% endif %}
<div class="alert alert-info">Cannot preview this file type. You can download it
instead.</div>
<a href="{{ url_for('download', file_id=file_id) }}" class="btn btn-primary">Download
```

```
File</a>

<div class="mt-4 text-center">
<a href="{{ url_for('my_files') }}" class="btn btn-secondary">Back to My Files</a>
</div>
{% endif %}

</div>
</div>
</div>
</div>
</div>
{% endblock %}


<!DOCTYPE html>
{% extends 'layout/Base.html' %}

{% block title %}Login{% endblock %}

{% block content %}
<div class="card" style="width: 100%; max-width: 400px;">
<h2 class="text-center mb-4">Login</h2>
<form method="POST">
<input name="username" class="form-control mb-3" placeholder="Username" required>
<input name="password" type="password" class="form-control mb-3"
placeholder="Password" required>
<button type="submit" class="btn btn-primary w-100">Login</button>
</form>
<p class="mt-3 text-center">
Don't have an account? <a href="{{ url_for('register') }}">Register</a>
</p>
{% with messages = get_flashed_messages(with_categories=true) %}
{% if messages %}
{% for category, message in messages %}
<div class="alert alert-{{ category }}">{{ message }}</div>
{% endfor %}
{% endif %}
{% endwith %}
</div>
```

```
{% endblock %}

{% extends 'layout/Base.html' %}

{% block title %}Register{% endblock %}

{% block content %}
<div class="card" style="width: 100%; max-width: 400px;">
<h2 class="text-center mb-4">Register</h2>
<form method="POST">
<input name="username" class="form-control mb-3" placeholder="Username" required>
<input name="password" type="password" class="form-control mb-3"
placeholder="Password" required>
<button type="submit" class="btn btn-primary w-100">Register</button>
</form>
<p class="mt-3 text-center">
Already have an account? <a href="{{ url_for('login') }}">Login</a>
</p>
{% with messages = get_flashed_messages(with_categories=true) %}
{% if messages %}
{% for category, message in messages %}
<div class="alert alert-{{ category }}">{{ message }}</div>
{% endfor %}
{% endif %}
{% endwith %}
</div>
{% endblock %}

{% extends 'layout/Base.html' %}

{% block title %}Upload File{% endblock %}

{% block content %}
<div class="card" style="width: 100%; max-width: 600px;">
<h2 class="text-center mb-4">Upload File</h2>
<form id="uploadForm" method="POST" enctype="multipart/form-data">
<input type="file" name="file" id="fileInput" class="form-control mb-3" required>
<input type="text" name="pin" class="form-control mb-3" placeholder="Enter a PIN"
required>
<button type="submit" class="btn btn-primary w-100 mb-3">Upload</button>
```

```
<div class="progress mb-3" style="display: none;">
<div id="progressBar" class="progress-bar" role="progressbar" style="width:
0%">0%</div>
</div>
</form>

<div id="packetPreview" style="display: none;">
<h5 class="text-center mt-4">Packet Preview:</h5>
<ul class="list-group" id="packetList"></ul>
</div>

{% with messages = get_flashed_messages(with_categories=true) %}
{% if messages %}
{% for category, message in messages %}
<div class="alert alert-{{ category }}">{{ message }}</div>
{% endfor %}
{% endif %}
{% endwith %}
</div>
{% endblock %}


{% extends 'layout/Base.html' %}

{% block title %}View_Packets{% endblock %}

{% block content %}
<div class="card" style="width: 100%; max-width: 700px; overflow-x: auto;">
<div class="text-center mt-4">
<a href="{{ url_for('download', file_id=file_id) }}" class="btn btn-primary">Download
File</a>
</div>
<hr>
<h2 class="text-center mb-4">Packets</h2>
<ul class="list-group">
{% for segment in segments %}
<li class="list-group-item">
<strong>Packet {{ segment[0] }}</strong>: {{ segment[1] }}
</li>
{% endfor %}
```

```
</ul>
</div>
{% endblock %}


<!DOCTYPE html>
<html lang="en">
<head>
<title>{% block title %}Flask App{% endblock %}</title>
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
<link href="{{ url_for('static', filename='css/style.css') }}" rel="stylesheet">
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-
icons@1.11.3/font/bootstrap-icons.min.css">
</head>
<body>
{% include 'layout/Header.html' %}
<main class="container my-5 d-flex flex-column align-items-center justify-content-center">
{% block content %}{% endblock %}
</main>
{% include 'layout/Footer.html' %}
<script src="{{ url_for('static', filename='js/upload_preview.js') }}"></script>
</body>
</html>


<nav class="navbar navbar-expand-lg navbar-light bg-white shadow-sm" id="mainNavbar">
<div class="container">
<a class="navbar-brand fw-bold" href="{{ url_for('my_files') }}">Secure File Storage</a>

<button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarContent">
<span class="navbar-toggler-icon"></span>
</button>

<div class="collapse navbar-collapse justify-content-end" id="navbarContent">
<ul class="navbar-nav ms-auto mb-2 mb-lg-0">
{% if session.get('user_id') %}
<li class="nav-item">
<a class="nav-link {% if request.endpoint == 'my_files' %}active-link{% endif %}"
```

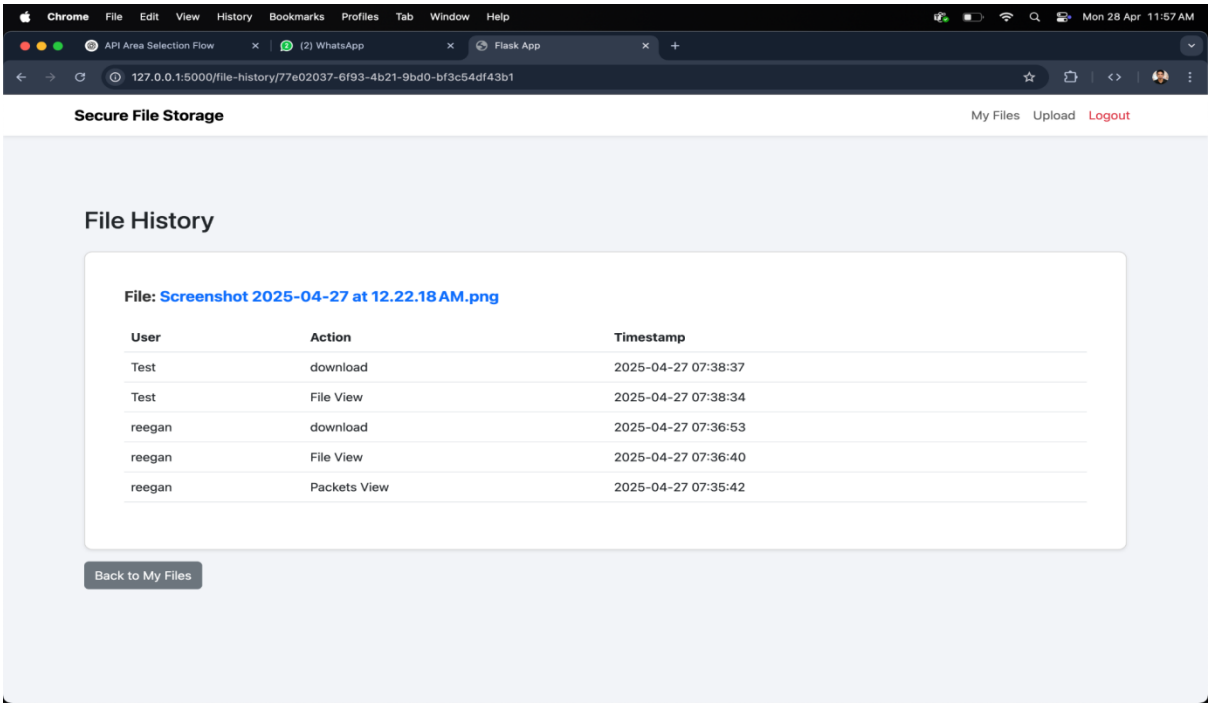
```
href="{{ url_for('my_files') }}">My Files</a>
</li>

<li class="nav-item">
<a class="nav-link {% if request.endpoint == 'upload' %}active-link{% endif %}" href="{{
url_for('upload') }}">Upload</a>
</li>

<li class="nav-item">
<a class="nav-link text-danger" href="{{ url_for('logout') }}">Logout</a>
</li>
{% else %}
<li class="nav-item">
<a class="nav-link {% if request.endpoint == 'login' %}active-link{% endif %}" href="{{
url_for('login') }}">Login</a>
</li>

<li class="nav-item">
<a class="nav-link {% if request.endpoint == 'register' %}active-link{% endif %}" href="{{
url_for('register') }}">Register</a>
</li>
{% endif %}

</ul>
</div>
</div>
</nav>


body {
background: #f2f4f8;
min-height: 100vh;
display: flex;
flex-direction: column;
}

.active-link {
font-weight: bold;
color: #0d6efd !important; /* Bootstrap Primary Blue */
border-bottom: 2px solid #0d6efd;
```

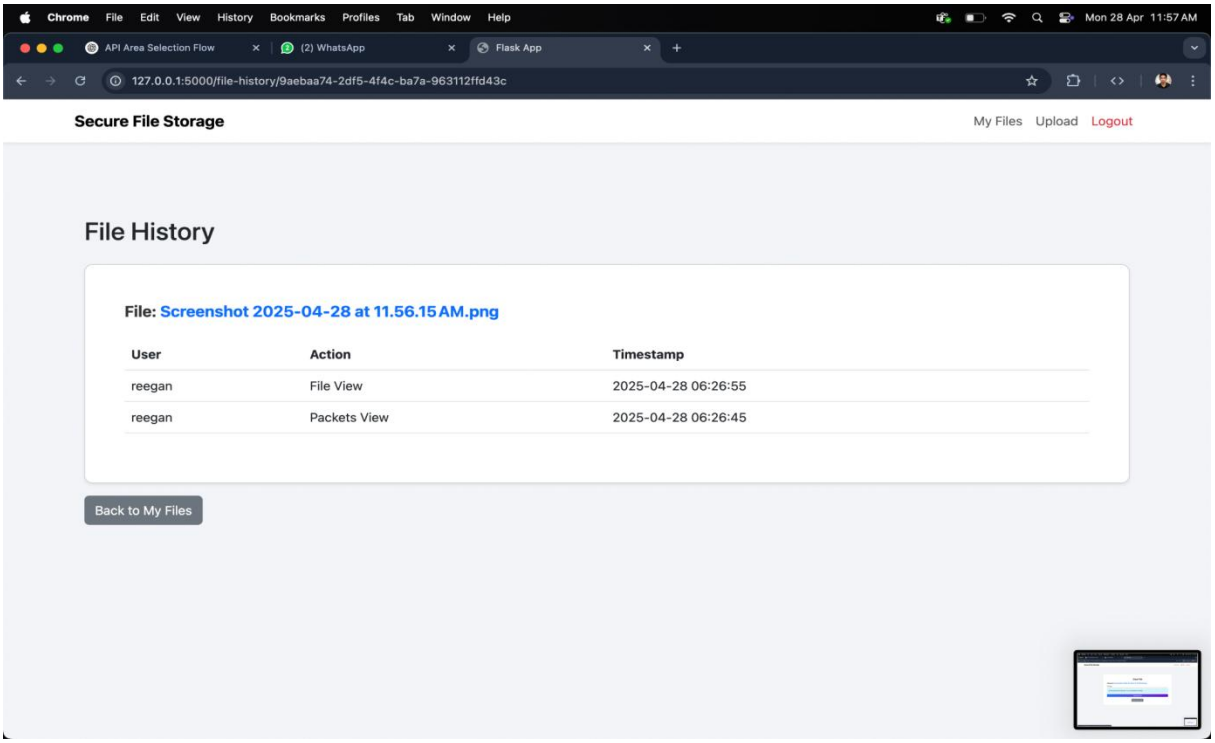
```
}


.card {
background: #fff;
border-radius: 10px;
padding: 2rem;
box-shadow: 0 0 15px rgba(0, 0, 0, 0.05);
transition: all 0.3s ease;
}
.card-title {
font-weight: bold;
color: #333;
}

.card-body {
display: flex;
flex-direction: column;
}

.card:hover {
transform: translateY(-3px);
transition: 0.3s ease;
}


.card:hover {
box-shadow: 0 0 25px rgba(0, 0, 0, 0.1);
}

.btn-primary {
background: linear-gradient(135deg, #6a11cb, #2575fc);
border: none;
}

.btn-primary:hover {
background: linear-gradient(135deg, #2575fc, #6a11cb);
}

.progress {
```

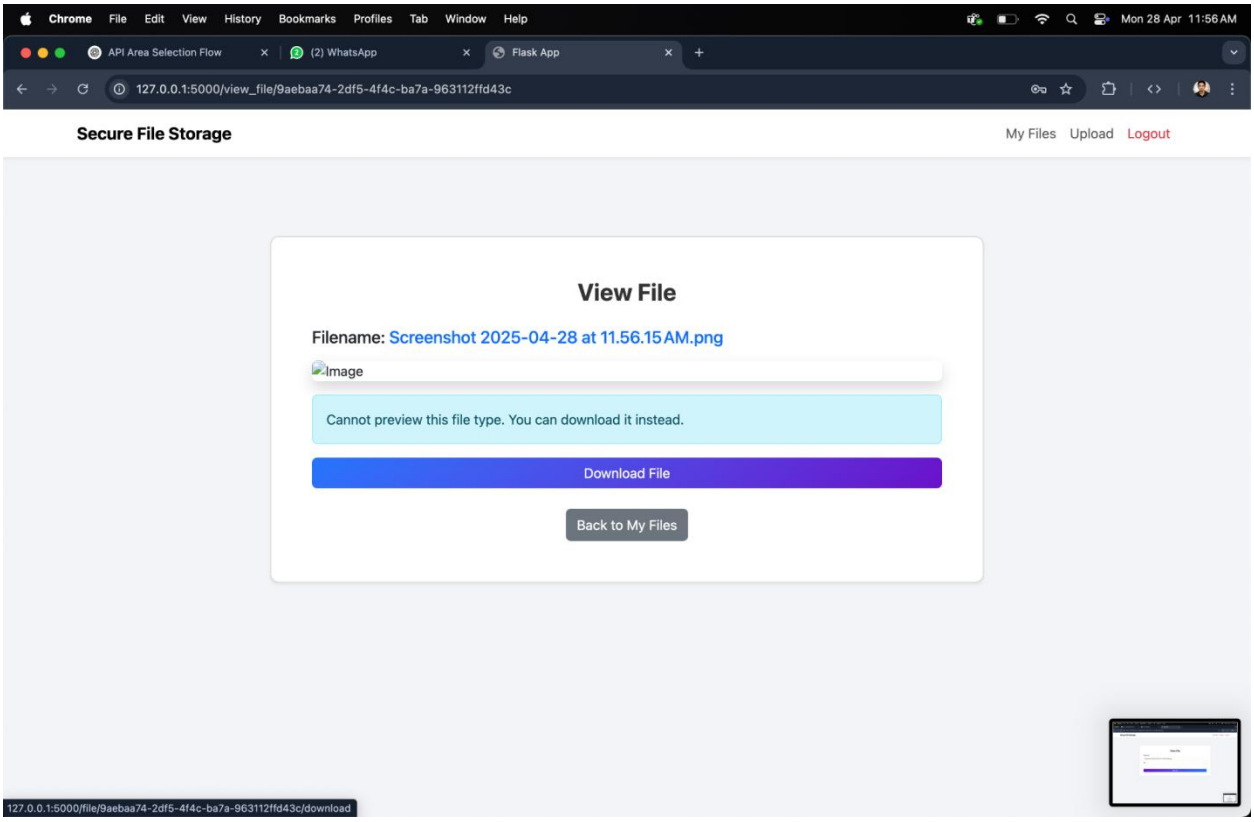
```
height: 20px;
}

.alert {
margin-top: 1rem;
}


document.addEventListener('DOMContentLoaded', function() {


const fileInput = document.getElementById('fileInput');
const packetList = document.getElementById('packetList');
const packetPreview = document.getElementById('packetPreview');
const progressBar = document.getElementById('progressBar');
const progressDiv = document.querySelector('.progress');

fileInput.addEventListener('change', function() {
const file = fileInput.files[0];
if (!file) return;

const reader = new FileReader();
reader.onload = function(e) {
const data = e.target.result;
const packetSize = 1024; // 1 KB per packet
packetList.innerHTML = '';
packetPreview.style.display = 'block';

for (let i = 0; i < data.length; i += packetSize) {
const packet = data.slice(i, i + packetSize);
const item = document.createElement('li');
item.className = 'list-group-item';
item.textContent = `Packet ${i / packetSize + 1}: ${btoa(packet).substring(0, 50)}...`;
packetList.appendChild(item);
}
};
reader.readAsBinaryString(file);
});

const uploadForm = document.getElementById('uploadForm');
```

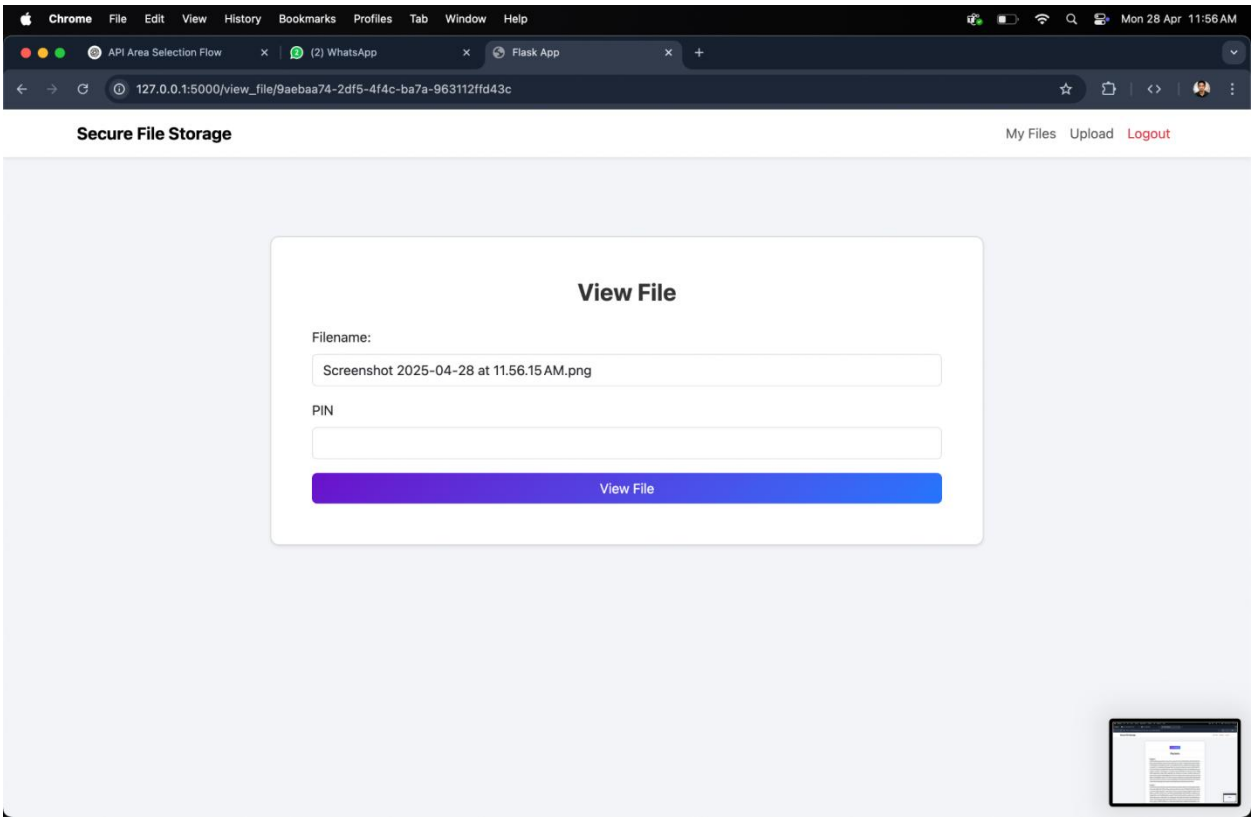
```
uploadForm.addEventListener('submit', function(e) {
progressDiv.style.display = 'block';
const fakeUpload = setInterval(() => {
let current = parseInt(progressBar.style.width) || 0;
if (current >= 100) {
clearInterval(fakeUpload);
} else {
current += 10;
progressBar.style.width = current + '%';
progressBar.textContent = current + '%';
}
}, 100);
});
});
```

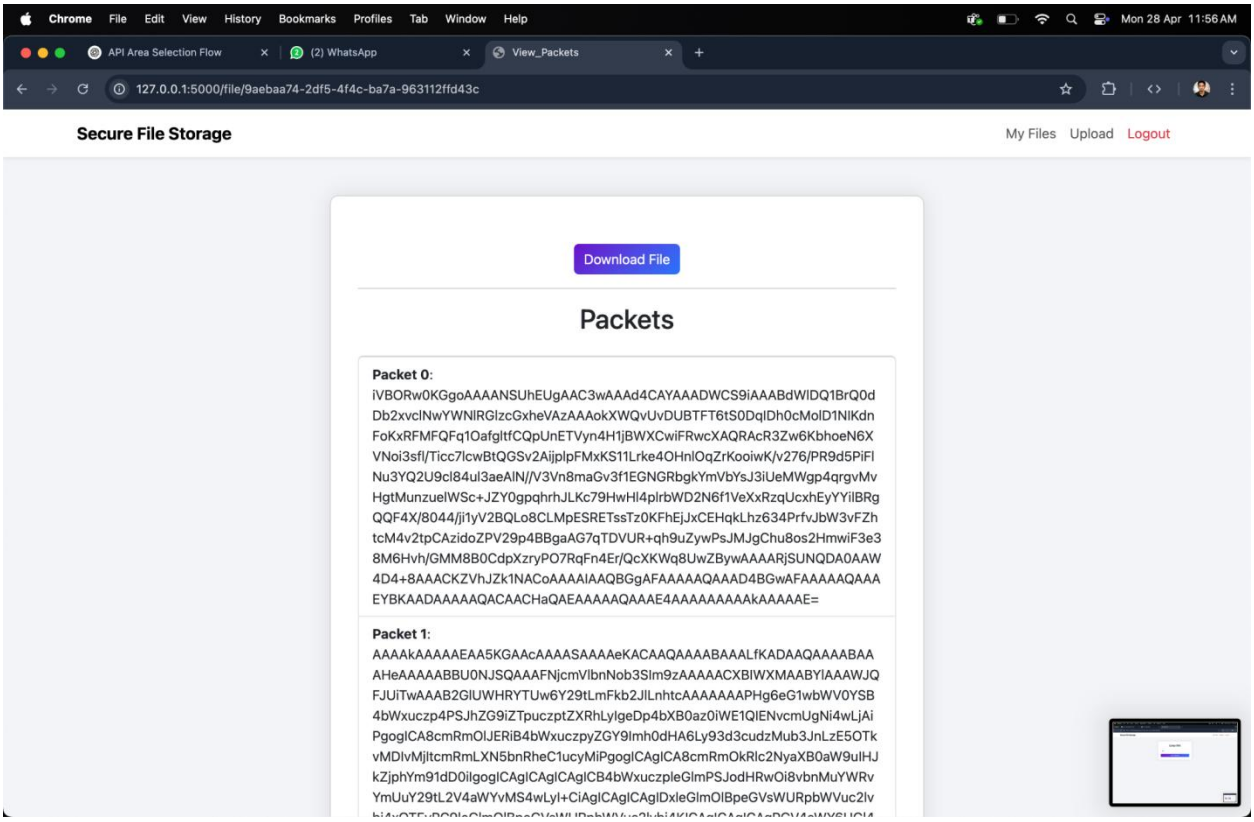
7.2 Screen Shots

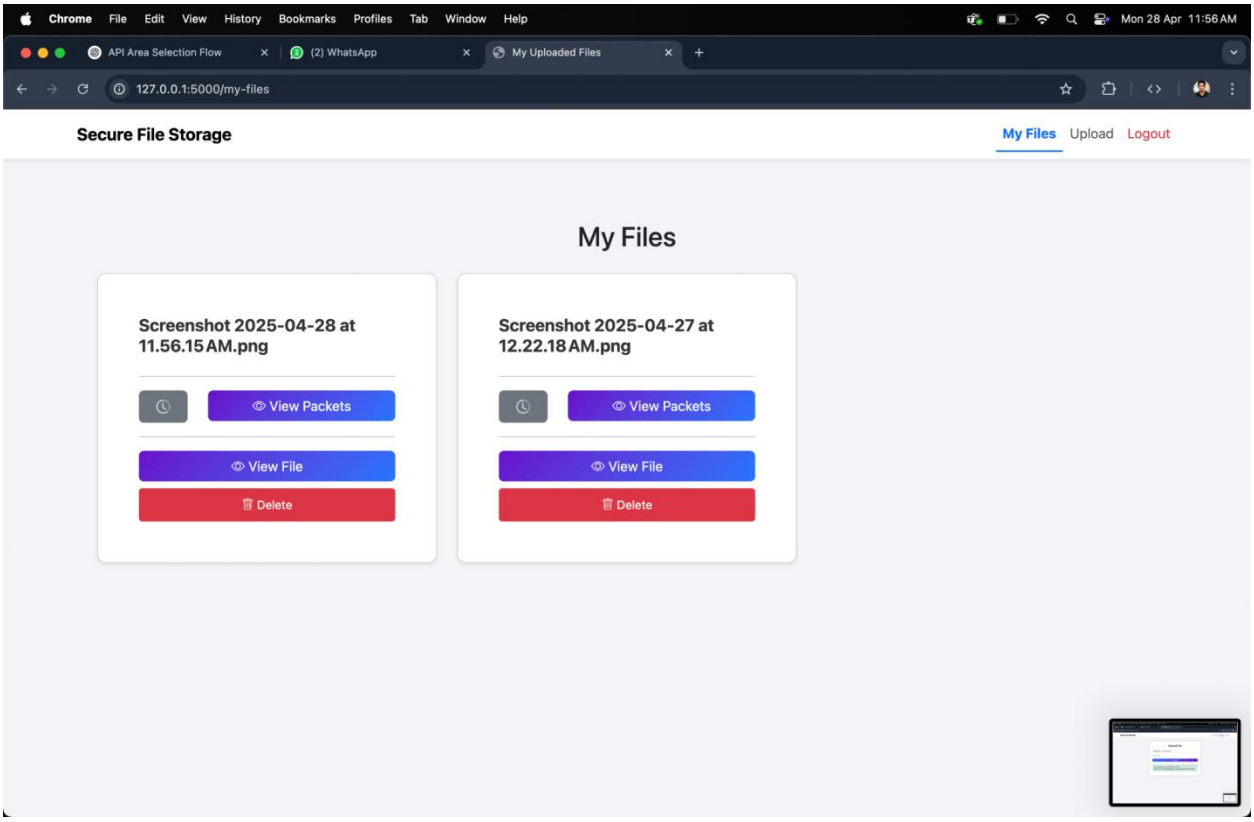


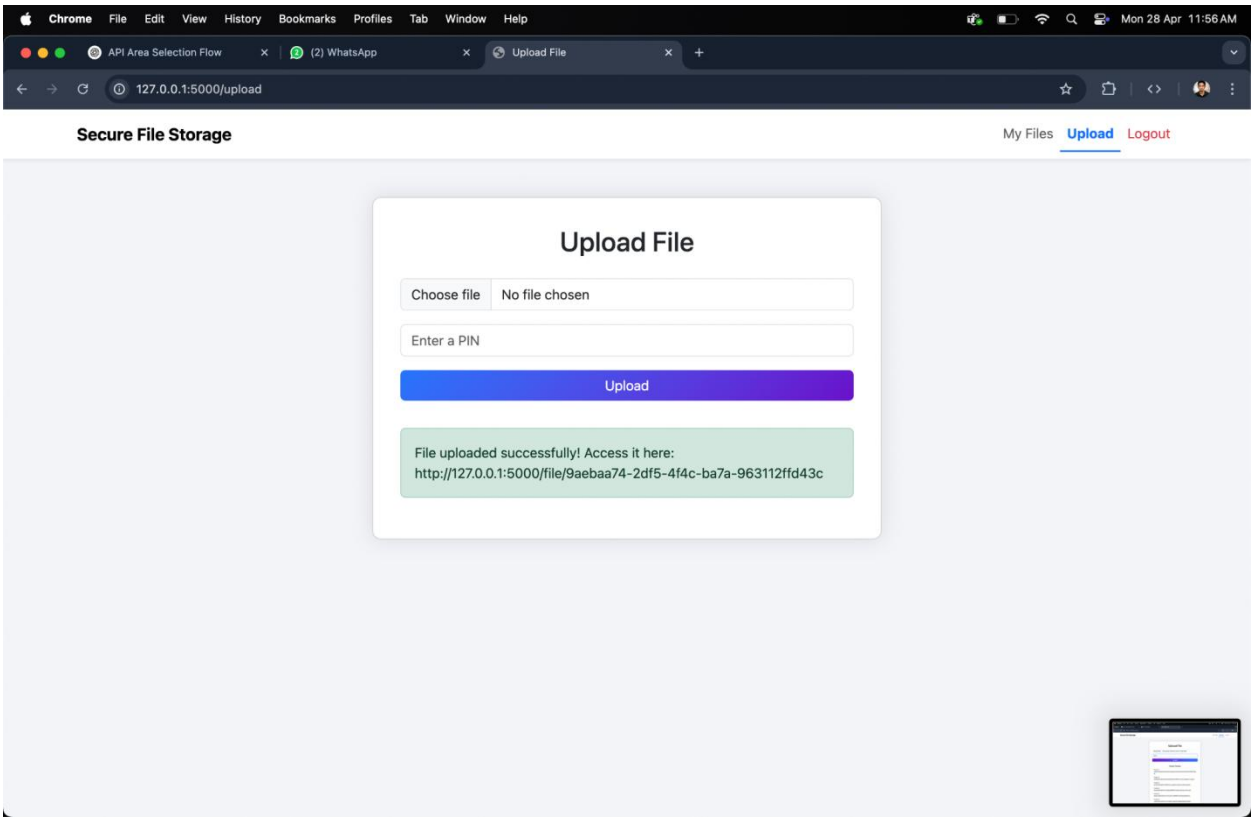


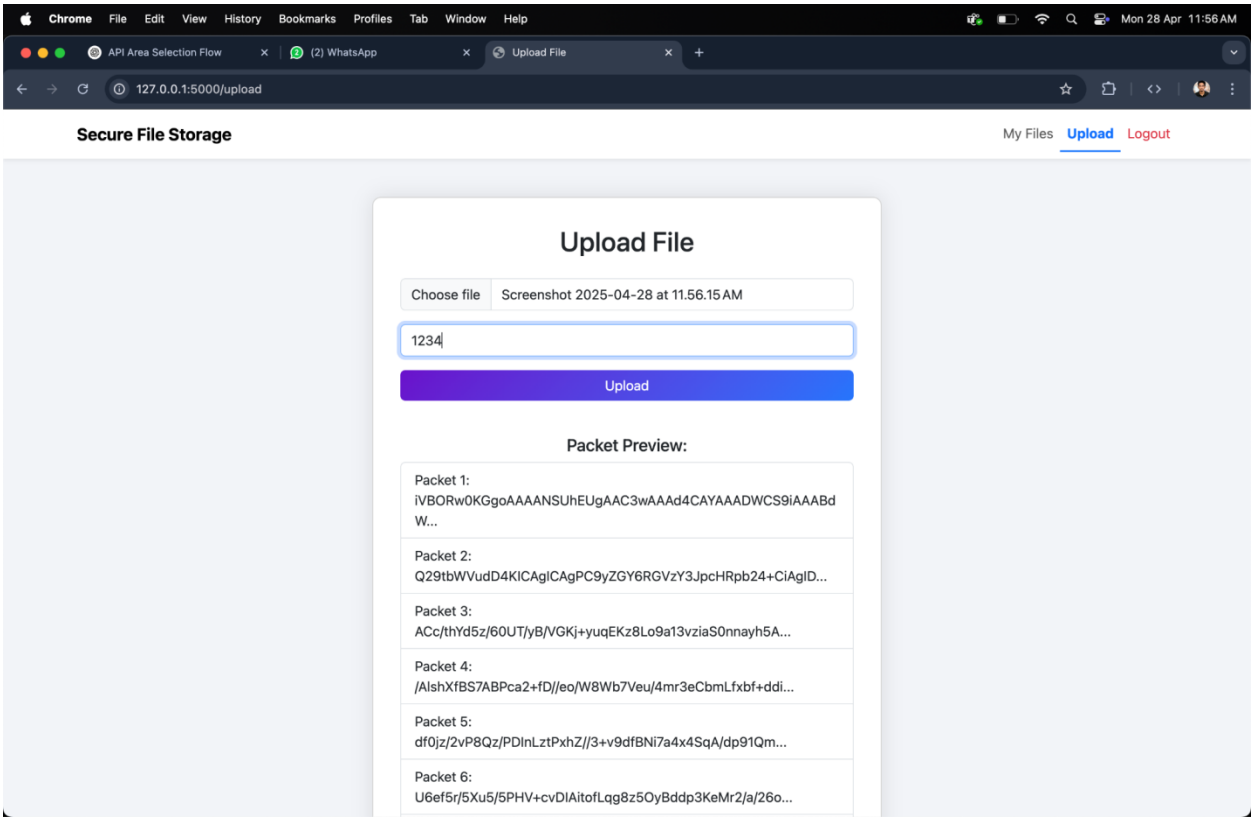

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon 28 Apr 11:56 AM
API Area Selection Flow
(2) WhatsApp
Flask App
127.0.0.1:5000/view_file/9aebaa74-2df5-4f4c-ba7a-963112ffd43c
Secure File Storage
My Files Upload Logout
View File
Filename: Screenshot 2025-04-28 at 11.56.15 AM.png
Image
Cannot preview this file type. You can download it instead.
Download File
Back to My Files
127.0.0.1:5000/file/9aebaa74-2df5-4f4c-ba7a-963112ffd43c/download

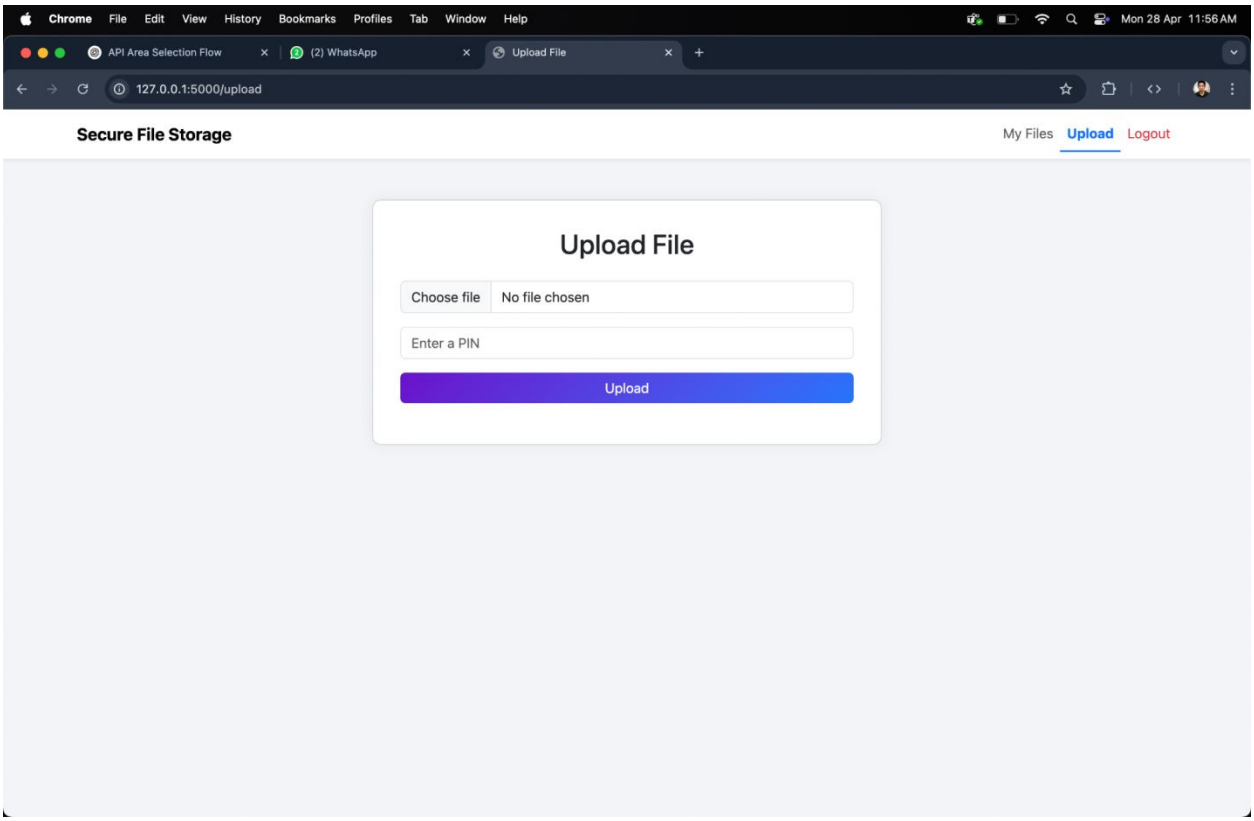

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon 28 Apr 11:56 AM
API Area Selection Flow
(2) WhatsApp
Flask App
127.0.0.1:5000/view_file/9aebaa74-2df5-4f4c-ba7a-963112ffd43c
Secure File Storage
My Files Upload Logout
View File
Filename:
Screenshot 2025-04-28 at 11.56.15 AM.png
PIN
View File

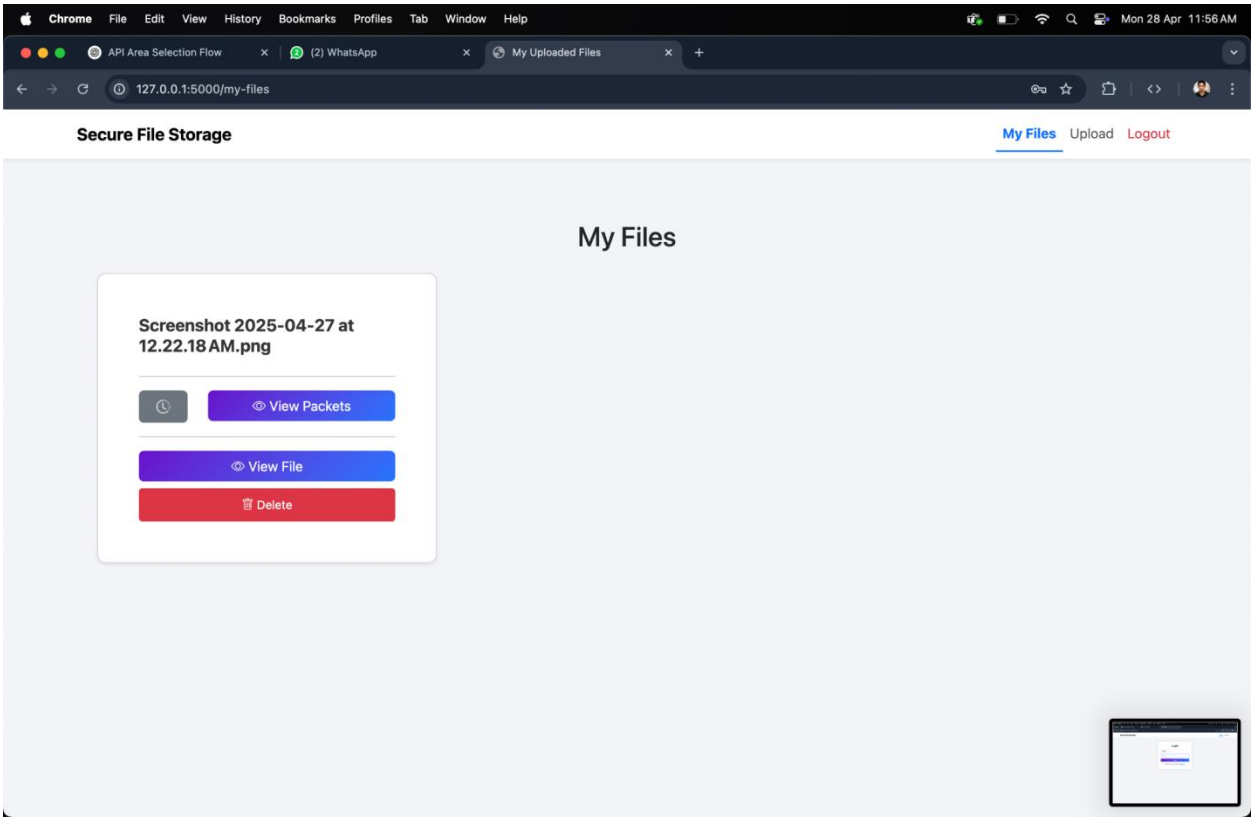

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon 28 Apr 11:56 AM
API Area Selection Flow
(2) WhatsApp
View_Packets
127.0.0.1:5000/file/9aebaa74-2df5-4f4c-ba7a-963112ffd43c
Secure File Storage
My Files Upload Logout
Download File
Packets
Packet 0:
iVBORw0KGgoAAAANSUhEUgAAC3wAAAd4CAYAAADWCS9iAAABdWlDQ1BrQ0d
Db2xvclNwYWNlRGlzcGxheVAzAAAokXWQvUvDUBTFT6tS0DqIDh0cMolD1NlKdn
FoKxRFMFQFq1OafgltfCQpUnETVyn4H1jBWXCwiFRwcXAQRAcR3Zw6KbhoeN6X
VNoi3sfl/Ticc7lcwBtQGSv2AijplpFMxKS11Lrke4OHnlOqZrKooiwK/v276/PR9d5PiFl
Nu3YQ2U9cl84ul3aeAlN//V3Vn8maGv3f1EGNGRbgkYmVbYsJ3iUeMWgp4qrgvMv
HgtMunzuelWSc+JZY0gpqhrhJLKc79HwHl4plrbWD2N6f1VeXxRzqUcxhEyYYilBRg
QQF4X/8044/ji1yV2BQLo8CLMpESRETssTz0KFhEjJxCEHqkLhz634PrfvJbW3vFZh
tcM4v2tpCAzidoZPV29p4BBgaAG7qTDVUR+qh9uZywPsJMJgChu8os2HmwiF3e3
8M6Hvh/GMM8B0CdpXzryPO7RqFn4Er/QcXKWq8UwZBywAAAARjSUNQDA0AAW
4D4+8AAACKZVhJZk1NACoAAAAIAAQBGgAFAAAAAQAAAD4BGwAFAAAAAQAAA
EYBKAADAAAAAQACAACHaQAEAAAAAQAAAE4AAAAAAAAAkAAAAE=
Packet 1:
AAAAkAAAAAEAA5KGAAcAAAASAAAAeKACAAQAAAABAAALfKADAAQAAAABAA
AHeAAAAABBU0NJSQAAAFNjcmVlbnNob3Slm9zAAAAACXBIWXMAABYlAAAWJQ
FJUiTwAAAB2GlUWHRYTUw6Y29tLmFkb2JlLnhtcAAAAAAAPHg6eG1wbWV0YSB
4bWxuczp4PSJhZG9iZTpuczptZXRhLylgeDp4bXB0az0iWE1QIENvcmUgNi4wLjAi
PgoglCA8cmRmOlJERiB4bWxuczpyZGY9Imh0dHA6Ly93d3cudzMub3JnLzE5OTk
vMDIvMjItcmRmLXN5bnRheC1ucyMiPgoglCAgICA8cmRmOkRlc2NyaXB0aW9uIHJ
kZjphYm91dD0iIgoglCAgICAgICAgICB4bWxuczpleGlmPSJodHRwOi8vbnMuYWRv
YmUuY29tL2V4aWYvMS4wLyI+CiAgICAgICAgICAgIDxleGlmOlBpeGVsWURpbWVuc2lv
bj4xOTE+PC9leGlmOlBpeGVsWURpbWVuzsIvbj4KICAgICAgICAgICAgPGV4aWY6UGl4
Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon 28 Apr 11:56 AM
API Area Selection Flow
(2) WhatsApp
Enter Pin
127.0.0.1:5000/file/9aebaa74-2df5-4f4c-ba7a-963112ffd43c
Secure File Storage
My Files Upload Logout
Enter PIN
PIN
View Packets

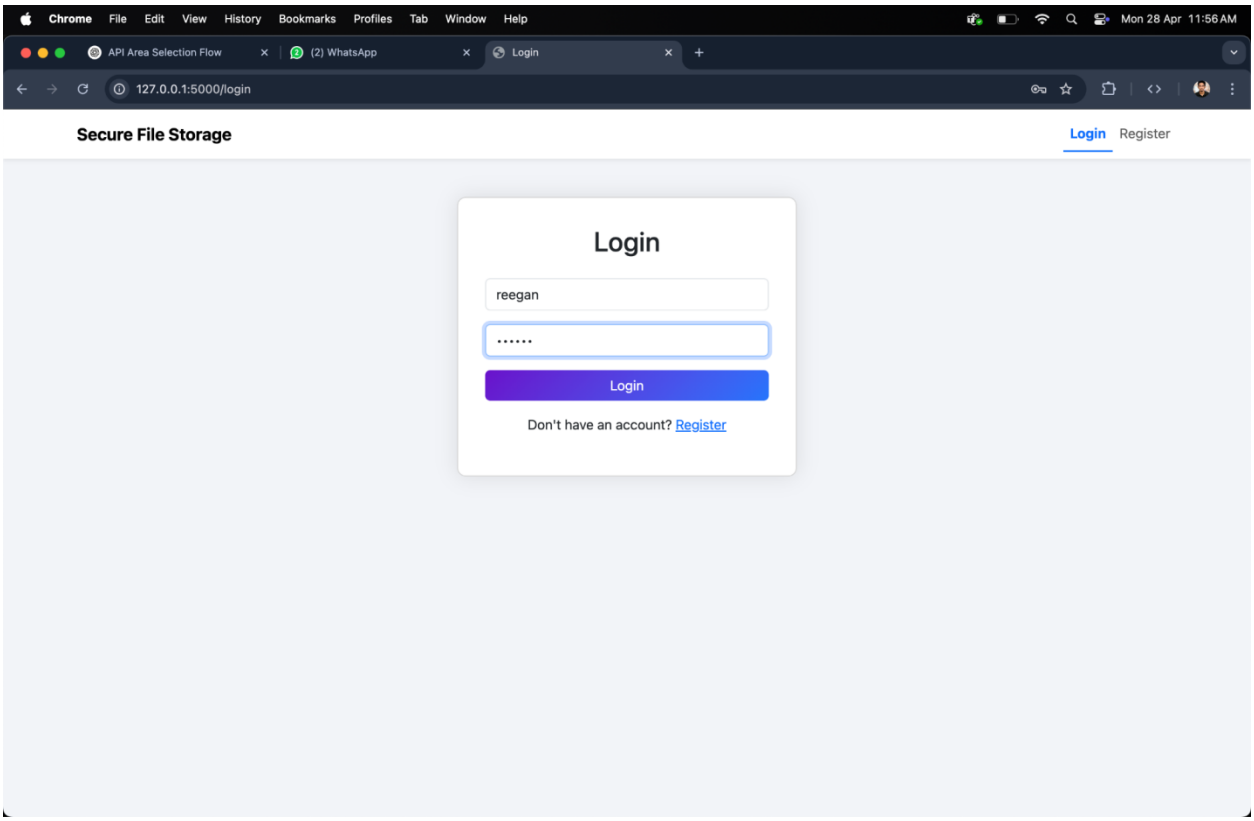

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon 28 Apr 11:56 AM
API Area Selection Flow
(2) WhatsApp
My Uploaded Files
127.0.0.1:5000/my-files
Secure File Storage
My Files Upload Logout
My Files
Screenshot 2025-04-28 at 11.56.15 AM.png
View Packets
View File
Delete
Screenshot 2025-04-27 at 12.22.18 AM.png
View Packets
View File
Delete

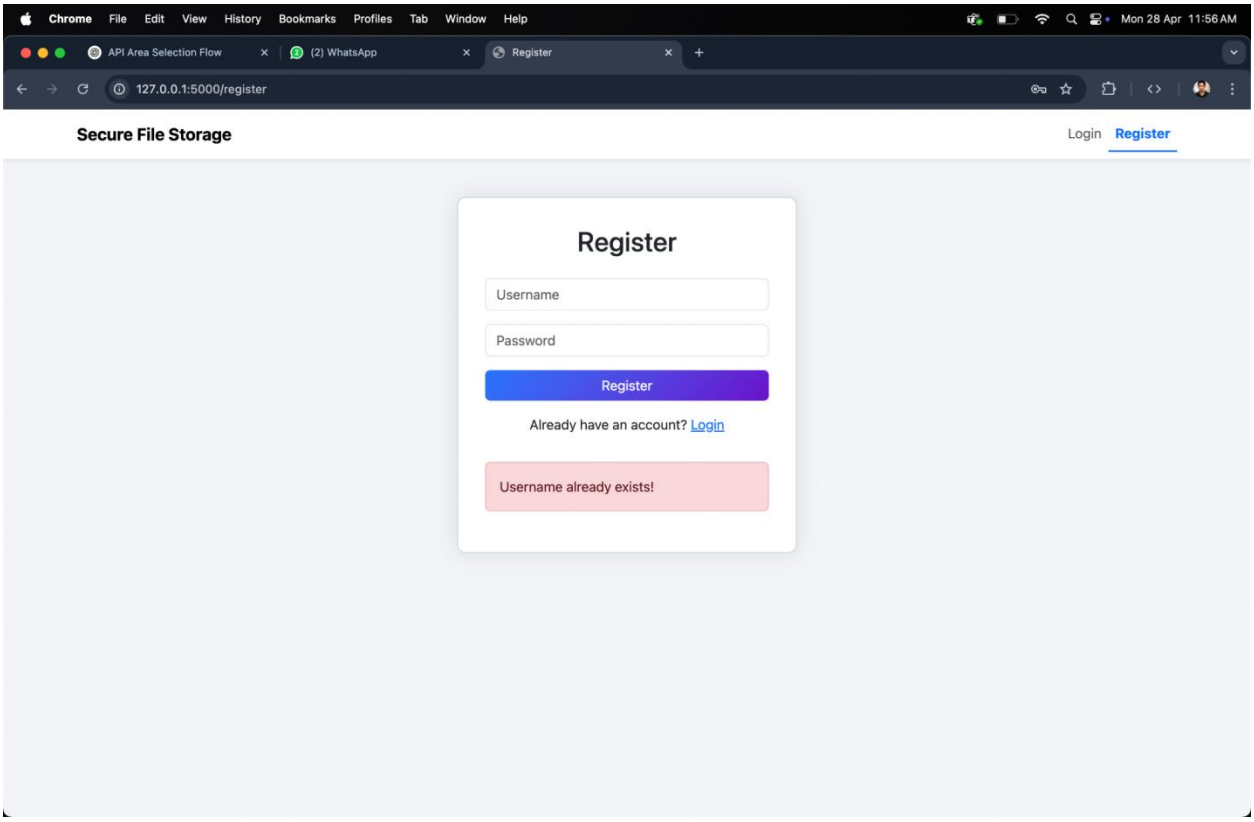

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon 28 Apr 11:56 AM
API Area Selection Flow
(2) WhatsApp
Upload File
127.0.0.1:5000/upload
Secure File Storage
My Files Upload Logout
Upload File
Choose file
No file chosen
Enter a PIN
Upload
File uploaded successfully! Access it here:
http://127.0.0.1:5000/file/9aebaa74-2df5-4f4c-ba7a-963112ffd43c

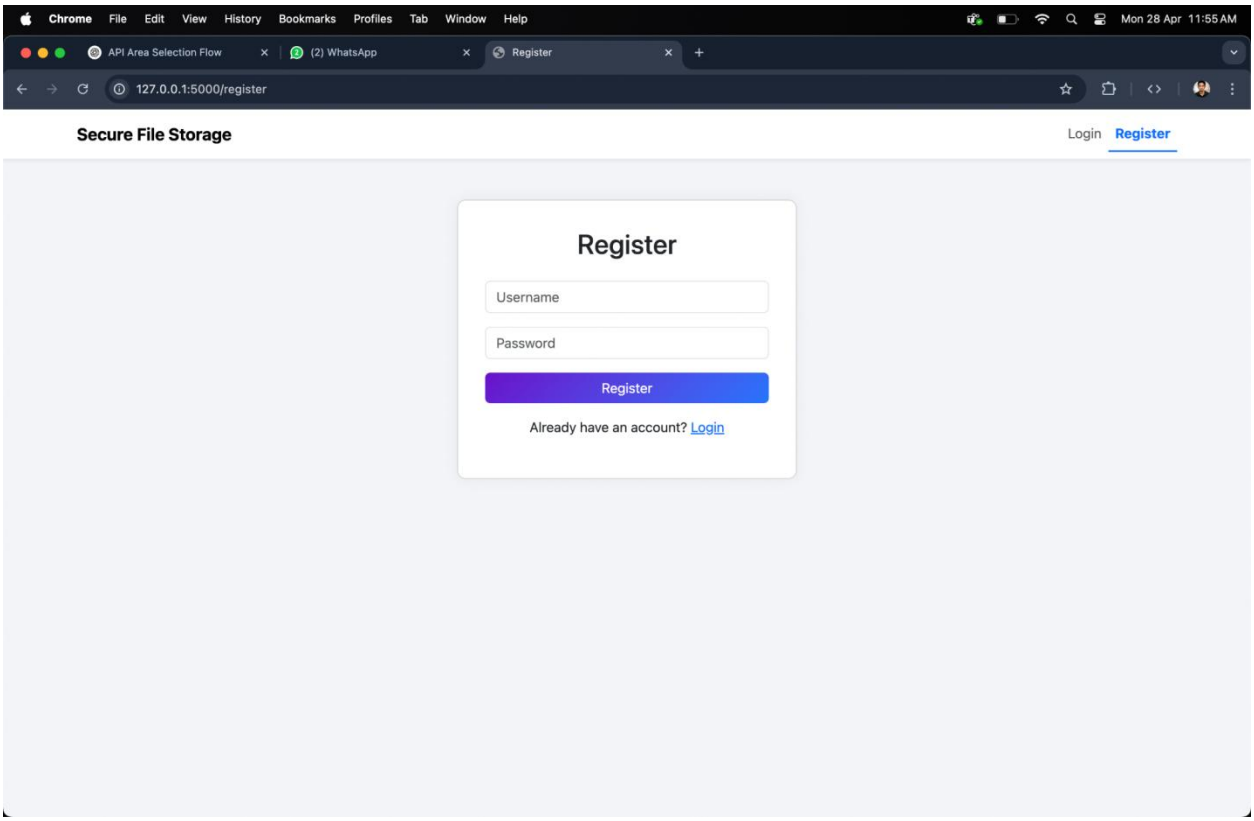

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon 28 Apr 11:56 AM
API Area Selection Flow
(2) WhatsApp
Upload File
127.0.0.1:5000/upload
Secure File Storage
My Files
Upload
Logout
Upload File
Choose file
Screenshot 2025-04-28 at 11.56.15 AM
1234
Upload
Packet Preview:
Packet 1:
iVBORw0KGgoAAAANSUhEUgAAC3wAAAd4CAYAAADWCS9iAAABdW...
Packet 2:
Q29tbWVudD4KICAgICAgPC9yZGY6RGVzY3JpcHRpb24+CiAgID...
Packet 3:
ACc/thYd5z/60UT/yB/VGKj+yuqEKz8Lo9a13vziaS0nnayh5A...
Packet 4:
/AlshXfBS7ABPca2+fD//eo/W8Wb7Veu/4mr3eCbmLfxbf+ddi...
Packet 5:
df0jz/2vP8Qz/PDlnLztPxhZ//3+v9dfBNi7a4x4SqA/dp91Qm...
Packet 6:
U6ef5r/5Xu5/5PHV+cvDlAitofLqg8z5OyBddp3KeMr2/a/26o...

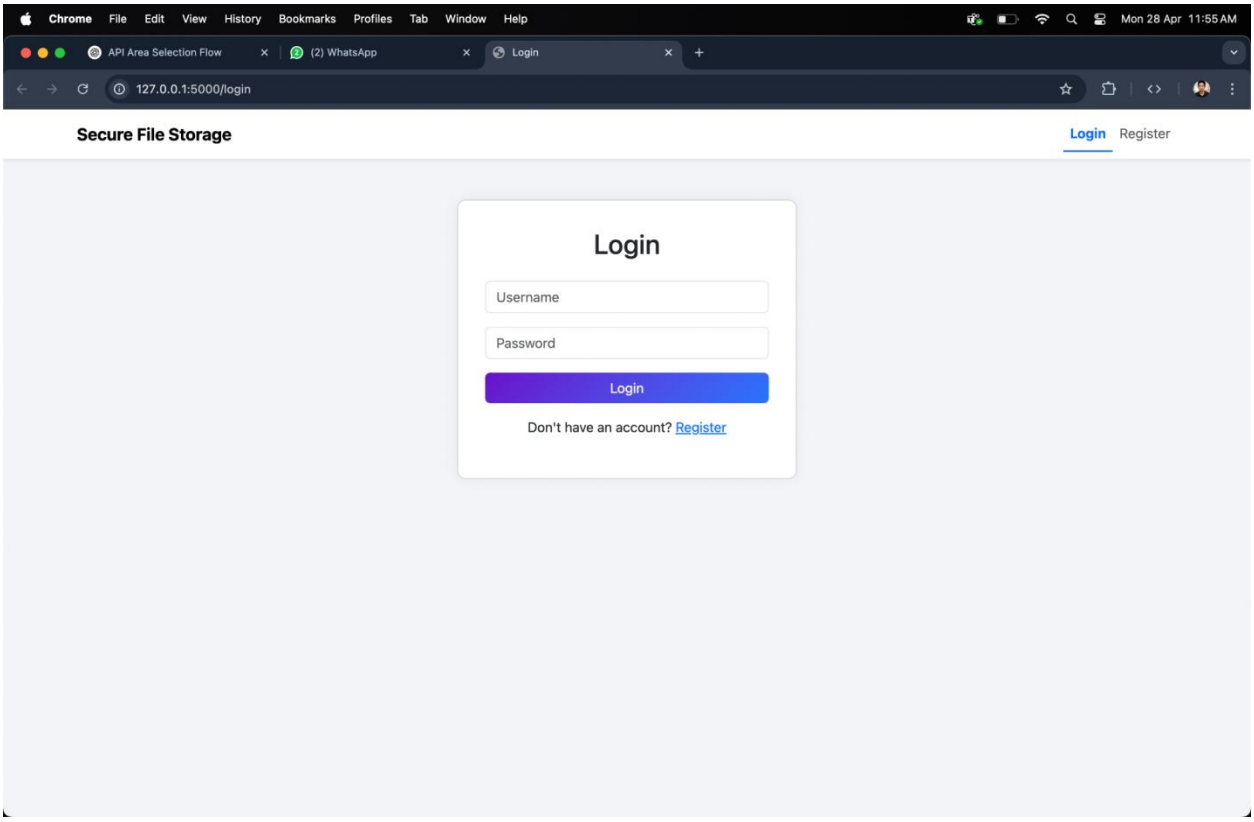

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon 28 Apr 11:56 AM
API Area Selection Flow
(2) WhatsApp
Upload File
127.0.0.1:5000/upload
Secure File Storage
My Files
Upload
Logout
Upload File
Choose file
No file chosen
Enter a PIN
Upload


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon 28 Apr 11:56 AM
API Area Selection Flow
(2) WhatsApp
My Uploaded Files
127.0.0.1:5000/my-files
Secure File Storage
My Files Upload Logout
My Files
Screenshot 2025-04-27 at 12.22.18 AM.png
View Packets
View File
Delete


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon 28 Apr 11:56 AM
API Area Selection Flow
(2) WhatsApp
Login
127.0.0.1:5000/login
Secure File Storage
Login Register
Login
reegan
Login
Don't have an account? Register


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon 28 Apr 11:56 AM
API Area Selection Flow
(2) WhatsApp
Register
127.0.0.1:5000/register
Secure File Storage
Login
Register
Register
Username
Password
Register
Already have an account? Login
Username already exists!


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon 28 Apr 11:55 AM
API Area Selection Flow
(2) WhatsApp
Register
127.0.0.1:5000/register
Secure File Storage
Login
Register
Register
Username
Password
Register
Already have an account? Login


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon 28 Apr 11:55 AM
API Area Selection Flow
(2) WhatsApp
Login
127.0.0.1:5000/login
Secure File Storage
Login
Register
Login
Username
Password
Login
Don't have an account? Register

CHAPTER 8

CONCLUSION

In this work, we presented an enhanced approach to privacy-preserving content-based retrieval in cloud repositories, addressing the growing need for secure and efficient access to outsourced data. Our proposed system successfully balances the dual goals of maintaining user privacy and enabling effective content-based search. By integrating advanced encryption schemes with secure indexing and query mechanisms, we ensure that sensitive information remains protected from both external threats and semi-trusted cloud service providers.

The implementation demonstrates significant improvements in retrieval accuracy, search efficiency, and resistance to inference attacks when compared to traditional methods. Moreover, the system supports scalability and adaptability to various cloud environments, making it suitable for real-world deployment.

Future work can explore the incorporation of machine learning techniques to further enhance retrieval accuracy while maintaining privacy guarantees. Additionally, expanding support for multimedia data and improving resistance against more sophisticated threat models would strengthen the robustness of the solution.

CHAPTER 9

FUTURE ENHANCEMENT

Homomorphic Encryption:

This allows computations to be performed on encrypted data without decrypting it first, ensuring that the cloud server only sees encrypted data and cannot learn the underlying content.

Searchable Encryption:

This enables searching within encrypted data using keywords or other criteria, while still protecting the privacy of the actual content, according to a research article.

Secure Multi-Party Computation:

This technique allows multiple parties to collaboratively perform computations without revealing their individual inputs or intermediate results, according to a research article.

CHAPTER 10

REFERENCES

[1] G. Ding, Y. Guo, and J. Zhou, “Collective matrix factorization hashing for multimodal data,” in Proc. IEEE Conf. Comput. Vis. Pattern Recog., 2021, pp. 2083–2090.

[2] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin, “Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 35, no. 12, pp. 2916–2929, Dec. 2023.

[3] H. J_egou, F. Perronnin, M. Douze, J. S_anchez, P. P_erez, and C. Schmid, “Aggregating local image descriptors into compact codes,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 34, no. 9, pp. 1704–1716, Sep. 2020.

[4] A. Karpathy and L. Fei-Fei, “Deep visual-semantic alignments for generating image descriptions,” in Proc. IEEE Conf. Comput. Vis. Pattern Recog., Boston, MA, USA, Jun. 2015, pp. 3128–3137.

[5] Y. Pan, T. Yao, T. Mei, H. Li, C.-W. Ngo, and Y. Rui, “Clickthrough-based cross-view learning for image search,” in Proc. 37th Int.ACMSIGIR Conf. Res. Develop. Inf. Retrieval, 2014, pp. 717–726.

[6] J. Song, Y. Yang, Y. Yang, Z. Huang, and H. T. Shen, “Inter-media hashing for large-scale retrieval from heterogeneous data sources,” in Proc. Int. Conf. Manage. Data, 2013, pp. 785–796.

[7] Z. Yu, F. Wu, Y. Yang, Q. Tian, J. Luo, and Y. Zhuang, “Discriminative coupled dictionary hashing for fast cross-media retrieval,” in Proc. 37th Int. ACM SIGIR Conf. Res. Develop. Inf.Retrieval, 2017, pp. 395–404.

[8] D. Zhai, H. Chang, Y. Zhen, X. Liu, X. Chen, and W. Gao, “Parametric local multimodal hashing for cross-view similarity search,” in Proc. 23rd Int. Joint Conf. Artif. Intell., 2021, pp. 2754–2760.

[9] J. Zhou, G. Ding, and Y. Guo, “Latent semantic sparse hashing for cross-modal similarity search,” in Proc. 37th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval, 2019, pp. 415–424.

[10] H. Zhang, J. Yuan, X. Gao, and Z. Chen, “Boosting cross-media retrieval via visual-auditory feature analysis and relevance feedback,” in Proc. ACM Int. Conf. Multimedia, 2014, pp. 953–956.