BOOSTING CLOUD SECURITY AND PERFORMANCE THROUGH SMART DATA DISTRIBUTION

A PROJECT REPORT

Submitted by

B.MUTHU	**(REG.NO:821921104047)**
J.SURENDAR	**(REG.NO:821921104072)**
T.TAMIL SELVAN	**(REG.NO:821921104074)**
Y.VENGADESAN	**(REG.NO:821921104079)**

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 **"BOOSTING CLOUD SECURITY AND PERFORMANCE THROUGH SMART DATA DISTRIBUTION"** is the bonafide certificate work of **Mr. B. MUTHU (821921104047), Mr. J. SURENDAR (821921104072), Mr. T. TAMIL SELVAN (821921104074), Mr. Y. VENGADESAN (821921104079)** who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Mr. P. MUTHAMIL SELVAN, M.E.,	**Mrs. S. MADHIVADHANI, M.E.,**
HEAD OF THE DEPARTMENT	**SUPERVISOR**
Department of Computer Science &Engg.	Department of Computer Science &Engg.
St.Joseph's College of Engg & Tech	St.Joseph's College of Engg & Tech
Thanjavur	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, B.E., 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 her wholehearted encouragement throughout this project.

We like to express our sincere gratitude to our internal project guide **Mrs. S. MADHIVADHANI, 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

In the rapidly evolving digital landscape, cloud computing has become the backbone of modern data storage and processing. However, as reliance on cloud infrastructure grows, so do concerns over data security, integrity, and performance. This paper proposes an integrated framework that enhances cloud security and performance by leveraging smart data distribution techniques alongside robust cryptographic methods. Smart data distribution ensures optimal allocation and redundancy of data across multiple cloud nodes, minimizing latency and increasing fault tolerance. Meanwhile, advanced cryptographic mechanisms including homomorphism encryption, attribute-based encryption, and zero-knowledge proofs safeguard sensitive information against unauthorized access and breaches. Together, these techniques create a synergistic system where data remains secure without compromising on access speed or computational efficiency. The proposed approach is evaluated through simulations and real-world scenarios, demonstrating significant improvements in data confidentiality, system responsiveness, and overall cloud resource utilization. This study highlights the potential for intelligent, security-conscious cloud architectures to meet the demands of next-generation applications.

சுருக்கம்

வேகமாக வளர்ந்து வரும் டிஜிட்டல் உலகில், நவீன தரவு சேமிப்பு மற்றும் செயலாக்கத்தின் முதுகெலும்பாக கிளவுட் கம்ப்யூட்டிங் மாறியுள்ளது. இருப்பினும், கிளவுட் உள்கட்டமைப்பை நம்பியிருப்பது வளரும்போது, தரவு பாதுகாப்பு, ஒருமைப்பாடு மற்றும் செயல்திறன் குறித்த கவலைகளும் அதிகரித்து வருகின்றன. வலுவான கிரிப்டோகிராஃபிக் முறைகளுடன் ஸ்மார்ட் டேட்டா விநியோக நுட்பங்களைப் பயன்படுத்துவதன் மூலம் கிளவுட் பாதுகாப்பு மற்றும் செயல்திறனை மேம்படுத்தும் ஒருங்கிணைந்த கட்டமைப்பை இந்த ஆய்வு முன்மொழிகிறது. ஸ்மார்ட் டேட்டா விநியோகம் பல கிளவுட் நோடுகளில் தரவின் உகந்த ஒதுக்கீடு மற்றும் பணிநீக்கத்தை உறுதி செய்கிறது, தாமதத்தைக் குறைக்கிறது மற்றும் தவறு சகிப்புத்தன்மையை அதிகரிக்கிறது. இதற்கிடையில், ஹோமோமார்பிசம் குறியாக்கம், பண்புக்கூறு அடிப்படையிலான குறியாக்கம் மற்றும் பூஜ்ஜிய-அறிவு சான்றுகள் உள்ளிட்ட மேம்பட்ட கிரிப்டோகிராஃபிக் வழிமுறைகள் அங்கீகரிக்கப்படாத அணுகல் மற்றும் மீறல்களுக்கு எதிராக முக்கியமான தகவல்களைப் பாதுகாக்கின்றன. ஒன்றாக, இந்த நுட்பங்கள் அணுகல் வேகம் அல்லது கணக்கீட்டு செயல்திறனில் சமரசம் செய்யாமல் தரவு பாதுகாப்பாக இருக்கும் ஒரு ஒருங்கிணைந்த அமைப்பை உருவாக்குகின்றன. முன்மொழியப்பட்ட அணுகுமுறை உருவகப்படுத்துதல்கள் மற்றும் நிஜ உலக சூழ்நிலைகள் மூலம் மதிப்பிடப்படுகிறது, இது தரவு ரகசியத்தன்மை, கணினி மறுமொழி மற்றும் ஒட்டுமொத்த கிளவுட் வள பயன்பாட்டில் குறிப்பிடத்தக்க முன்னேற்றங்களை நிரூபிக்கிறது. அடுத்த தலைமுறை பயன்பாடுகளின் தேவைகளைப் பூர்த்தி செய்வதற்கான அறிவார்ந்த பாதுகாப்பு உணர்வுள்ள கிளவுட் கட்டமைப்புகளுக்கான திறனை இந்த ஆய்வு எடுத்துக்காட்டுகிறது.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

In the age of cloud computing, businesses and individuals alike rely heavily on the cloud to store, process, and access massive volumes of data. With this growing dependency comes the critical need to ensure both the security and performance of cloud-based systems. One of the most effective strategies to address these concerns is the intelligent distribution and duplication of data across cloud environments.

Smart data distribution ensures that data is efficiently allocated across various servers and regions, optimizing latency, load balancing, and fault tolerance. Meanwhile, strategic data duplication adds a robust layer of security and reliability by maintaining redundant copies in multiple locations. This approach not only minimizes data loss in the event of system failures or cyber-attacks but also supports faster data retrieval, improving overall system responsiveness.

By combining these techniques, organizations can significantly enhance their cloud infrastructure ensuring data integrity, reducing downtime, and delivering a seamless user experience. This paper explores the methodologies, benefits, and challenges associated with smart data distribution and duplication, and how they collectively contribute to a more secure and high-performing cloud ecosystem.

CHAPTER 2

LITERATURE SURVEY

2.1 Division and Replication of Data in Cloud for Optimal Performance and Security

Outsourcing data to a third-party administrative control, as is done in cloud computing, gives rise to security concerns. The data compromise may occur due to attacks by other users and nodes within the cloud. Therefore, high security measures are required to protect data within the cloud. However, the employed security strategy must also take into account the optimization of the data retrieval time. In this paper, we propose division and replication of data in the cloud for optimal performance and security (DROPS) that collectively approaches the security and performance issues. In the DROPS methodology, we divide a file into fragments, and replicate the fragmented data over the cloud nodes. Each of the nodes stores only a single fragment of a particular data file that ensures that even in case of a successful attack, no meaningful information is revealed to the attacker. Moreover, the nodes storing the fragments, are separated with certain distance by means of graph T-coloring to prohibit an attacker of guessing the locations of the fragments. Furthermore, the DROPS methodology does not rely on the traditional cryptographic techniques for the data security; thereby relieving the system of computationally expensive methodologies. We show that the probability to locate and compromise all of the nodes storing the fragments of a single file is extremely low. We also compare the performance of the DROPS methodology with 10 other schemes. The higher level of security with slight performance overhead was observed.

2.2 Data Distribution Scheme For a Survivable Storage System

We present a new verifiable secret redistribution protocol for threshold sharing schemes that forms a key component of a proposed archival storage system. Our protocol supports redistribution from (m,n) to (m',n') threshold sharing schemes without requiring reconstruction of the original data. The design is motivated by archive systems for which the added security of threshold sharing of data must be accompanied by the flexibility of dynamic shareholder changes. Our protocol enables the dynamic addition or removal of shareholders, and also guards against mobile adversaries. We observe that existing protocols either cannot be extended readily to allow redistribution between different access structures, or have vulnerabilities that allow faulty old shareholders to distribute invalid shares to new shareholders. Our primary contribution is that in our protocol, new shareholders can verify the validity of their shares after redistribution between different access structures.

2.3 Internet data replication techniques

Re-appropriating information to an untouchable authoritative control, as is done in cloud dealing with, offers move to security concerns. The information arrangement may occur in perspective on assaults by different clients and focus focuses inside the cloud. In this way, high safety efforts are required to ensure information inside the cloud. Notwithstanding, the utilized security framework should besides consider the improvement of the information recovery time. In this paper, we propose Division and Replication of Data in the Cloud for Optimal

Performance and Security (DROPS) those outright philosophies the security and execution issues. In the DROPS framework, we separate a file into pieces, and mirror the divided information over the cloud focus focuses. The majority of the inside point's stores just a solitary zone of a specific information record that guarantees that paying little mind to whether there should be an occasion of a beneficial trap, no fundamental data is uncovered to the attacker. Additionally, the inside focuses verifying the pieces are separated with certain parcel by techniques for layout T-shading to keep an assailant from guaranteeing speculating the domains of the pieces. What is more, the DROPS approach does not depend upon the standard cryptographic methodology for the information security; thusly calming the game-plan of computationally costly frameworks. We display that the likelihood to find and arrangement a large portion of the focuses verifying the bits of a solitary record is extraordinarily low. We also dismember the execution of the DROPS philosophy with ten particular plans. The more raised proportion of security with slight execution overhead.

2.4 Data Replication In Cloud Computing Datacenters

Cloud computing is an emerging paradigm that provides computing resources as a service over a network. Communication resources often become a bottleneck in service provisioning for many cloud applications. Therefore, data replication, which brings data (e.g., databases) closer to data consumers (e.g., cloud applications), is seen as a promising solution. It allows minimizing network delays and bandwidth usage. In this paper we study data replication in cloud computing data centers. Unlike other approaches available in the literature, we consider both energy efficiency and bandwidth consumption of the system, in addition to the improved Quality of Service (QoS) as a result of the reduced communication delays. The evaluation results obtained during extensive simulations help to unveil performance and energy efficiency tradeoffs and guide the design of future data replication solutions.

2.5 Distributed Data Replication Using Genetic Algorithms

Fast dissemination and access of information in large distributed systems, such as the Internet, has become a norm of our daily life. However, undesired long delays experienced by end-users, especially during the peak hours, continue to be a common problem. Replicating some of the objects at multiple sites is one possible solution in decreasing network traffic. The decision of what to replicate where, requires solving a constraint optimization problem which is NP-complete in general. Such problems are known to stretch the capacity of a Genetic Algorithm (GA) to its limits. Nevertheless, we propose a GA to solve the problem when the read/write demands remain static and experimentally prove the superior solution quality obtained compared to an intuitive greedy method. Unfortunately, the static GA approach involves high running time and may not be useful when read/write demands continuously change, as is the case with breaking news. To tackle such case we propose a hybrid GA that takes as input the current replica distribution and computes a new one using knowledge about the network attributes and the changes occurred. Keeping in view more pragmatic scenarios in today's distributed information environments, we evaluate these algorithms with respect to the storage capacity constraint of each site as well as variations in the popularity of objects, and also examine the trade-off between running time and solution quality

CHAPTER 3
SYSTEM ANALYSIS

3.1 Existing System

In the current cloud computing landscape, data storage and management systems largely rely on centralized architectures. Data is typically stored on single or replicated cloud servers managed by service providers. While this model offers scalability and accessibility, it introduces several vulnerabilities and limitations, especially in terms of **security, performance, and data integrity**.

Drawbacks of existing systems:

1. **Centralized Data Storage:**
    - Most cloud platforms store user data on centralized servers.
    - A single point of failure can compromise data availability and security.
    - It also makes the system a prime target for cyber attacks like DDoS or ransom ware.
2. **Limited Use of Cryptography:**
    - While basic encryption (such as SSL/TLS, AES) is used during data transit and at rest, many systems don't implement **end-to-end encryption** or **advanced cryptographic techniques** like homomorphism encryption or multi-party computation.
    - Key management is often handled by the provider, raising trust concerns.
3. **Inefficient Data Distribution:**
    - Current systems may replicate data for fault tolerance, but they don't optimize for **load balancing** or **geographic proximity**, which can cause performance lags.

   - Smart data distribution strategies (e.g., based on access patterns, network latency, or user behaviour) are underutilized.
4. **Security Risks:**
   - Insider threats, unauthorized access, data breaches, and insecure APIs remain common risks.
   - Lack of transparency in how data is managed and protected.
5. **Compliance and Trust Issues:**
   - Users have limited control over their data once it is uploaded.
   - Meeting compliance standards like GDPR, HIPAA, etc., can be complex with current architectures.

3.2 Proposed System

The proposed system aims to enhance cloud computing environments by integrating intelligent data distribution mechanisms with advanced cryptographic techniques. This dual approach is designed to address two critical challenges in cloud computing: **data security** and **system performance**.

At its core, the system introduces a **Smart Data Distribution Framework** that analyzes the nature, sensitivity, and usage frequency of data to determine optimal storage and transmission paths across multiple cloud servers. This intelligent distribution minimizes latency, balances the load, and ensures that critical data is readily available while maintaining efficient resource utilization.

To reinforce data confidentiality and integrity, the system incorporates **robust cryptographic algorithms** such as AES (Advanced Encryption Standard) for data-at-rest and TLS (Transport Layer Security) for data-in-transit. In addition, the system may leverage **homomorphic encryption** and **attribute-based encryption**

(ABE) to enable secure computations and fine-grained access control without exposing the underlying data.

Advantages of the proposed system:

- **Dynamic Data Classification**: Automatically categorizes data based on sensitivity and access patterns.
- **Performance Monitoring Module**: Continuously assesses system throughput and dynamically adjusts distribution policies to maintain high performance.
- **Scalability and Fault Tolerance**: Ensures the system adapts to varying workloads and handles failures gracefully without compromising data security.

3.3 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 4

SOFTWARE SPECIFICATION

4.1 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: <p>. 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: </p>. 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:

<p>This is a paragraph.</p>

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.

avaScript 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.

4.2 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.

CHAPTER 5

SOFTWARE DEVELOPMENT

5.1 MODULES DESCRIPTION

1. Cloud Client

- Data Owner
- Data User
- Admin

2. Cloud Server

- Fragmentation
- Encryption & Decryption
- Allocation

1) Cloud Client: - Cloud client should be Data owner or Data user.

5.1.1 Data Owner: -Data owner is responsible for uploading file on cloud as well as view files uploaded by him or others. Data owner has information about the placed fragment and its replicas with their node numbers in cloud.

Data Owner Workflow

1. Authentication

- **Step 1.1:** Data Owner logs into the system.
- **Step 1.2:** Credentials are verified by the **Admin**.
- **Step 1.3:** On successful verification, access is granted to the Data Owner dashboard.

2. File Upload

- **Step 2.1:** Data Owner selects the file to upload.
- **Step 2.2:** The file is sent to the **Cloud Server** for processing.

3. File Processing on Cloud Server

- **Step 3.1: Fragmentation Module** splits the file into multiple fragments.
- **Step 3.2: Encryption Module** encrypts each file fragment.
- **Step 3.3: Replication** (optional): Creates replicas of each fragment for redundancy.
- **Step 3.4: Allocation Module** assigns fragments (and replicas) to different cloud storage nodes.
- **Step 3.5:** Metadata (fragment location, node ID, etc.) is sent back to the Data Owner.

4. Metadata Management

- **Step 4.1:** Data Owner stores metadata about:
    - Fragment identifiers
    - Encryption keys (if applicable)
    - Node numbers / locations
- **Step 4.2:** This metadata is required later for file retrieval and verification.

5. View Uploaded Files

- **Step 5.1:** Data Owner can view a list of files they have uploaded.

- **Step 5.2:** Can also see files uploaded by others (if permitted).
- **Step 5.3:** Option to manage files (e.g., delete, update, or track access).

6. File Sharing / Access Control (Optional)

- **Step 6.1:** Data Owner can share file access with authenticated Data Users.
- **Step 6.2:** Admin may validate or log these sharing activities.

7. Logout

- **Step 7.1:** Data Owner logs out of the system securely.

5.1.2 Data User: -Data user is the one who is responsible for downloading files or view files uploaded by others. To download file from cloud he has to be authenticated user otherwise he will be considered as attacker.

Data User Workflow

1. Authentication

- **Step 1.1:** Data User accesses the system and logs in.
- **Step 1.2:** Credentials are sent to the **Admin** for validation.
- **Step 1.3:** On successful authentication, access is granted to the Data User dashboard.
- **Step 1.4:** If authentication fails, the user is treated as an **attacker** and access is denied.

2. File Browsing

- **Step 2.1:** Data User can view a list of available files uploaded by authorized **Data Owners**.
- **Step 2.2:** May use search or filter options to find specific files.

3. File Request / Download

- **Step 3.1:** User selects a file to download.
- **Step 3.2:** System verifies if the user has access permission to the selected file.
- **Step 3.3:** If access is granted:
    - The system fetches file fragment metadata (locations, keys, etc.).
    - Requests each encrypted fragment from respective cloud storage nodes.

4. File Reconstruction on Client Side

- **Step 4.1:** The encrypted fragments are received by the Data User.
- **Step 4.2:** The **Decryption Module** decrypts the fragments using keys (if provided).
- **Step 4.3:** The **Reconstruction Module** reassembles the original file from decrypted fragments.

5. File Viewing / Usage

- **Step 5.1:** Data User views or uses the file as needed after full reconstruction.

- **Step 5.2:** Actions may be logged for auditing by the **Admin**.

6. Logout

- **Step 6.1:** Data User securely logs out of the system.

5.1.3 Admin: -Admin is an authorized person who has rights to validate authorized data owner and user. He is also responsible for allocation of block and maintains information and authentication.

Admin Workflow

1. Admin Login

- **Step 1.1:** Admin accesses the system via the admin portal.
- **Step 1.2:** Admin enters credentials for authentication.
- **Step 1.3:** On successful login, Admin is granted access to the admin dashboard.

2. User Validation

- **Step 2.1:** Admin reviews new registration requests from Data Owners and Data Users.
- **Step 2.2:** Verifies identity and authenticity of each user.
- **Step 2.3:** Approves or rejects user access.
- **Step 2.4:** Maintains a list of valid and active users.

3. File & Storage Monitoring

- **Step 3.1:** Admin monitors all uploaded files and their status.
- **Step 3.2:** Checks fragment placement and ensures replication strategies are followed.
- **Step 3.3:** Oversees fragment allocation across cloud storage nodes.

4. Block & Fragment Allocation

- **Step 4.1:** When a file is uploaded by a Data Owner, Admin assists or validates:
    - Fragmentation logic
    - Replication of file fragments
    - Allocation of fragments to specific nodes in the cloud
- **Step 4.2:** Ensures fragments are distributed securely and efficiently.

5. Security Management

- **Step 5.1:** Detects unauthorized access attempts (e.g., failed Data User login).
- **Step 5.2:** Maintains logs of user actions, access times, and file transfers.
- **Step 5.3:** Flags suspicious behavior or access anomalies for review.

6. Access Control Management

- **Step 6.1:** Manages user roles and file permissions.
- **Step 6.2:** Grants/revokes access to specific files or features based on role (Owner/User).

7. System Maintenance & Audit

- **Step 7.1:** Performs regular audits of user activity and file integrity**.**
- **Step 7.2:** Maintains backup plans, and ensures high availability and performance.
- **Step 7.3:** May update encryption standards or security policies.

8. Logout

- **Step 8.1:** Admin securely logs out after completing tasks.

2) Cloud Server:-

5.1.4 Fragmentation: -This approach is used for fragmenting the file for security purpose at sever side. This approach runs the Fragmentation algorithm. It has file as input and produces the file fragments as output.

5.1.5 Encryption & Decryption: **-** Encryption is the process of converting readable data (plaintext) into an unreadable format (ciphertext), while decryption is the reverse process of converting ciphertext back into plaintext. Encryption protects data from unauthorized access, and decryption is used to restore data to its original form.

5.1.6 Allocation: -After the file is spitted and replicas are generated then we have to allocate that fragments at cloud server for storing data. While storing or allocating that fragments we have consider security issues.

CHAPTER 6

SYSTEM DESIGN AND ANALYSIS

6.1 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.

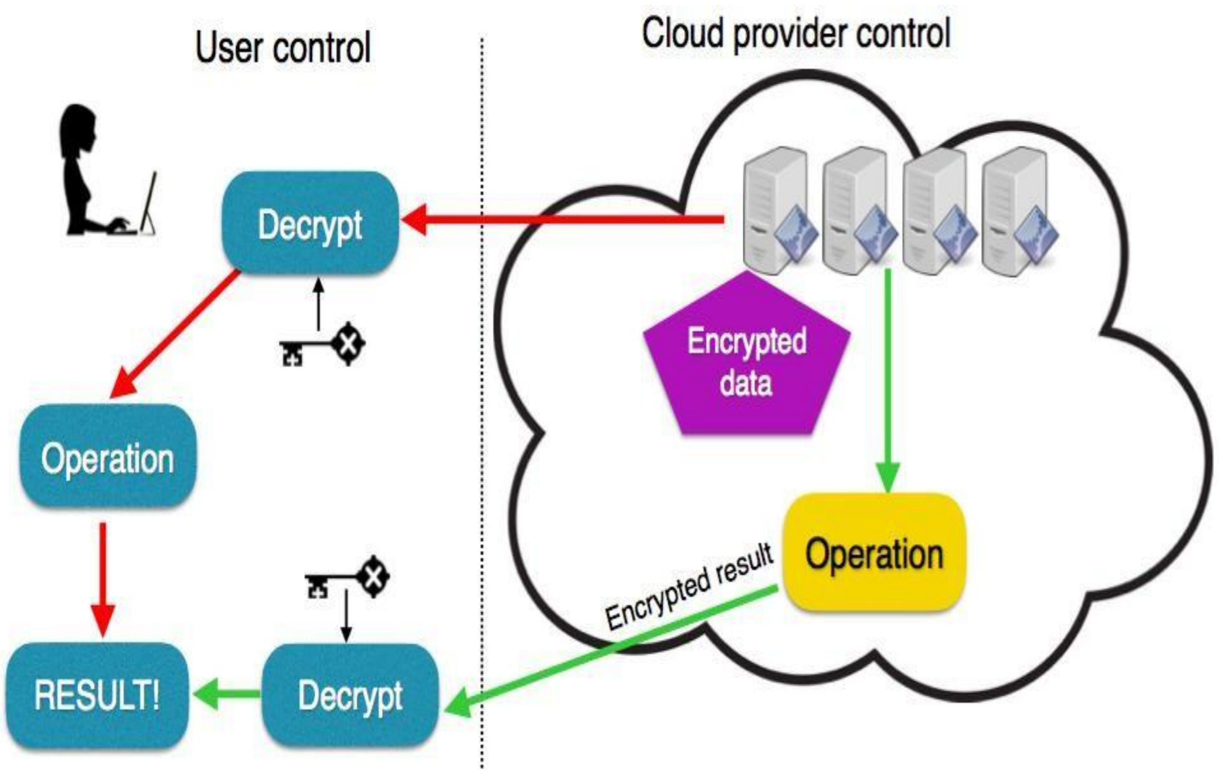


Fig5.1 Architecture Diagram

6.2 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.

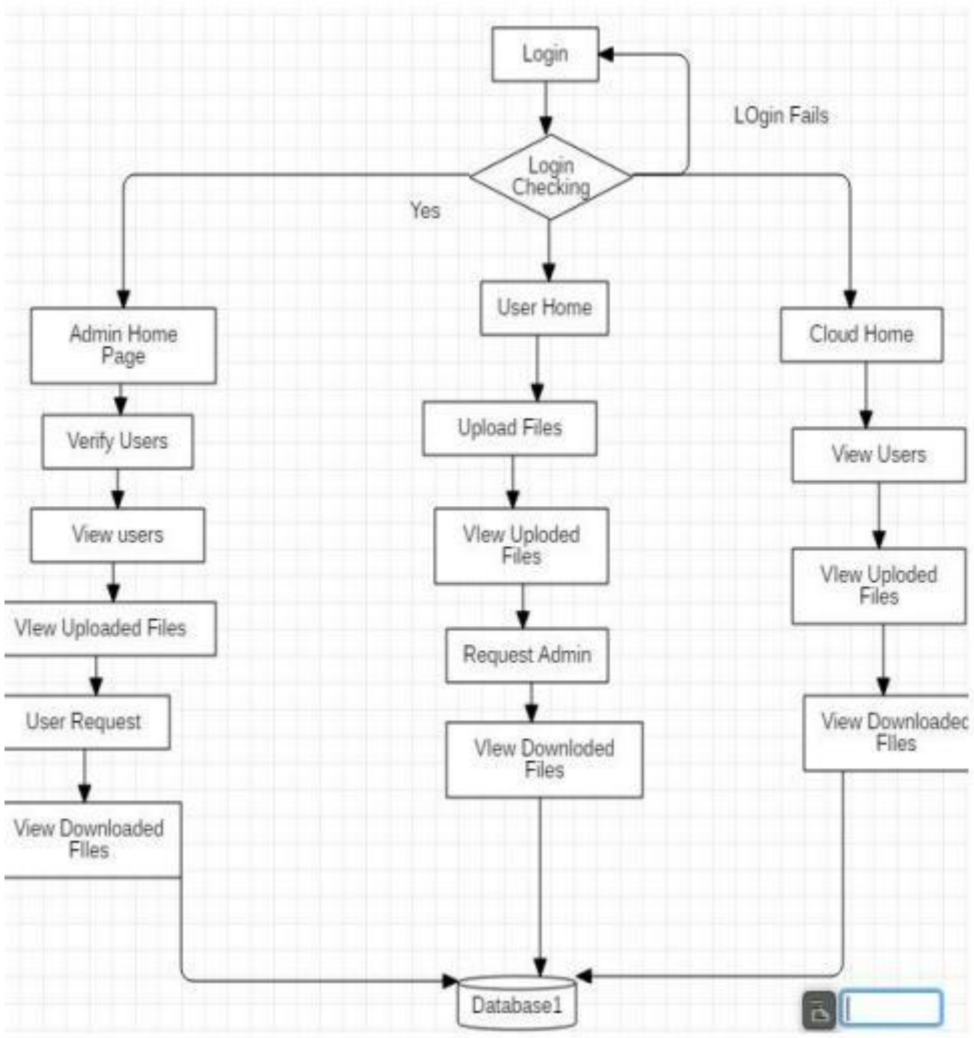


Fig 5.2 Data Flow Diagram

6.3 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).

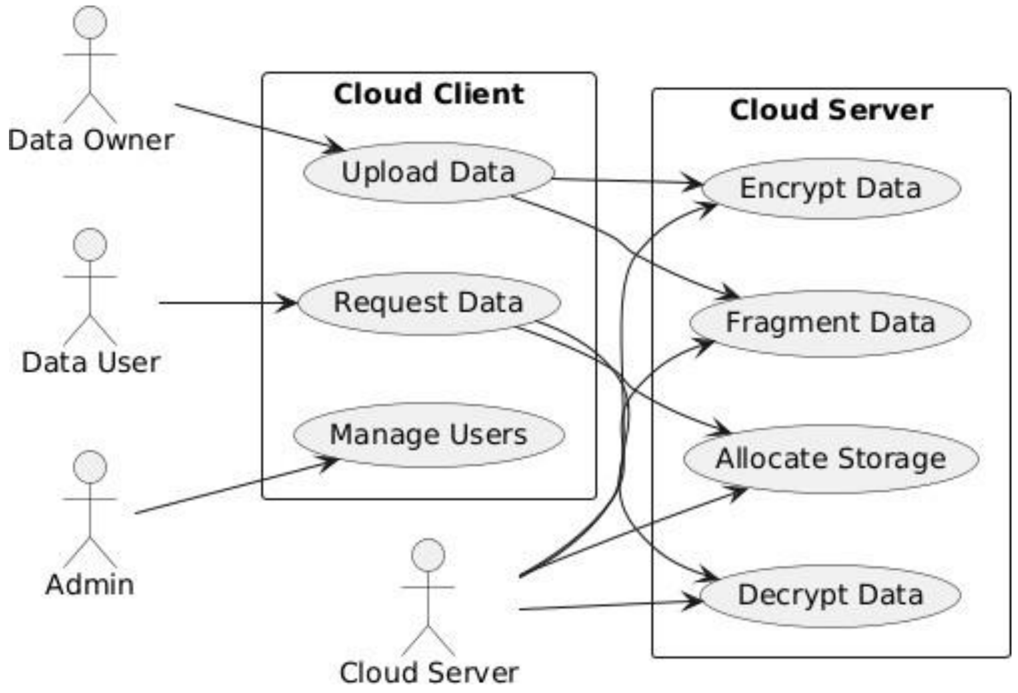


Fig 5.3 Use case Diagram

6.4 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.

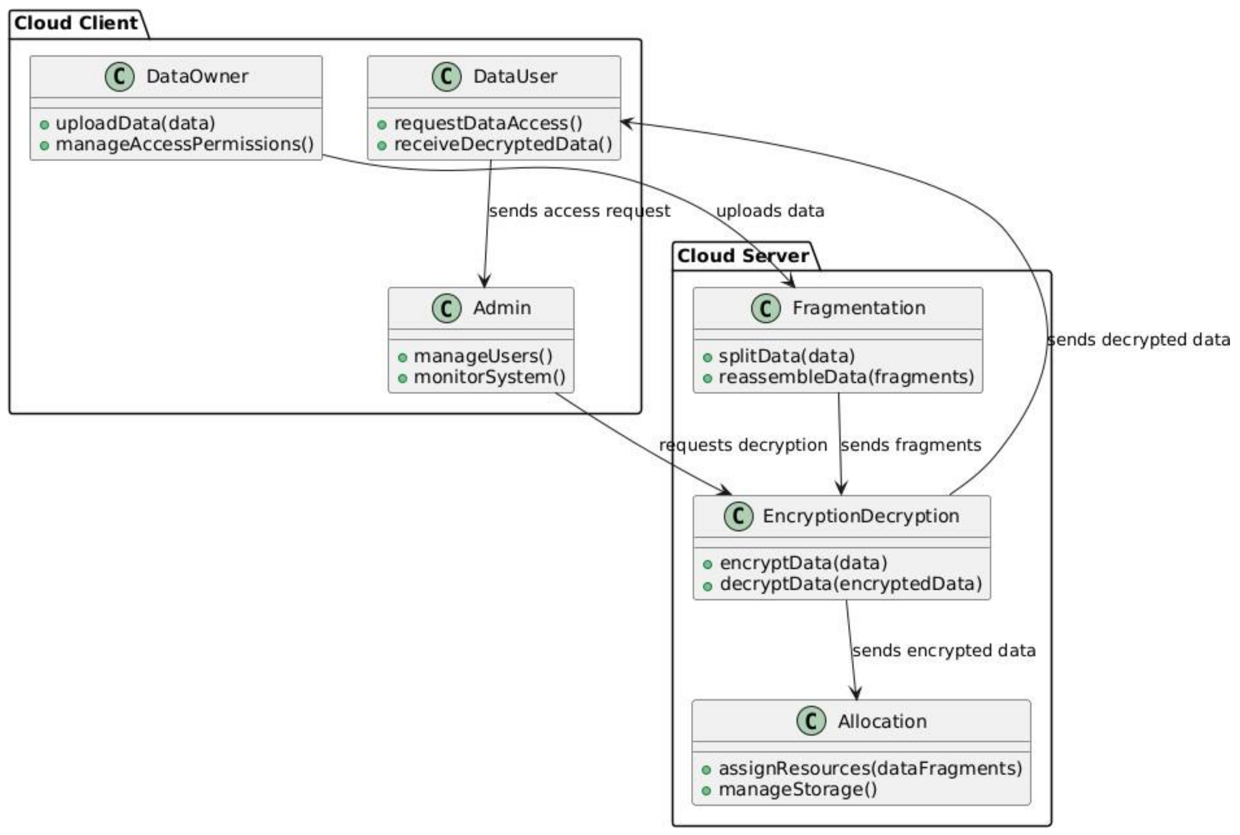


Fig 5.4 Class Diagram

6.5 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.

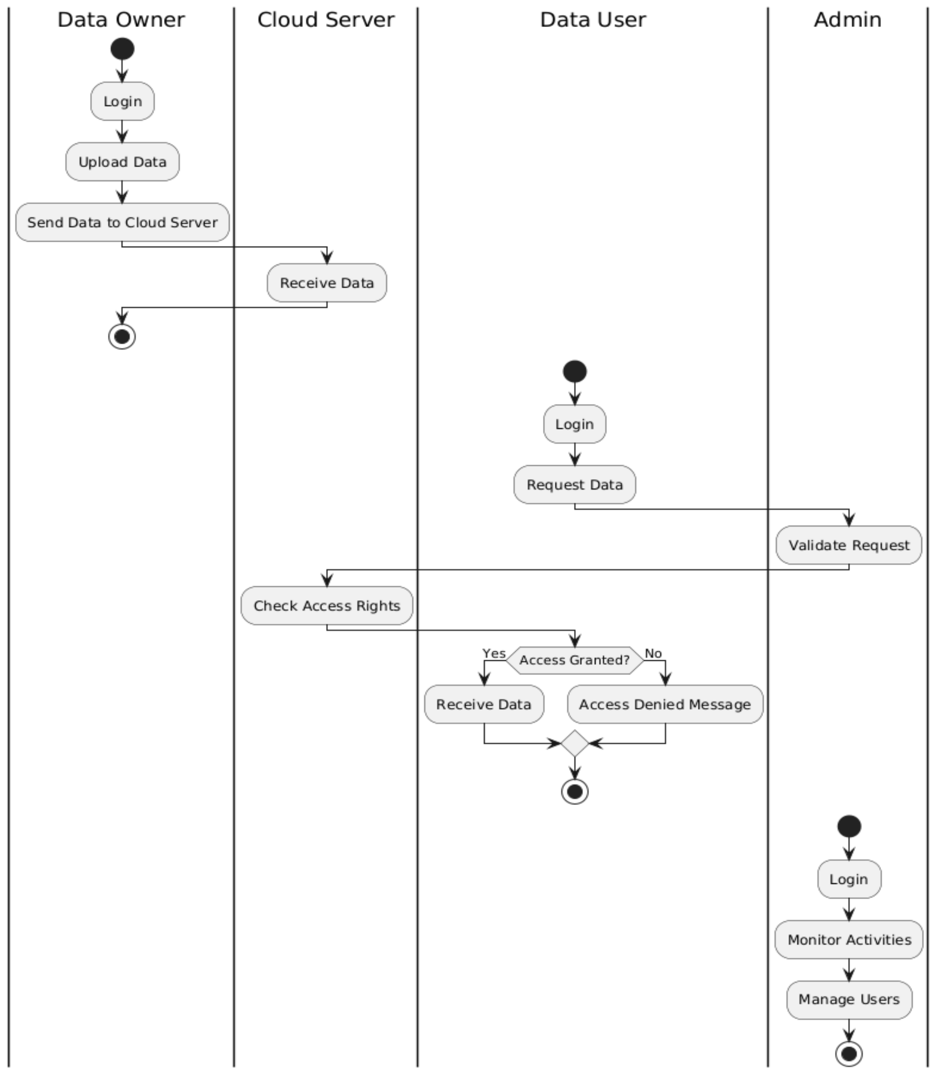


Fig 5.5 Activity Diagram

CHAPTER 7

SYSTEM TESTING AND IMPLEMENTATION

7.1 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:

7.1.1 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.

7.1.2 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.

7.1.3 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.

7.1.4 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.

7.1.5 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.

7.1.6 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.

7.1.7 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

7.2 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 and

effectively. 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.

CHAPTER 8

APPENDIX

8.1 Source Code

```
from flask importFlask,session, render_template,request,redirect, url_for
app = Flask(__name__)
app.secret_key = '23826b72637tskx86'
import sqlite3
import base64

@app.route('/logout')
deflogout():
session.clear()
returnredirect("/")

@app.route('/',methods = ['POST', 'GET'])
defindex():
ifrequest.method == 'POST':
email = request.form['email']
password = request.form['password']
conn = sqlite3.connect("database.db")
conn.row_factory = sqlite3.Row
cur = conn.cursor()
cur.execute("select * from users WHERE email=? AND password=?",(email,password))
rows = cur.fetchall()
iflen(rows) >0:
for row in rows:
session['id']=row['id']
session['name']=row['name']
session['user']='user'
```

```
return redirect("/home")
else:
returnredirect("/?err=Email or Password is Wrong!")
else:
return render_template('index.html')

@app.route('/signup',methods = ['POST', 'GET'])
defsignup():
ifrequest.method == 'POST':
name = request.form['name']
email = request.form['email']
password = request.form['password']
conn = sqlite3.connect("database.db")
conn.execute('INSERT INTO
users(name,email,password)VALUES(?,?,?)',(name,email,password))
conn.commit()
returnredirect("/?msg=Signup Successfully!")
else:
return render_template('signup.html')

@app.route('/home')
defHome():
if'user'in session and session['user']=='user':
userid=str(session['id'])
con = sqlite3.connect("database.db")
con.row_factory = sqlite3.Row
cur = con.cursor()
cur.execute("select * FROM files where user=?",(userid))
files = cur.fetchall()
return render_template('user/home.html',files=len(files),user=session['name'])
else:
```

```
returnredirect('/?err=Illegal Access')

@app.route('/files')
defFiles():
if'user'in session and session['user']=='user':
userid=str(session['id'])
con = sqlite3.connect("database.db")
con.row_factory = sqlite3.Row
cur = con.cursor()
cur.execute("select * FROM files where user=?",(userid))
files = cur.fetchall()
return render_template('user/files.html',files=files,user=session['name'])
else:
returnredirect('/?err=Illegal Access')

@app.route('/files/view')
defFilesView():
attid = request.args.get('id')
con = sqlite3.connect("database.db")
con.row_factory = sqlite3.Row
cur = con.cursor()
cur.execute("select * FROM files where id=?",(attid))
file = cur.fetchall()
return render_template('password.html',file=file[0])

@app.route('/files/open',methods = ['POST', 'GET'])
defFilesOpen():
ifrequest.method == 'POST':
fileid = request.form['id']
password = request.form['password']
conn = sqlite3.connect("database.db")
```

```
conn.row_factory = sqlite3.Row
cur = conn.cursor()
cur.execute("select * from files WHERE id=? AND password=?",(fileid,password))
file = cur.fetchall()
iflen(file) >0:
return render_template('file.html',type=file[0]['type'],file=file[0]['data'].decode())
else:
returnredirect("/files/view?err=File Password is Wrong!&id="+fileid)
else:
return render_template('index.html')

@app.route('/files/add',methods = ['POST', 'GET'])
defFileAdd():
ifrequest.method == 'POST':
user=str(session['id'])
title = request.form['title']
password = request.form['password']
file = request.files['file']
data = base64.b64encode(file.read())
conn = sqlite3.connect("database.db")
conn.execute('INSERT INTO
files(title,data,password,user,type)VALUES(?,?,?,?,?)',(title,data,password,user,file.content_type
))
conn.commit()
conn = sqlite3.connect("database1.db")
conn.execute('INSERT INTO
files(title,data,password,user,type)VALUES(?,?,?,?,?)',(title,data,password,user,file.content_type
))
conn.commit()
conn = sqlite3.connect("database2.db")
conn.execute('INSERT INTO
files(title,data,password,user,type)VALUES(?,?,?,?,?)',(title,data,password,user,file.content_type
))
conn.commit()
returnredirect("/files?msg=File Uploaded Successfully!")
else:
returnredirect("/err=Illegal Access")

@app.route('/files/delete')
defFilesDelete():
if'user'in session and session['user']=='user':
attid = request.args.get('id')
conn = sqlite3.connect('database.db')
conn.execute('DELETE FROM files WHERE id=?',(attid))
conn.commit()
conn = sqlite3.connect('database1.db')
conn.execute('DELETE FROM files WHERE id=?',(attid))
conn.commit()
conn = sqlite3.connect('database2.db')
conn.execute('DELETE FROM files WHERE id=?',(attid))
conn.commit()
returnredirect('/files?msg=File Deleted Successfully!')
else:
returnredirect('/?err=Illegal Access')

@app.route('/profile')
defProfile():
if'user'in session and session['user']=='user':
userid=str(session['id'])
con = sqlite3.connect("database.db")
con.row_factory = sqlite3.Row
cur = con.cursor()
```

```
cur.execute("select * from users WHERE id=?",(userid))
rows = cur.fetchall()
return render_template('user/profile.html',rows=rows[0],user=session['name'])
else:
returnredirect('/?err=Illegal Access')

@app.route('/profile/update',methods = ['POST', 'GET'])
defProfileUpdate():
ifrequest.method=='POST':
if'user'in session and session['user']=='user':
userid=str(session['id'])
oldPassword = request.form['oldPassword']
newPassword = request.form['newPassword']
conn = sqlite3.connect("database.db")
conn.row_factory = sqlite3.Row
cur = conn.cursor()
cur.execute("select * from users WHERE id=? AND password=?",(userid,oldPassword))
rows = cur.fetchall()
iflen(rows)>0:
conn.execute('UPDATE users SET password=? WHERE id=?',(newPassword,userid))
conn.commit()
conn.close()
returnredirect('/profile?msg=Password Changed Successfully!')
else:
returnredirect('/profile?msg=Old Password is Wrong!')
else:
returnredirect('/?err=Illegal Access')
else:
returnredirect('/?err=Illegal Access')


#
```

```
#
#
#
#
# Admin
@app.route('/admin',methods = ['POST', 'GET'])
defadmin():
ifrequest.method == 'POST':
email = request.form['email']
password = request.form['password']
if email=='admin@gmail.com'and password=='admin':
session['admin']='admin'
return redirect("/admin/home")
else:
returnredirect("/admin?err=Email or Password is Wrong!")
else:
return render_template('admin.html')

@app.route('/admin/home')
defadminHome():
if'admin'in session and session['admin']=='admin':
con = sqlite3.connect("database.db")
con.row_factory = sqlite3.Row
cur = con.cursor()
cur.execute("select * FROM users")
users = cur.fetchall()
cur.execute("select * FROM files")
files = cur.fetchall()
return render_template('admin/home.html',users=len(users),files=len(files))
else:
returnredirect('/admin?err=Illegal Access')
```

```
@app.route('/admin/users')
defadminUsers():
if'admin'in session and session['admin']=='admin':
con = sqlite3.connect("database.db")
con.row_factory = sqlite3.Row
cur = con.cursor()
cur.execute("select * from users")
users = cur.fetchall()
return render_template('admin/users.html',users=users)
else:
returnredirect('/admin?err=Illegal Access')

@app.route('/admin/users/delete')
defadminUsersDelete():
if'admin'in session and session['admin']=='admin':
attid = request.args.get('id')
conn = sqlite3.connect('database.db')
conn.execute('DELETE FROM users WHERE id=?',(attid))
conn.commit()
returnredirect('/admin/users?msg=Users Deleted Successfully!')
else:
returnredirect('/admin?err=Illegal Access')

@app.route('/admin/files')
defadminFiles():
if'admin'in session and session['admin']=='admin':
con = sqlite3.connect("database.db")
con.row_factory = sqlite3.Row
cur = con.cursor()
cur.execute("select * from files JOIN users ON users.id=files.user")
```

```
files = cur.fetchall()
return render_template('admin/files.html',files=files)
else:
returnredirect('/admin?err=Illegal Access')

@app.route('/admin/files/delete')
defadminFilesDelete():
if'admin'in session and session['admin']=='admin':
attid = request.args.get('id')
conn = sqlite3.connect('database.db')
conn.execute('DELETE FROM files WHERE id=?',(attid))
conn.commit()
conn = sqlite3.connect('database1.db')
conn.execute('DELETE FROM files WHERE id=?',(attid))
conn.commit()
conn = sqlite3.connect('database2.db')
conn.execute('DELETE FROM files WHERE id=?',(attid))
conn.commit()
returnredirect('/admin/files?msg=File Deleted Successfully!')
else:
returnredirect('/admin?err=Illegal Access')

# DB
@app.route('/db')
defdb():
conn = sqlite3.connect('database.db')
conn.execute('CREATE TABLE users(id INTEGER PRIMARY KEY
AUTOINCREMENT,name TEXT,email TEXT,password TEXT,created TIMESTAMP NOT
NULL DEFAULT CURRENT_TIMESTAMP)')
conn.execute('CREATE TABLE files(id INTEGER PRIMARY KEY AUTOINCREMENT,title
TEXT,data TEXT,type TEXT,password TEXT,user INTEGER,created TIMESTAMP NOT
NULL DEFAULT CURRENT_TIMESTAMP)')
conn.close()
conn = sqlite3.connect('database1.db')
conn.execute('CREATE TABLE files(id INTEGER PRIMARY KEY AUTOINCREMENT,title
TEXT,data TEXT,type TEXT,password TEXT,user INTEGER,created TIMESTAMP NOT
NULL DEFAULT CURRENT_TIMESTAMP)')
conn.close()
conn = sqlite3.connect('database2.db')
conn.execute('CREATE TABLE files(id INTEGER PRIMARY KEY AUTOINCREMENT,title
TEXT,data TEXT,type TEXT,password TEXT,user INTEGER,created TIMESTAMP NOT
NULL DEFAULT CURRENT_TIMESTAMP)')
conn.close()
return"Table Created"


if__name___== '__main__':
app.run(debug=True)
```

signup.html:

```
<!DOCTYPEhtml>

<!-- =========================================================
Sneat - Bootstrap 5 HTML Admin Template - Pro | v1.0.0
==============================================================


Product Page: https://themeselection.com/products/sneat-bootstrap-html-admin-template/
Created by: ThemeSelection
License: You must have a valid license purchased in order to legally use the theme for your
project.
Copyright ThemeSelection (https://themeselection.com)

=========================================================
-->
<!-- beautify ignore:start -->
<html
lang="en"
class="light-style customizer-hide"
dir="ltr"
data-theme="theme-default"
data-assets-path="/static/"
data-template="vertical-menu-template-free"
>
<head>
<metacharset="utf-8"/>
<meta
name="viewport"
content="width=device-width, initial-scale=1.0, user-scalable=no, minimum-scale=1.0,
maximum-scale=1.0"
/>

<title>User</title>

<metaname="description"content=""/>

<!-- Favicon -->
<linkrel="icon"type="image/x-icon"href="/static/img/favicon/favicon.ico"/>

<!-- Fonts -->
<linkrel="preconnect"href="https://fonts.googleapis.com"/>
<linkrel="preconnect"href="https://fonts.gstatic.com"crossorigin/>
<link
href="https://fonts.googleapis.com/css2?family=Public+Sans:ital,wght@0,300;0,400;0,500;0,60
0;0,700;1,300;1,400;1,500;1,600;1,700&display=swap"
```

```
rel="stylesheet"
/>

<!-- Icons. Uncomment required icon fonts -->
<linkrel="stylesheet"href="/static/vendor/fonts/boxicons.css"/>

<!-- Core CSS -->
<linkrel="stylesheet"href="/static/vendor/css/core.css"class="template-customizer-core-css"/>
<linkrel="stylesheet"href="/static/vendor/css/theme-default.css"class="template-customizer-
theme-css"/>
<linkrel="stylesheet"href="/static/css/demo.css"/>

<!-- Vendors CSS -->
<linkrel="stylesheet"href="/static/vendor/libs/perfect-scrollbar/perfect-scrollbar.css"/>

<!-- Page CSS -->
<!-- Page -->
<linkrel="stylesheet"href="/static/vendor/css/pages/page-auth.css"/>
<!-- Helpers -->
<scriptsrc="/static/vendor/js/helpers.js"></script>

<!--! Template customizer & Theme config files MUST be included after core stylesheets and
helpers.js in the <head> section -->
<!--? Config:  Mandatory theme config file contain global vars & default theme options, Set your
preferred theme option in this file.  -->
<scriptsrc="/static/js/config.js"></script>
</head>

<body>
<!-- Content -->
```

```
<divclass="container-xxl">
<divclass="authentication-wrapper authentication-basic container-p-y">
<divclass="authentication-inner">
<!-- Register -->
<divclass="card">
<divclass="card-body">
<!-- /Logo -->
<h4class="mt-3">User Signup!</h4>

<formid="formAuthentication"class="mb-3"action="/signup"method="POST">
<divclass="mb-3">
<labelfor="email"class="form-label">Name</label>
<input
type="text"
class="form-control"
id="name"
required
name="name"
placeholder="Enter your Name"
autofocus
/>
</div>
<divclass="mb-3">
<labelfor="email"class="form-label">Email</label>
<input
type="email"
class="form-control"
id="email"
required
name="email"
placeholder="Enter your email"
```

```
autofocus
/>
</div>
<divclass="mb-3 form-password-toggle">
<divclass="d-flex justify-content-between">
<labelclass="form-label"for="password">Password</label>
</div>
<divclass="input-group input-group-merge">
<input
type="password"
id="password"
class="form-control"
name="password"
required
placeholder="&#xb7;&#xb7;&#xb7;&#xb7;&#xb7;&#xb7;&#xb7;&#xb7;&#xb7;&#xb7;&#xb7
;&#xb7;"
aria-describedby="password"
/>
<spanclass="input-group-text cursor-pointer"><iclass="bx bx-hide"></i></span>
</div>
</div>
<divclass="mb-3">
<buttonclass="btn btn-primary d-grid w-100"type="submit">Sign Up</button>
</div>
</form>
</div>
<divclass="text-end p-3">
<ahref="/">Login</a>
</div>
</div>
<!-- /Register -->
```

```
</div>
</div>
</div>
<script>
constqueryString = window.location.search;
consturlParams = newURLSearchParams(queryString);
if(urlParams.get('err')){
document.write("<div id='err' style='position:fixed;bottom:30px; right:30px;background-
color:tomato;padding:10px;border-radius:10px;box-shadow:2px 2px 4px #aaa;color:white;font-
weight:600'>"+urlParams.get('err')+"</div>")
setTimeout(()=>{
document.getElementById("err").style.display="none"
}, 5000)
}
if(urlParams.get('msg')){
document.write("<div id='msg' style='position:fixed;bottom:30px; right:30px;background-
color:green;padding:10px;border-radius:10px;box-shadow:2px 2px 4px #aaa;color:white;font-
weight:600'>"+urlParams.get('msg')+"</div>")
setTimeout(()=>{
document.getElementById("msg").style.display="none"
}, 5000)
}
</script>

<!-- Core JS -->
<!-- build:js assets/vendor/js/core.js -->
<scriptsrc="/static/vendor/libs/jquery/jquery.js"></script>
<scriptsrc="/static/vendor/libs/popper/popper.js"></script>
<scriptsrc="/static/vendor/js/bootstrap.js"></script>
<scriptsrc="/static/vendor/libs/perfect-scrollbar/perfect-scrollbar.js"></script>
```

```
<scriptsrc="/static/vendor/js/menu.js"></script>
<!-- endbuild -->

<!-- Vendors JS -->

<!-- Main JS -->
<scriptsrc="/static/js/main.js"></script>

<scriptasyncdefersrc="https://buttons.github.io/buttons.js"></script>
</body>
</html>
```

Password.html:

```
<!DOCTYPEhtml>

<!-- =========================================================
Sneat - Bootstrap 5 HTML Admin Template - Pro | v1.0.0
==============================================================

Product Page: https://themeselection.com/products/sneat-bootstrap-html-admin-template/
Created by: ThemeSelection
License: You must have a valid license purchased in order to legally use the theme for your
project.
Copyright ThemeSelection (https://themeselection.com)

=========================================================
-->
<!-- beautify ignore:start -->
<html
lang="en"
class="light-style customizer-hide"
```

```
dir="ltr"
data-theme="theme-default"
data-assets-path="/static/"
data-template="vertical-menu-template-free"
>
<head>
<metacharset="utf-8"/>
<meta
name="viewport"
content="width=device-width, initial-scale=1.0, user-scalable=no, minimum-scale=1.0,
maximum-scale=1.0"
/>

<title>Admin</title>

<metaname="description"content=""/>

<!-- Favicon -->
<linkrel="icon"type="image/x-icon"href="/static/img/favicon/favicon.ico"/>

<!-- Fonts -->
<linkrel="preconnect"href="https://fonts.googleapis.com"/>
<linkrel="preconnect"href="https://fonts.gstatic.com"crossorigin/>
<link
href="https://fonts.googleapis.com/css2?family=Public+Sans:ital,wght@0,300;0,400;0,500;0,60
0;0,700;1,300;1,400;1,500;1,600;1,700&display=swap"
rel="stylesheet"
/>

<!-- Icons. Uncomment required icon fonts -->
<linkrel="stylesheet"href="/static/vendor/fonts/boxicons.css"/>
```

```
<!-- Core CSS -->
<linkrel="stylesheet"href="/static/vendor/css/core.css"class="template-customizer-core-css"/>
<linkrel="stylesheet"href="/static/vendor/css/theme-default.css"class="template-customizer-
theme-css"/>
<linkrel="stylesheet"href="/static/css/demo.css"/>

<!-- Vendors CSS -->
<linkrel="stylesheet"href="/static/vendor/libs/perfect-scrollbar/perfect-scrollbar.css"/>

<!-- Page CSS -->
<!-- Page -->
<linkrel="stylesheet"href="/static/vendor/css/pages/page-auth.css"/>
<!-- Helpers -->
<scriptsrc="/static/vendor/js/helpers.js"></script>

<!--! Template customizer & Theme config files MUST be included after core stylesheets and
helpers.js in the <head> section -->
<!--? Config:  Mandatory theme config file contain global vars & default theme options, Set your
preferred theme option in this file.  -->
<scriptsrc="/static/js/config.js"></script>
</head>

<body>
<!-- Content -->

<divclass="container-xxl">
<divclass="authentication-wrapper authentication-basic container-p-y">
<divclass="authentication-inner">
<!-- Register -->
<divclass="card">
```

```
<divclass="card-body">
<!-- /Logo -->
<h4class="mt-3">Enter Password to Open {{file['title']}}</h4>

<formid="formAuthentication"class="mb-3"action="/files/open"method="POST">
<inputtype="hidden"name="id"value="{{file['id']}}">
<divclass="mb-3 form-password-toggle">
<divclass="d-flex justify-content-between">
<labelclass="form-label"for="password">Password</label>
</div>
<divclass="input-group input-group-merge">
<input
type="password"
id="password"
class="form-control"
name="password"
required
placeholder="&#xb7;&#xb7;&#xb7;&#xb7;&#xb7;&#xb7;&#xb7;&#xb7;&#xb7;&#xb7;&#xb7
;&#xb7;"
aria-describedby="password"
/>
<spanclass="input-group-text cursor-pointer"><iclass="bx bx-hide"></i></span>
</div>
</div>
<divclass="mb-3">
<buttonclass="btn btn-primary d-grid w-100"type="submit">Open File</button>
</div>
</form>
</div>
</div>
<!-- /Register -->
```

```
</div>
</div>
</div>
<script>
constqueryString = window.location.search;
consturlParams = newURLSearchParams(queryString);
if(urlParams.get('err')){
document.write("<div id='err' style='position:fixed;bottom:30px; right:30px;background-
color:tomato;padding:10px;border-radius:10px;box-shadow:2px 2px 4px #aaa;color:white;font-
weight:600'>"+urlParams.get('err')+"</div>")
setTimeout(()=>{
document.getElementById("err").style.display="none"
}, 5000)
}
if(urlParams.get('msg')){
document.write("<div id='msg' style='position:fixed;bottom:30px; right:30px;background-
color:green;padding:10px;border-radius:10px;box-shadow:2px 2px 4px #aaa;color:white;font-
weight:600'>"+urlParams.get('msg')+"</div>")
setTimeout(()=>{
document.getElementById("msg").style.display="none"
}, 5000)
}
</script>
<!-- Core JS -->
<!-- build:js assets/vendor/js/core.js -->
<scriptsrc="/static/vendor/libs/jquery/jquery.js"></script>
<scriptsrc="/static/vendor/libs/popper/popper.js"></script>
<scriptsrc="/static/vendor/js/bootstrap.js"></script>
<scriptsrc="/static/vendor/libs/perfect-scrollbar/perfect-scrollbar.js"></script>

<scriptsrc="/static/vendor/js/menu.js"></script>
```

```
<!-- endbuild -->

<!-- Vendors JS -->

<!-- Main JS -->
<scriptsrc="/static/js/main.js"></script>

<scriptasyncdefersrc="https://buttons.github.io/buttons.js"></script>
</body>
</html>
```

Index.html:

```
<!DOCTYPEhtml>

<!-- =========================================================
Sneat - Bootstrap 5 HTML Admin Template - Pro | v1.0.0
==============================================================

Product Page: https://themeselection.com/products/sneat-bootstrap-html-admin-template/
Created by: ThemeSelection
License: You must have a valid license purchased in order to legally use the theme for your
project.
Copyright ThemeSelection (https://themeselection.com)

=========================================================
-->
<!-- beautify ignore:start -->
<html
lang="en"
class="light-style customizer-hide"
```

```
dir="ltr"
data-theme="theme-default"
data-assets-path="/static/"
data-template="vertical-menu-template-free"
>
<head>
<metacharset="utf-8"/>
<meta
name="viewport"
content="width=device-width, initial-scale=1.0, user-scalable=no, minimum-scale=1.0,
maximum-scale=1.0"
/>

<title>User</title>

<metaname="description"content=""/>

<!-- Favicon -->
<linkrel="icon"type="image/x-icon"href="/static/img/favicon/favicon.ico"/>

<!-- Fonts -->
<linkrel="preconnect"href="https://fonts.googleapis.com"/>
<linkrel="preconnect"href="https://fonts.gstatic.com"crossorigin/>
<link
href="https://fonts.googleapis.com/css2?family=Public+Sans:ital,wght@0,300;0,400;0,500;0,60
0;0,700;1,300;1,400;1,500;1,600;1,700&display=swap"
rel="stylesheet"
/>

<!-- Icons. Uncomment required icon fonts -->
<linkrel="stylesheet"href="/static/vendor/fonts/boxicons.css"/>
```

```
<!-- Core CSS -->
<linkrel="stylesheet"href="/static/vendor/css/core.css"class="template-customizer-core-css"/>
<linkrel="stylesheet"href="/static/vendor/css/theme-default.css"class="template-customizer-
theme-css"/>
<linkrel="stylesheet"href="/static/css/demo.css"/>

<!-- Vendors CSS -->
<linkrel="stylesheet"href="/static/vendor/libs/perfect-scrollbar/perfect-scrollbar.css"/>

<!-- Page CSS -->
<!-- Page -->
<linkrel="stylesheet"href="/static/vendor/css/pages/page-auth.css"/>
<!-- Helpers -->
<scriptsrc="/static/vendor/js/helpers.js"></script>

<!--! Template customizer & Theme config files MUST be included after core stylesheets and
helpers.js in the <head> section -->
<!--? Config:  Mandatory theme config file contain global vars & default theme options, Set your
preferred theme option in this file.  -->
<scriptsrc="/static/js/config.js"></script>
</head>

<body>
<!-- Content -->

<divclass="container-xxl">
<divclass="authentication-wrapper authentication-basic container-p-y">
<divclass="authentication-inner">
<!-- Register -->
<divclass="card">
```

```
<divclass="card-body">
<!-- /Logo -->
<h4class="mt-3">User Login!</h4>

<formid="formAuthentication"class="mb-3"action="/"method="POST">
<divclass="mb-3">
<labelfor="email"class="form-label">Email</label>
<input
type="email"
class="form-control"
id="email"
required
name="email"
placeholder="Enter your email"
autofocus
/>
</div>
<divclass="mb-3 form-password-toggle">
<divclass="d-flex justify-content-between">
<labelclass="form-label"for="password">Password</label>
</div>
<divclass="input-group input-group-merge">
<input
type="password"
id="password"
class="form-control"
name="password"
required
placeholder="&#xb7;&#xb7;&#xb7;&#xb7;&#xb7;&#xb7;&#xb7;&#xb7;&#xb7;&#xb7;&#xb7
;&#xb7;"
aria-describedby="password"
```

```
/>
<spanclass="input-group-text cursor-pointer"><iclass="bx bx-hide"></i></span>
</div>
</div>
<divclass="mb-3">
<buttonclass="btn btn-primary d-grid w-100"type="submit">Sign in</button>
</div>
</form>
</div>
<divclass="text-end p-3">
<ahref="/signup">Signup</a>
</div>
</div>
<!-- /Register -->
</div>
</div>
</div>
<script>
constqueryString = window.location.search;
consturlParams = newURLSearchParams(queryString);
if(urlParams.get('err')){
document.write("<div id='err' style='position:fixed;bottom:30px; right:30px;background-
color:tomato;padding:10px;border-radius:10px;box-shadow:2px 2px 4px #aaa;color:white;font-
weight:600'>"+urlParams.get('err')+"</div>")
setTimeout(()=>{
document.getElementById("err").style.display="none"
}, 5000)
}
if(urlParams.get('msg')){
document.write("<div id='msg' style='position:fixed;bottom:30px; right:30px;background-
color:green;padding:10px;border-radius:10px;box-shadow:2px 2px 4px #aaa;color:white;font-
weight:600'>"+urlParams.get('msg')+"</div>")
setTimeout(()=>{
document.getElementById("msg").style.display="none"
}, 5000)
}
</script>

<!-- Core JS -->
<!-- build:js assets/vendor/js/core.js -->
<scriptsrc="/static/vendor/libs/jquery/jquery.js"></script>
<scriptsrc="/static/vendor/libs/popper/popper.js"></script>
<scriptsrc="/static/vendor/js/bootstrap.js"></script>
<scriptsrc="/static/vendor/libs/perfect-scrollbar/perfect-scrollbar.js"></script>

<scriptsrc="/static/vendor/js/menu.js"></script>
<!-- endbuild -->

<!-- Vendors JS -->

<!-- Main JS -->
<scriptsrc="/static/js/main.js"></script>

<scriptasyncdefersrc="https://buttons.github.io/buttons.js"></script>
</body>
</html>
```

File.html:

```
<!DOCTYPEhtml>
<htmllang="en">
<head>
```

```
<metacharset="UTF-8">
<metaname="viewport"content="width=device-width, initial-scale=1.0">
<title>Secure File View</title>
</head>
<bodystyle="margin:0px;padding:0px">
<style>html,body{height:100svh}</style>
<iframesrc="data:{{type}};base64,{{file}}"style="width:
100%;height:100%"frameborder="0"></iframe>
</body>
</html>
```

Admin.html:

```
<!DOCTYPEhtml>

<!-- =========================================================
Sneat - Bootstrap 5 HTML Admin Template - Pro | v1.0.0
==============================================================

Product Page: https://themeselection.com/products/sneat-bootstrap-html-admin-template/
Created by: ThemeSelection
License: You must have a valid license purchased in order to legally use the theme for your
project.
Copyright ThemeSelection (https://themeselection.com)

=========================================================
-->
<!-- beautify ignore:start -->
<html
lang="en"
class="light-style customizer-hide"
```

```
}, 5000)
}
</script>
<!-- Core JS -->
<!-- build:js assets/vendor/js/core.js -->
<scriptsrc="/static/vendor/libs/jquery/jquery.js"></script>
<scriptsrc="/static/vendor/libs/popper/popper.js"></script>
<scriptsrc="/static/vendor/js/bootstrap.js"></script>
<scriptsrc="/static/vendor/libs/perfect-scrollbar/perfect-scrollbar.js"></script>

<scriptsrc="/static/vendor/js/menu.js"></script>
<!-- endbuild -->

<!-- Vendors JS -->

<!-- Main JS -->
<scriptsrc="/static/js/main.js"></script>

<scriptasyncdefersrc="https://buttons.github.io/buttons.js"></script>
</body>
</html>
```

User/files.html:

```
<!DOCTYPEhtml>

<!-- =========================================================
Sneat - Bootstrap 5 HTML Admin Template - Pro | v1.0.0
==============================================================

Product Page: https://themeselection.com/products/sneat-bootstrap-html-admin-template/
```

```
Created by: ThemeSelection
License: You must have a valid license purchased in order to legally use the theme for your
project.
Copyright ThemeSelection (https://themeselection.com)

=================================================================
-->
<!-- beautify ignore:start -->
<html
lang="en"
class="light-style layout-menu-fixed"
dir="ltr"
data-theme="theme-default"
data-assets-path="/static/"
data-template="vertical-menu-template-free"
>
<head>
<metacharset="utf-8"/>
<meta
name="viewport"
content="width=device-width, initial-scale=1.0, user-scalable=no, minimum-scale=1.0,
maximum-scale=1.0"
/>

<title>User</title>
<scriptsrc="https://cdn.jsdelivr.net/npm/apexcharts"></script>

<metaname="description"content=""/>

<!-- Favicon -->
<linkrel="icon"type="image/x-icon"href="/static/img/favicon/favicon.ico"/>
```

```
<!-- Fonts -->
<linkrel="preconnect"href="https://fonts.googleapis.com"/>
<linkrel="preconnect"href="https://fonts.gstatic.com"crossorigin/>
<link
href="https://fonts.googleapis.com/css2?family=Public+Sans:ital,wght@0,300;0,400;0,500;0,60
0;0,700;1,300;1,400;1,500;1,600;1,700&display=swap"
rel="stylesheet"
/>

<!-- Icons. Uncomment required icon fonts -->
<linkrel="stylesheet"href="/static/vendor/fonts/boxicons.css"/>

<!-- Core CSS -->
<linkrel="stylesheet"href="/static/vendor/css/core.css"class="template-customizer-core-css"/>
<linkrel="stylesheet"href="/static/vendor/css/theme-default.css"class="template-customizer-
theme-css"/>
<linkrel="stylesheet"href="/static/css/demo.css"/>

<!-- Vendors CSS -->
<linkrel="stylesheet"href="/static/vendor/libs/perfect-scrollbar/perfect-scrollbar.css"/>

<!-- Page CSS -->

<!-- Helpers -->
<scriptsrc="/static/vendor/js/helpers.js"></script>

<!--! Template customizer & Theme config files MUST be included after core stylesheets and
helpers.js in the <head> section -->
<!--? Config:  Mandatory theme config file contain global vars & default theme options, Set your
preferred theme option in this file.  -->
```

```
<scriptsrc="/static/js/config.js"></script>
</head>
<script>
constqueryString = window.location.search;
consturlParams = newURLSearchParams(queryString);
if(urlParams.get('err')){
document.write("<div id='err' style='position:fixed;bottom:30px; right:30px;background-
color:tomato;padding:10px;border-radius:10px;box-shadow:2px 2px 4px #aaa;color:white;font-
weight:600'>"+urlParams.get('err')+"</div>")
setTimeout(()=>{
document.getElementById("err").style.display="none"
}, 5000)
}
if(urlParams.get('msg')){
document.write("<div id='msg' style='position:fixed;bottom:30px; right:30px;background-
color:green;padding:10px;border-radius:10px;box-shadow:2px 2px 4px #aaa;color:white;font-
weight:600'>"+urlParams.get('msg')+"</div>")
setTimeout(()=>{
document.getElementById("msg").style.display="none"
}, 5000)
}
</script>
<body>
<!-- Layout wrapper -->
<divclass="layout-wrapper layout-content-navbar">
<divclass="layout-container">
<!-- Menu -->

<asideid="layout-menu"class="layout-menu menu-vertical menu bg-menu-theme">
<divclass="app-brand demo">
<ahref=""class="app-brand-link ">
```

```
<spanclass="h3 text-primary demo menu-text fw-bolder ms-2">User</span>
</a>

<ahref="javascript:void(0);"class="layout-menu-toggle menu-link text-large ms-auto d-block d-
xl-none">
<iclass="bx bx-chevron-left bx-sm align-middle"></i>
</a>
</div>

<divclass="menu-inner-shadow"></div>

<ulclass="menu-inner py-1">
<!-- Dashboard -->
<liclass="menu-item ">
<ahref="/home"class="menu-link">
<iclass="menu-icon tf-icons bx bx-home-circle"></i>
<divdata-i18n="Analytics">Dashboard</div>
</a>
</li>
<liclass="menu-item active">
<ahref="/files"class="menu-link">
<iclass="menu-icon tf-icons bx bx-detail"></i>
<divdata-i18n="Analytics">Files</div>
</a>
</li>
<liclass="menu-item">
<ahref="/profile"class="menu-link">
<iclass="menu-icon tf-icons bx bx-user"></i>
<divdata-i18n="Analytics">Profile</div>
</a>
</li>
```

```
</ul>
</aside>
<!-- Layout container -->
<divclass="layout-page">
<!-- Navbar -->

<nav
class="layout-navbar container-fluid navbar navbar-expand-xl navbar-detached align-items-
center bg-navbar-theme"
id="layout-navbar"
>

<divclass="navbar-nav-right d-flex align-items-center"id="navbar-collapse">
<!-- Search -->
<spanclass="fw-bolder display-6">Dashboard</span>
<!-- /Search -->

<ulclass="navbar-nav flex-row align-items-center ms-auto">

<!-- User -->
<liclass="nav-item navbar-dropdown dropdown-user dropdown">
<aclass="nav-link dropdown-toggle hide-arrow"href="javascript:void(0);"data-bs-
toggle="dropdown">
<divclass="avatar avatar-online">
<imgsrc="/static/img/avatars/1.png"altclass="w-px-40 h-auto rounded-circle"/>
</div>
</a>
<ulclass="dropdown-menu dropdown-menu-end">
<li>
<aclass="dropdown-item"href="#">
<divclass="d-flex">
```

```
<divclass="flex-shrink-0 me-3">
<divclass="avatar avatar-online">
<imgsrc="/static/img/avatars/1.png"altclass="w-px-40 h-auto rounded-circle"/>
</div>
</div>
<divclass="flex-grow-1">
<spanclass="fw-semibold d-block">{{user}}</span>
<!--<small class="text-muted">Admin</small> -->
</div>
</div>
</a>
</li>
<!--<li>
<div class="dropdown-divider"></div>
</li> -->
<!--<li>
<a class="dropdown-item" href="#">
<i class="bx bx-user me-2"></i>
<span class="align-middle">My Profile</span>
</a>
</li> -->
<li>
<divclass="dropdown-divider"></div>
</li>
<li>
<aclass="dropdown-item"href="/logout">
<iclass="bx bx-power-off me-2"></i>
<spanclass="align-middle">Log Out</span>
</a>
</li>
</ul>
```

```
</li>
<!--/ User -->
</ul>
</div>
</nav>


<div
class="modal fade"
id="modalToggle"
aria-labelledby="modalToggleLabel"
tabindex="-1"
style="display: none"
aria-hidden="true"
>
<divclass="modal-dialog modal-dialog-centered">
<divclass="modal-content">
<divclass="modal-header">
<h5class="modal-title"id="modalToggleLabel">Upload Files</h5>
<button
type="button"
class="btn-close"
data-bs-dismiss="modal"
aria-label="Close"
></button>
</div>
<divclass="modal-body">
<formaction="/files/add"method="post"enctype = "multipart/form-data">
<divclass="mb-3">
<labelfor="email"class="form-label">File Title :</label>
<input
type="text"
```

```
class="form-control"
id="email"
name="title"
placeholder="File Title ..."
autofocus
required
/>
</div>
<divclass="mb-3">
<labelfor="email"class="form-label">File :</label>
<input
type="file"
class="form-control"
id="email"
name="file"
placeholder="Enter Worked Hours"
autofocus
required
/>
</div>
<divclass="mb-3">
<labelfor="email"class="form-label">File Password :</label>
<input
type="password"
class="form-control"
id="email"
name="password"
placeholder="File Password ..."
autofocus
required
/>
```

```
<divclass="container-fluid flex-grow-1 container-p-y">
<!-- Layout Demo -->
<divclass="row">
<divclass="col-4">
<divclass="card p-3">
<h4>Total Files</h4>
<h2>{{files}}</h2>
</div>
</div>
</div>
<!--/ Layout Demo -->
</div>
<!-- / Content -->
<!-- / Footer -->

<divclass="content-backdrop fade"></div>
</div>
<!-- Content wrapper -->
</div>
<!-- / Layout page -->
</div>

<!-- Overlay -->
<divclass="layout-overlay layout-menu-toggle"></div>
</div>
<!-- / Layout wrapper -->
<!-- Core JS -->
<!-- build:js assets/vendor/js/core.js -->
<scriptsrc="/static/vendor/libs/jquery/jquery.js"></script>
<scriptsrc="/static/vendor/libs/popper/popper.js"></script>
```

```
<scriptsrc="/static/vendor/js/bootstrap.js"></script>
<scriptsrc="/static/vendor/libs/perfect-scrollbar/perfect-scrollbar.js"></script>

<scriptsrc="/static/vendor/js/menu.js"></script>
<!-- endbuild -->

<!-- Vendors JS -->

<!-- Main JS -->
<scriptsrc="/static/js/main.js"></script>

<!-- Page JS -->

<!-- Place this tag in your head or just before your close body tag. -->
<scriptasyncdefersrc="https://buttons.github.io/buttons.js"></script>
</body>
</html>
```

User/profile.html:

```
<!DOCTYPEhtml>

<!-- =========================================================
Sneat - Bootstrap 5 HTML Admin Template - Pro | v1.0.0
==============================================================

Product Page: https://themeselection.com/products/sneat-bootstrap-html-admin-template/
Created by: ThemeSelection
License: You must have a valid license purchased in order to legally use the theme for your
project.
Copyright ThemeSelection (https://themeselection.com)
```

```
============================================================
-->
<!-- beautify ignore:start -->
<html
lang="en"
class="light-style layout-menu-fixed"
dir="ltr"
data-theme="theme-default"
data-assets-path="/static/"
data-template="vertical-menu-template-free"
>
<head>
<metacharset="utf-8"/>
<meta
name="viewport"
content="width=device-width, initial-scale=1.0, user-scalable=no, minimum-scale=1.0,
maximum-scale=1.0"
/>

<title>User</title>

<metaname="description"content=""/>

<!-- Favicon -->
<linkrel="icon"type="image/x-icon"href="/static/img/favicon/favicon.ico"/>

<!-- Fonts -->
<linkrel="preconnect"href="https://fonts.googleapis.com"/>
<linkrel="preconnect"href="https://fonts.gstatic.com"crossorigin/>
<link
```

```
<divclass="modal-body">
<formaction="/admin/manager/add"method="post">
<divclass="mb-3">
<labelfor="email"class="form-label">Staff</label>
<selectname="user"id=""class="form-select">
<optionvalue=""selecteddisabledhidden>Select Staff</option>
{% for user in users %}
<optionvalue="{{user['id']}}">{{user['name']}}</option>
{% endfor%}
</select>
</div>
<divclass="mb-3">
<labelfor="email"class="form-label">Attendance</label>
<selectname="attendance"class="form-select"id="">
<optionvalue=""selecteddisabledhidden>Select Attendance</option>
<optionvalue="Present">Present</option>
<optionvalue="Absent">Absent</option>
</select>
</div>
<divclass="mb-3">
<labelfor="email"class="form-label">Worked Hours</label>
<input
type="number"
class="form-control"
id="email"
name="hour"
placeholder="Enter Worked Hours"
autofocus
required
/>
</div>
```

```
<divclass="mb-3">
<labelfor="email"class="form-label">Date</label>
<input
type="date"
class="form-control"
id="email"
name="date"
placeholder="Enter date"
autofocus
required
/>
</div>
<divclass="text-end">
<buttonclass="btn btn-primary">Add</button>
</div>
</form>
</div>
</div>
</div>
</div>
<!-- / Layout wrapper -->
<!-- Core JS -->
<!-- build:js assets/vendor/js/core.js -->
<scriptsrc="/static/vendor/libs/jquery/jquery.js"></script>
<scriptsrc="/static/vendor/libs/popper/popper.js"></script>
<scriptsrc="/static/vendor/js/bootstrap.js"></script>
<scriptsrc="/static/vendor/libs/perfect-scrollbar/perfect-scrollbar.js"></script>

<scriptsrc="/static/vendor/js/menu.js"></script>
<!-- endbuild -->
```

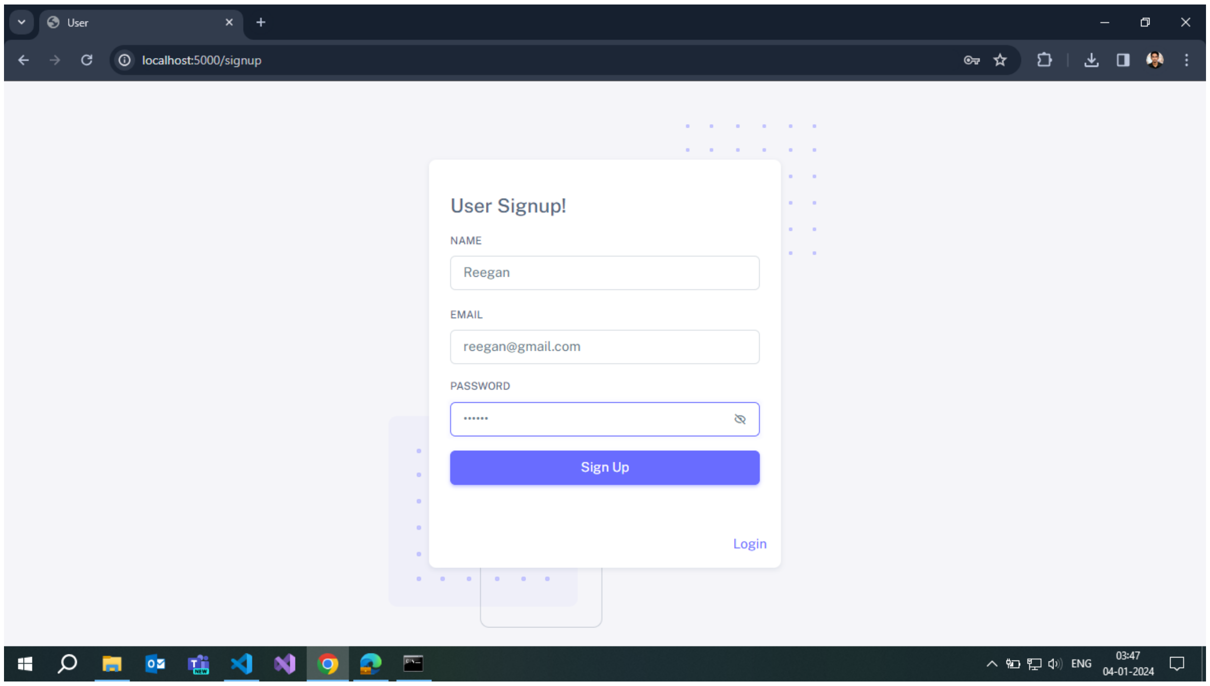
```
<!-- Vendors JS -->

<!-- Main JS -->
<scriptsrc="/static/js/main.js"></script>

<!-- Page JS -->

<!-- Place this tag in your head or just before your close body tag. -->
<scriptasyncdefersrc="https://buttons.github.io/buttons.js"></script>
</body>
</html>
```

8.2 Screen Shots

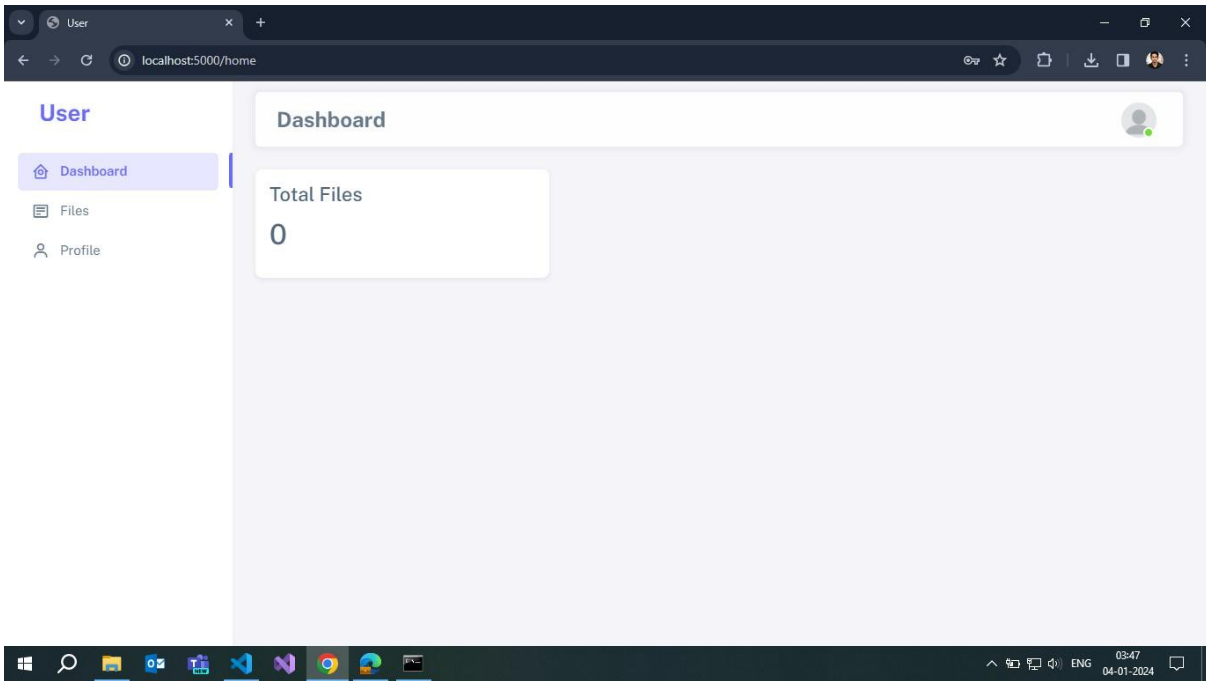


User Signup!

NAME: Reegan

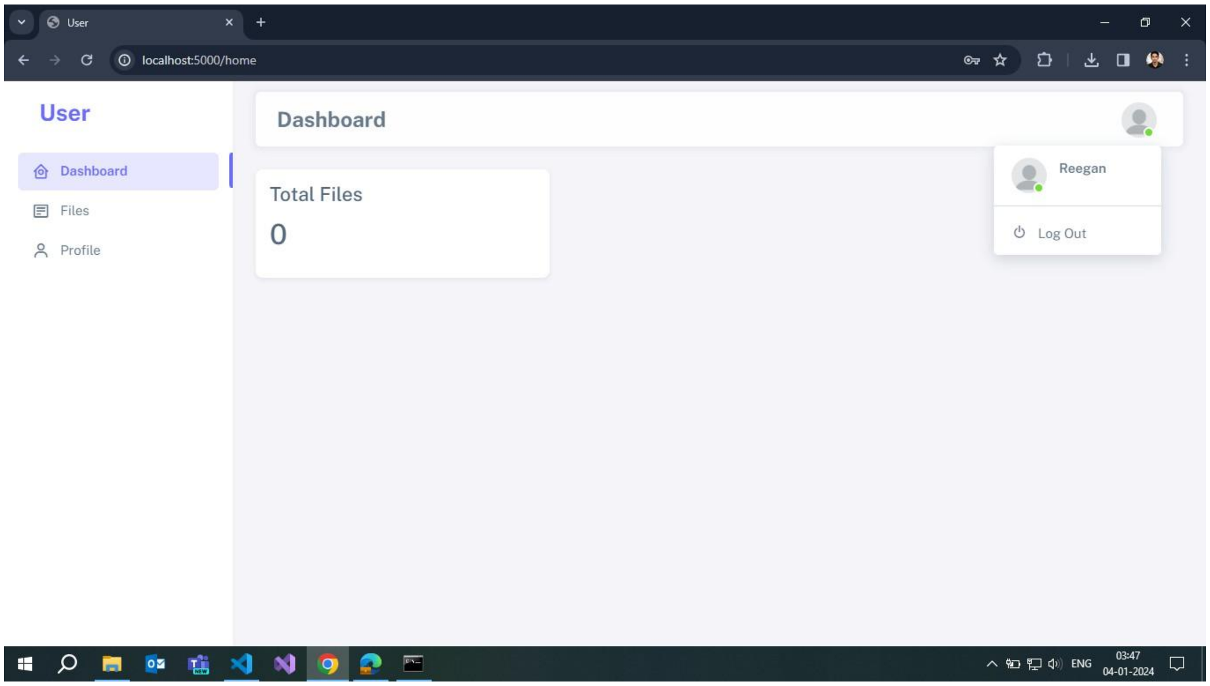
EMAIL: reegan@gmail.com

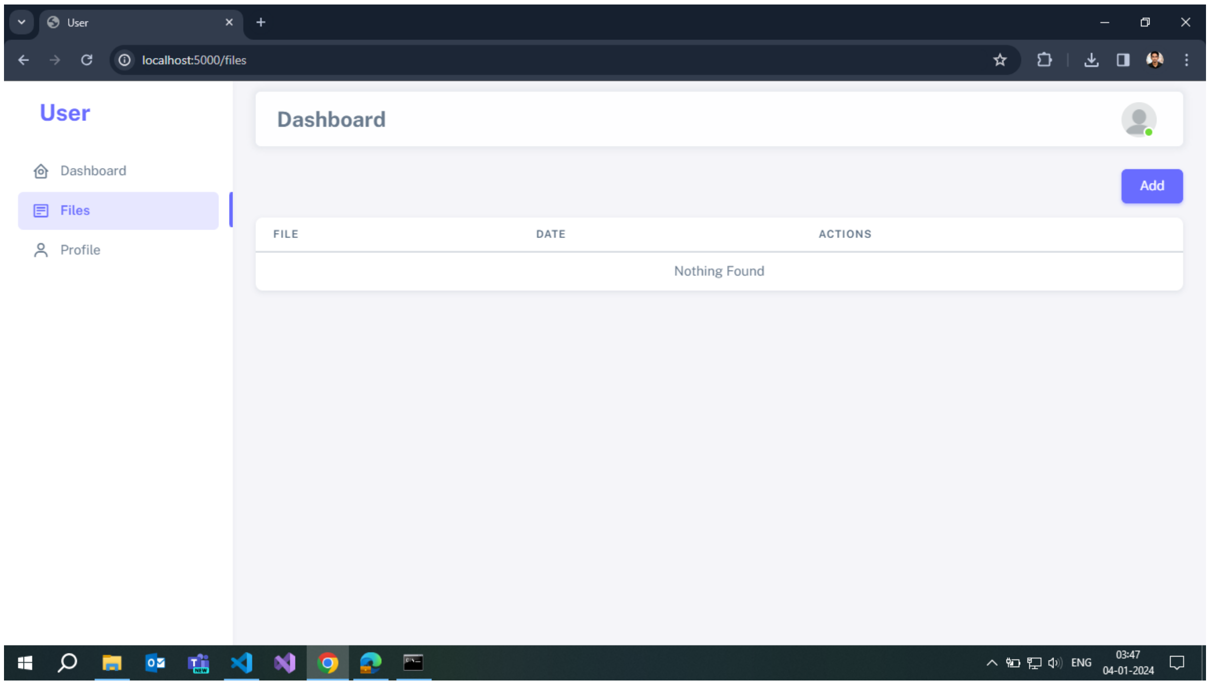
PASSWORD: ••••••



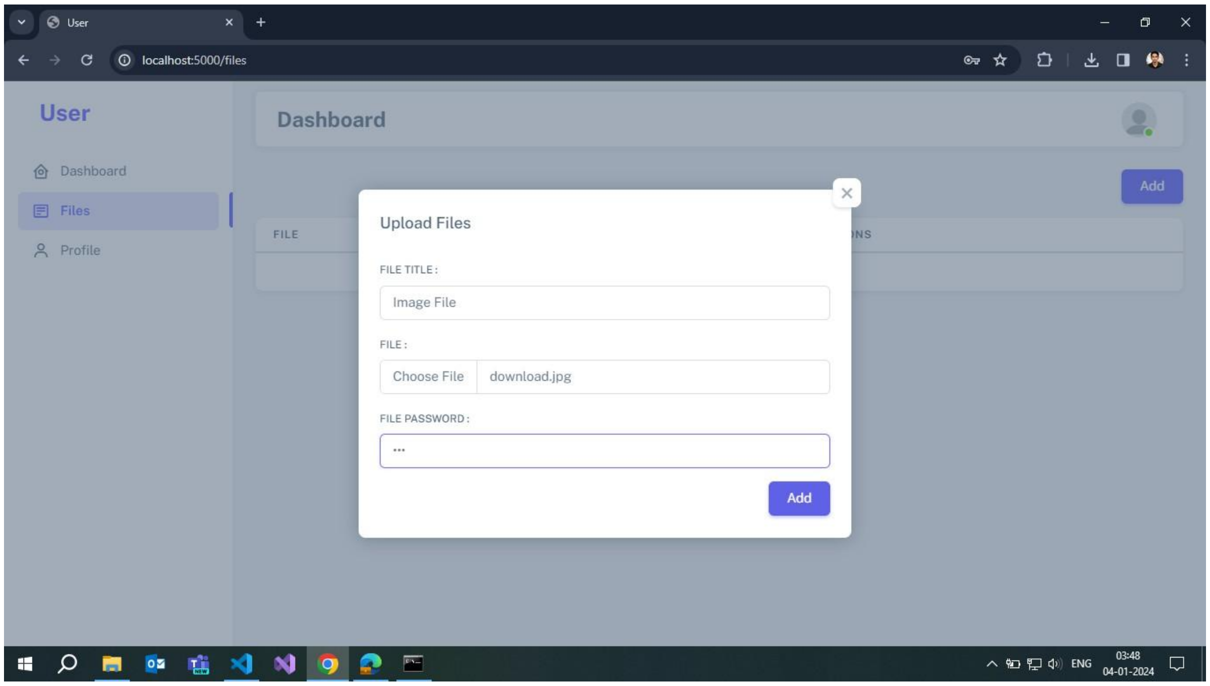
Total Files

0





FILE	DATE	ACTIONS
Nothing Found		

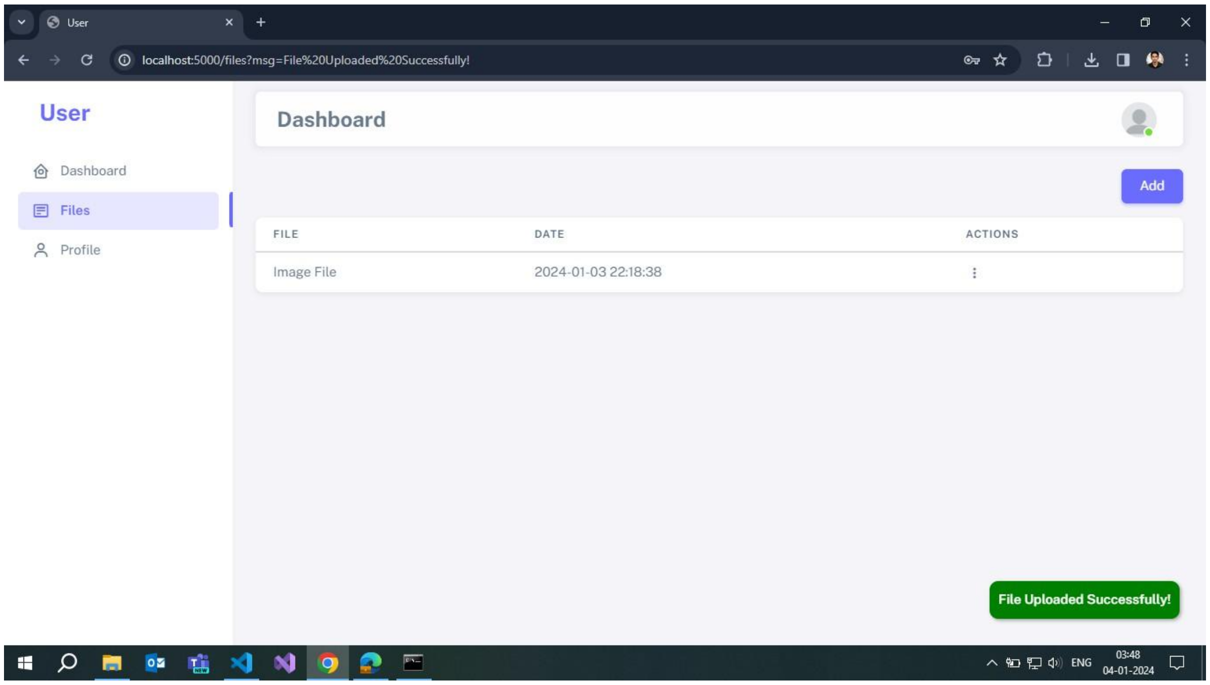

localhost:5000/files

Upload Files

FILE TITLE: Image File

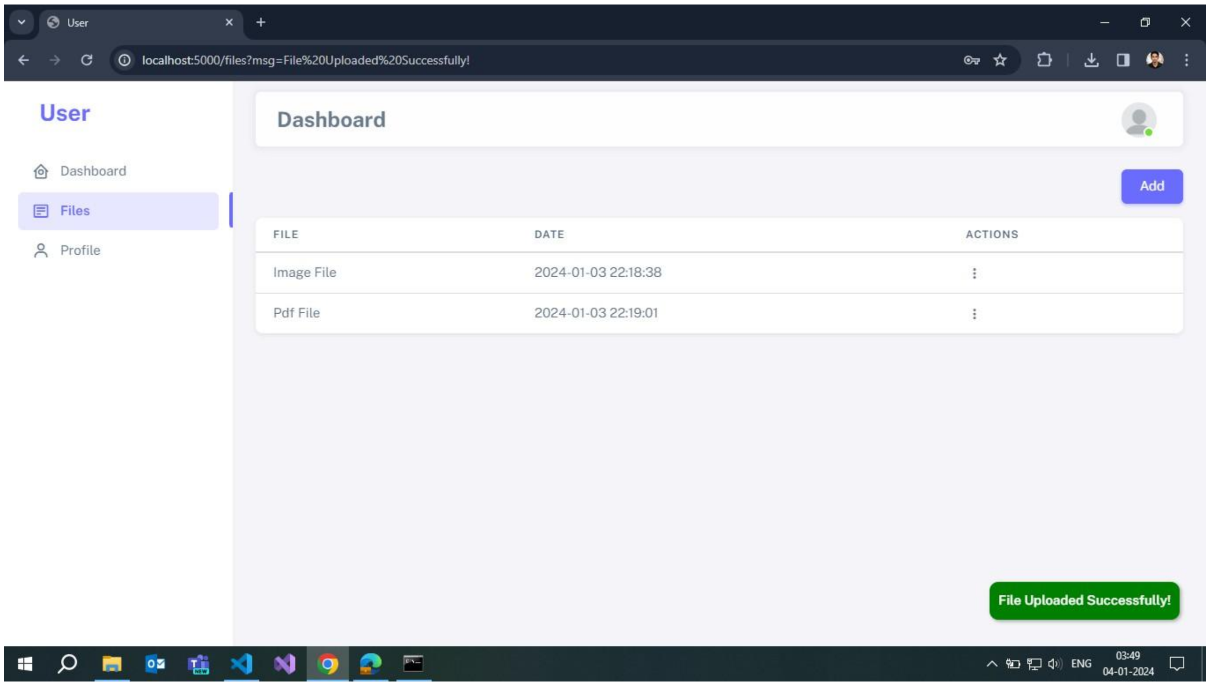
FILE: Choose File download.jpg

FILE PASSWORD: •••


localhost:5000/files?msg=File%20Uploaded%20Successfully!

FILE	DATE	ACTIONS
Image File	2024-01-03 22:18:38	⋮

File Uploaded Successfully!


User
localhost:5000/files?msg=File%20Uploaded%20Successfully!
User
Dashboard
Files
Profile
Dashboard
Add
File Uploaded Successfully!
03:49
04-01-2024

FILE	DATE	ACTIONS
Image File	2024-01-03 22:18:38	⋮
Pdf File	2024-01-03 22:19:01	⋮

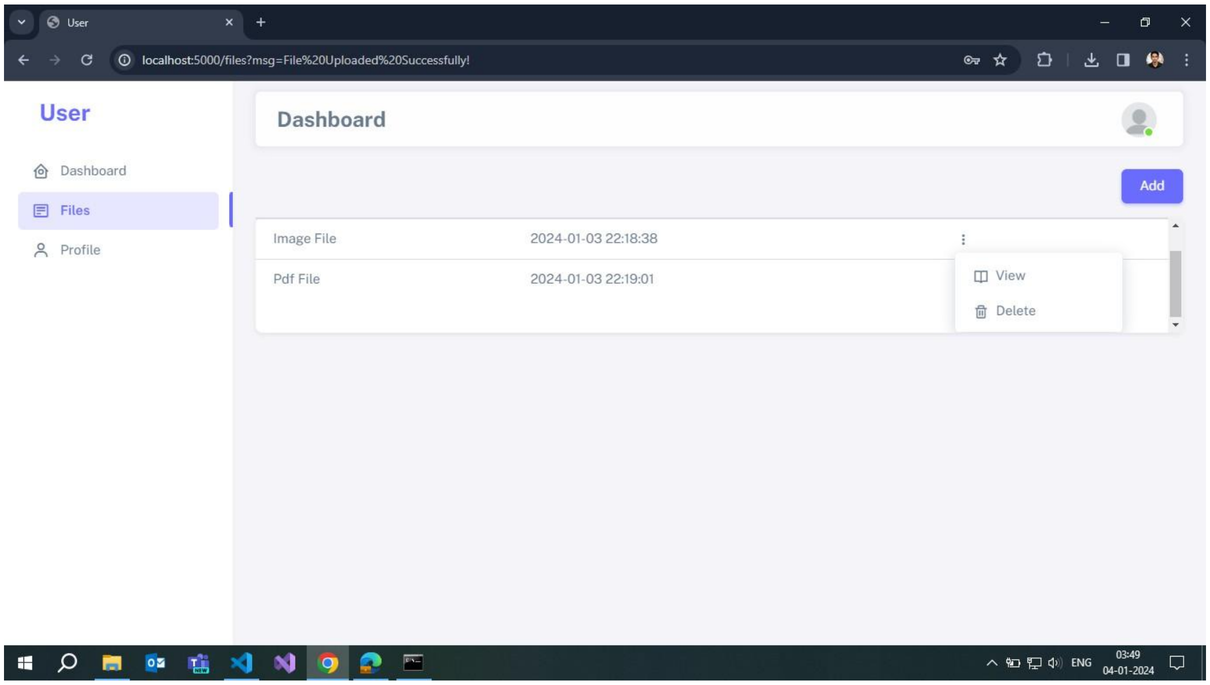
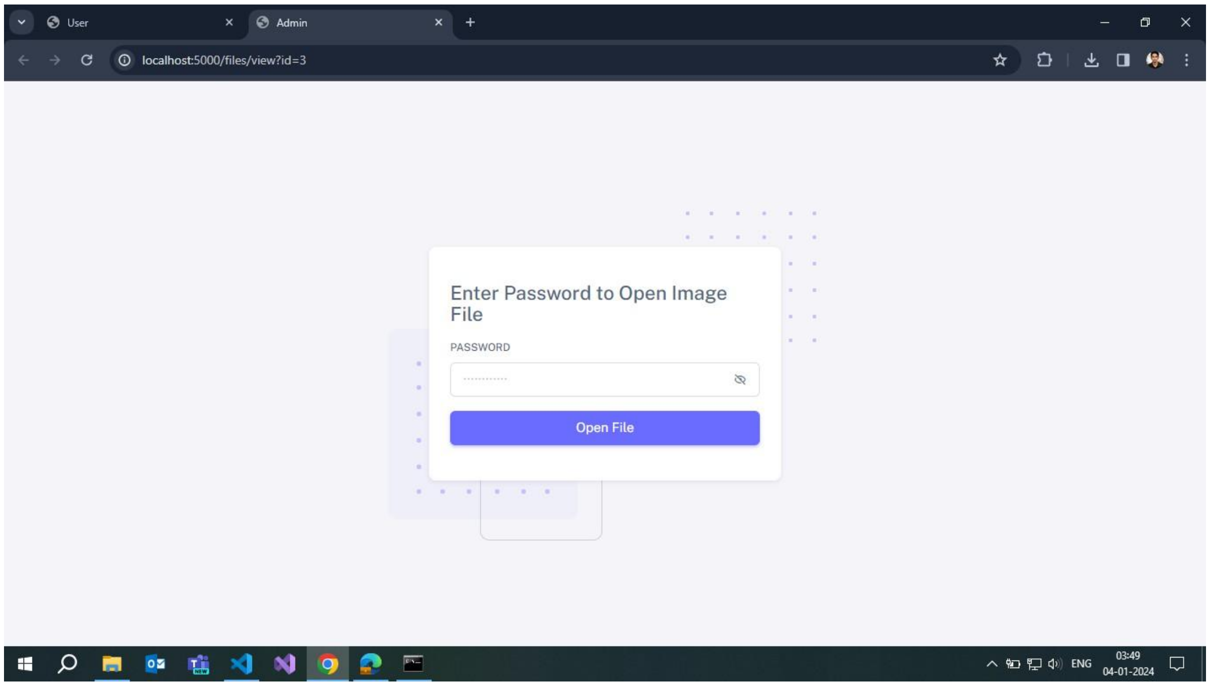
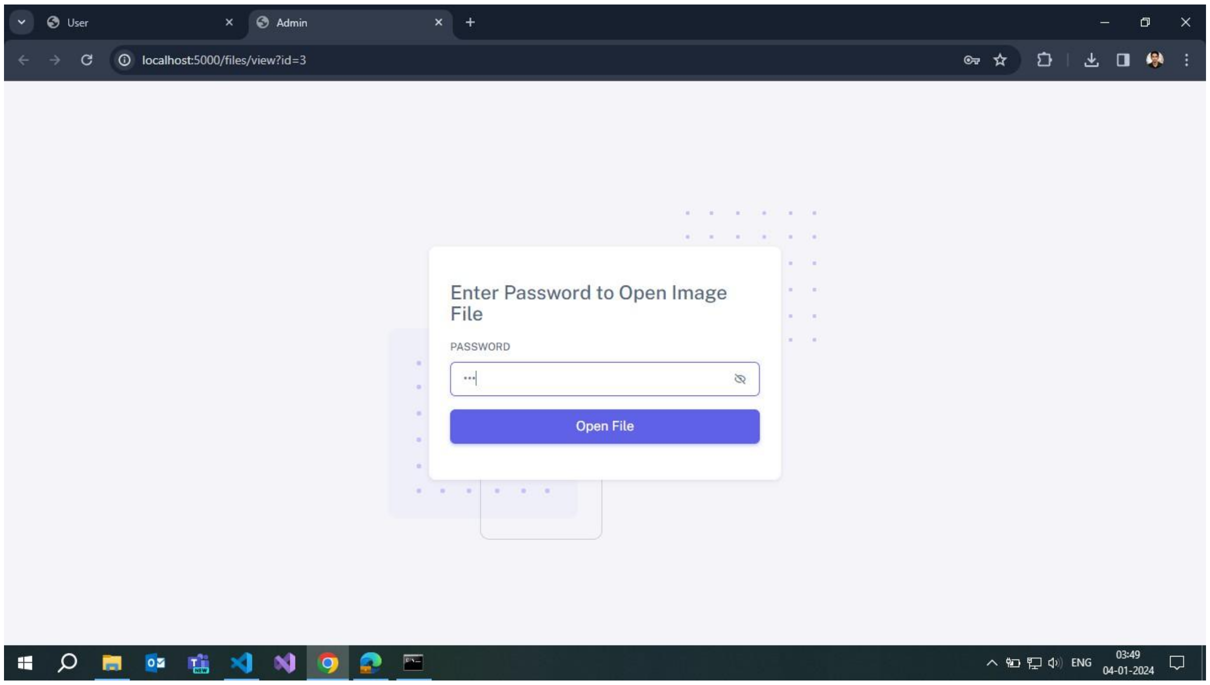
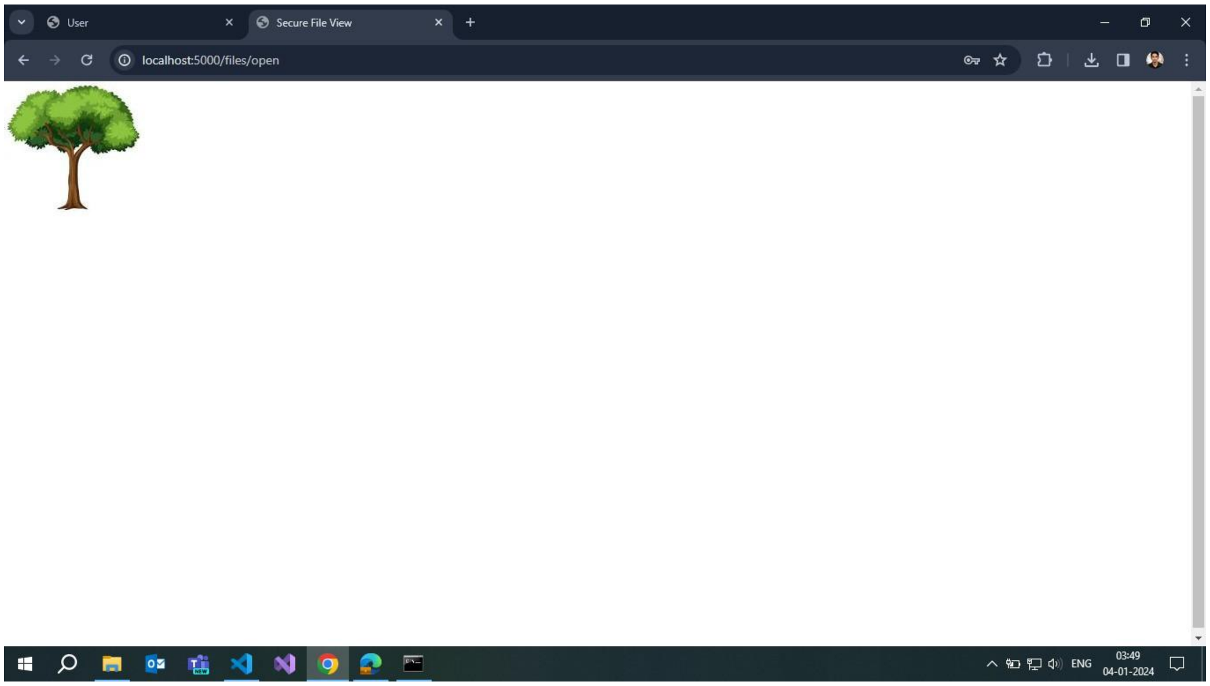
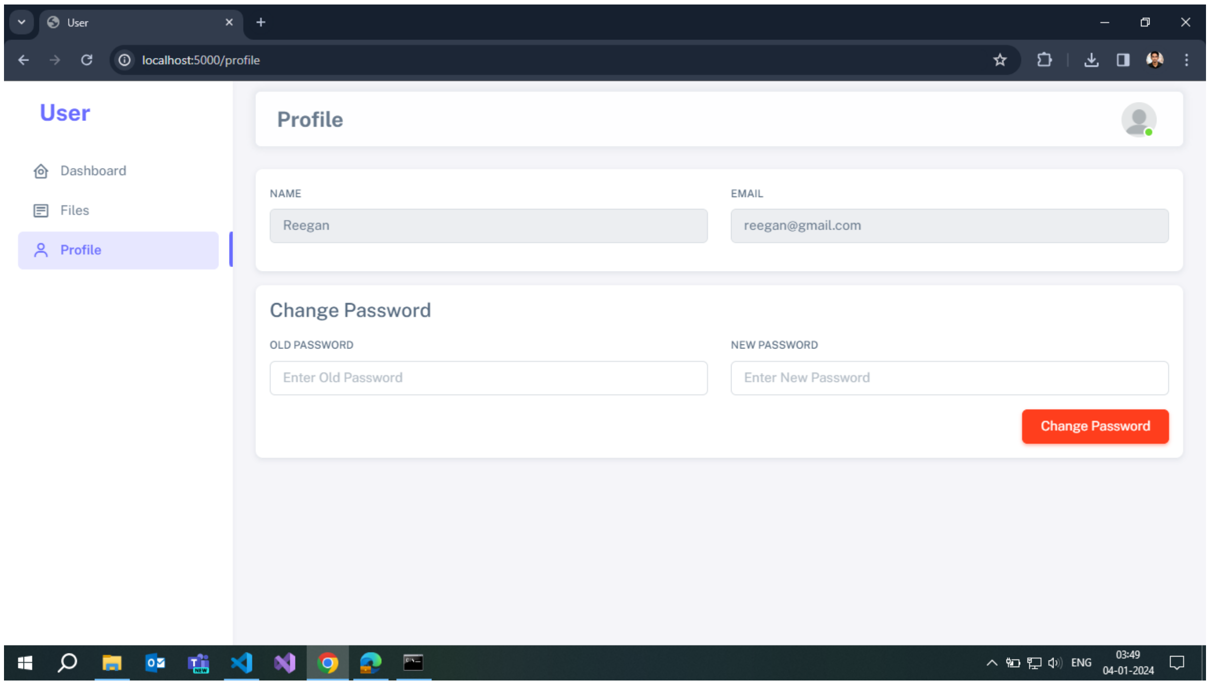


Image File	2024-01-03 22:18:38	⋮
Pdf File	2024-01-03 22:19:01	


User
Admin
localhost:5000/files/view?id=3
Enter Password to Open Image File
PASSWORD
Open File


User
Admin
localhost:5000/files/view?id=3
Enter Password to Open Image File
PASSWORD
Open File





NAME

Reegan

EMAIL

reegan@gmail.com

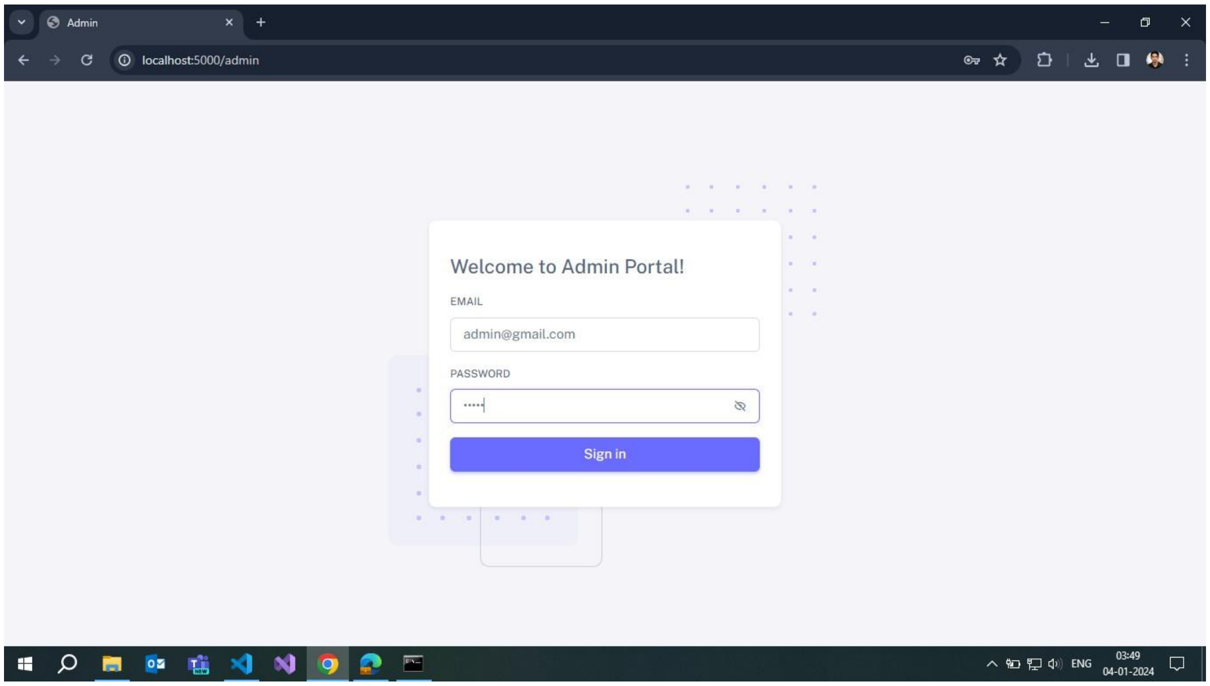
Change Password

OLD PASSWORD

Enter Old Password

NEW PASSWORD

Enter New Password



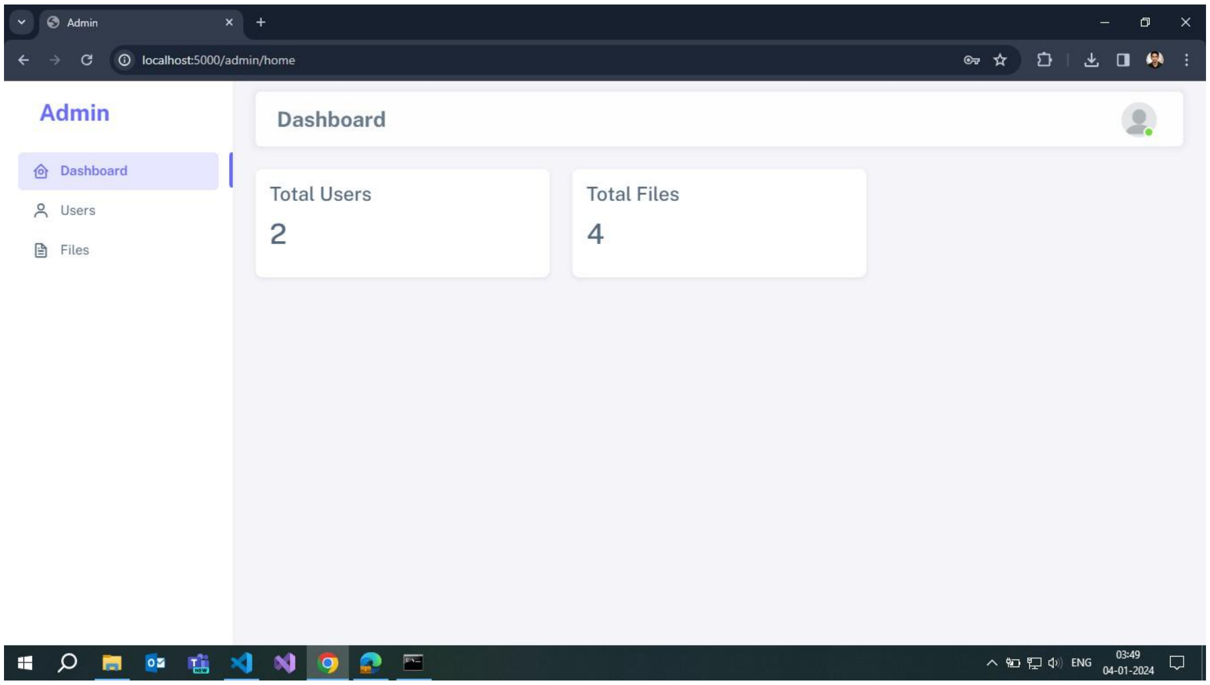
Welcome to Admin Portal!

EMAIL

admin@gmail.com

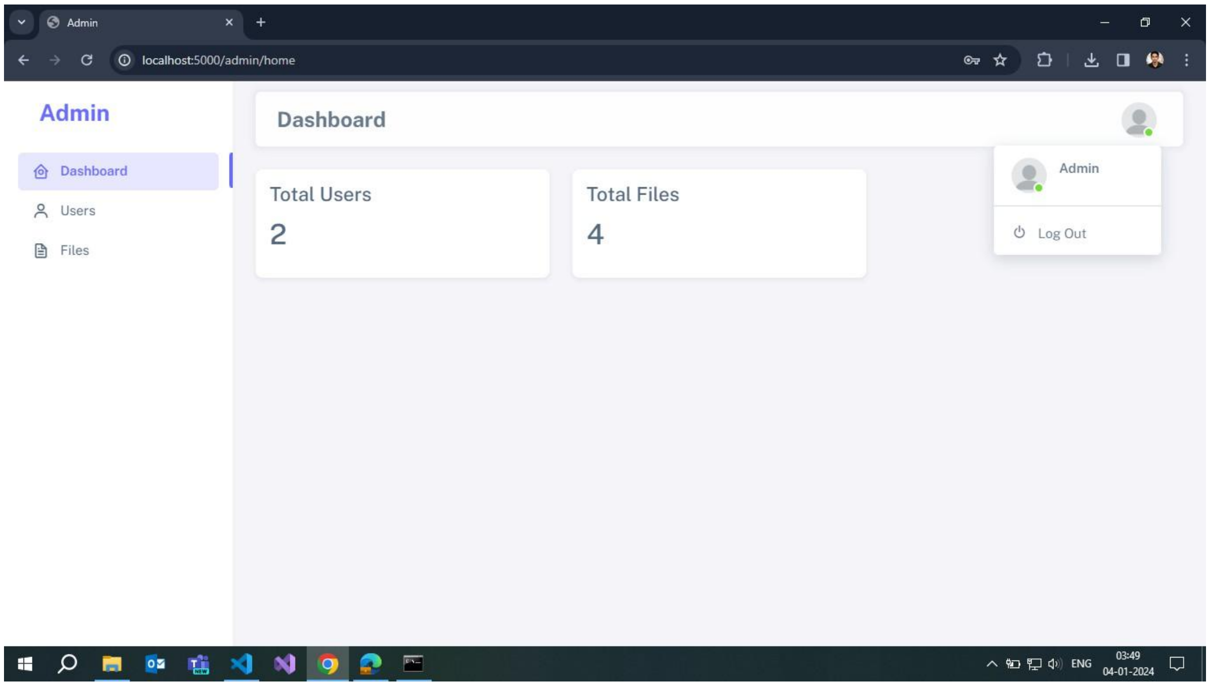
PASSWORD

•••••

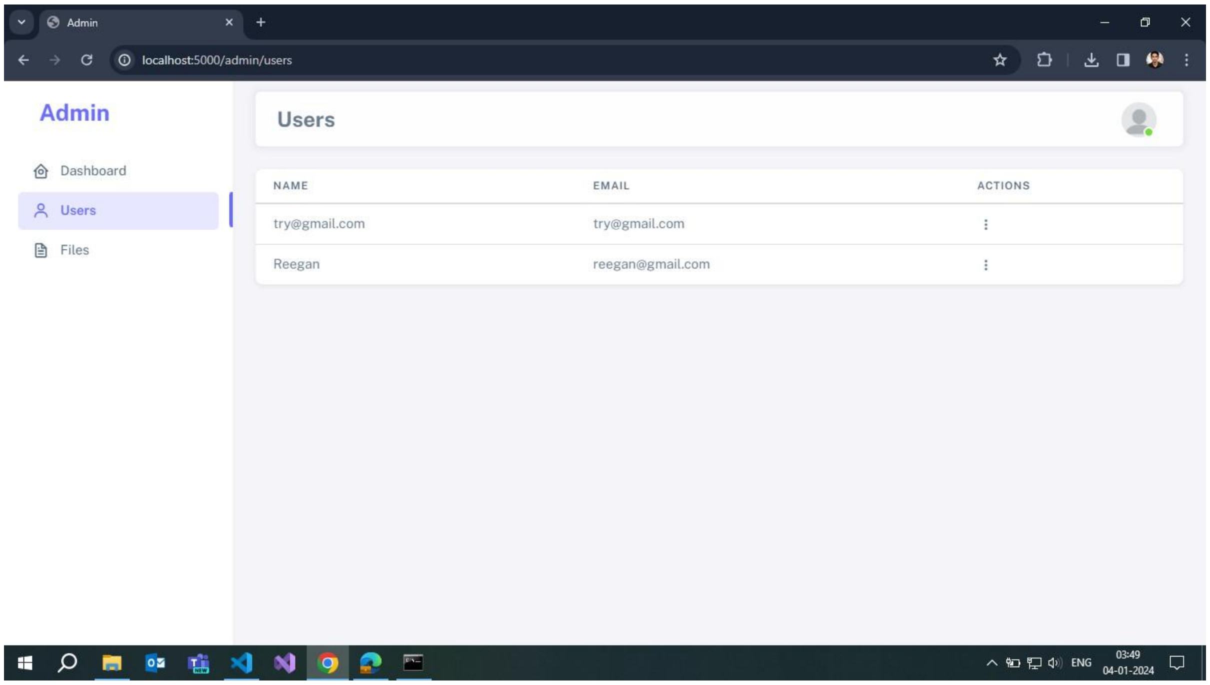


Dashboard

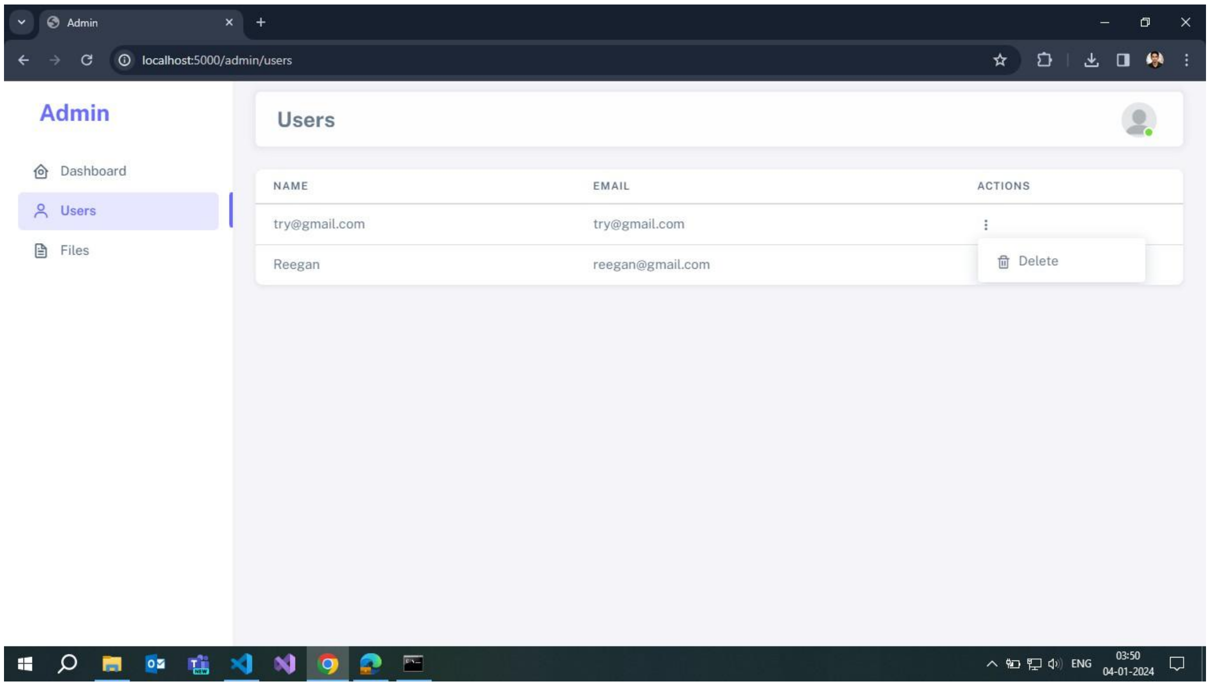
Total Users	Total Files
2	4



Total Users	Total Files
2	4



NAME	EMAIL	ACTIONS
try@gmail.com	try@gmail.com	⋮
Reegan	reegan@gmail.com	⋮


Admin

localhost:5000/admin/users

Admin

Dashboard

Users

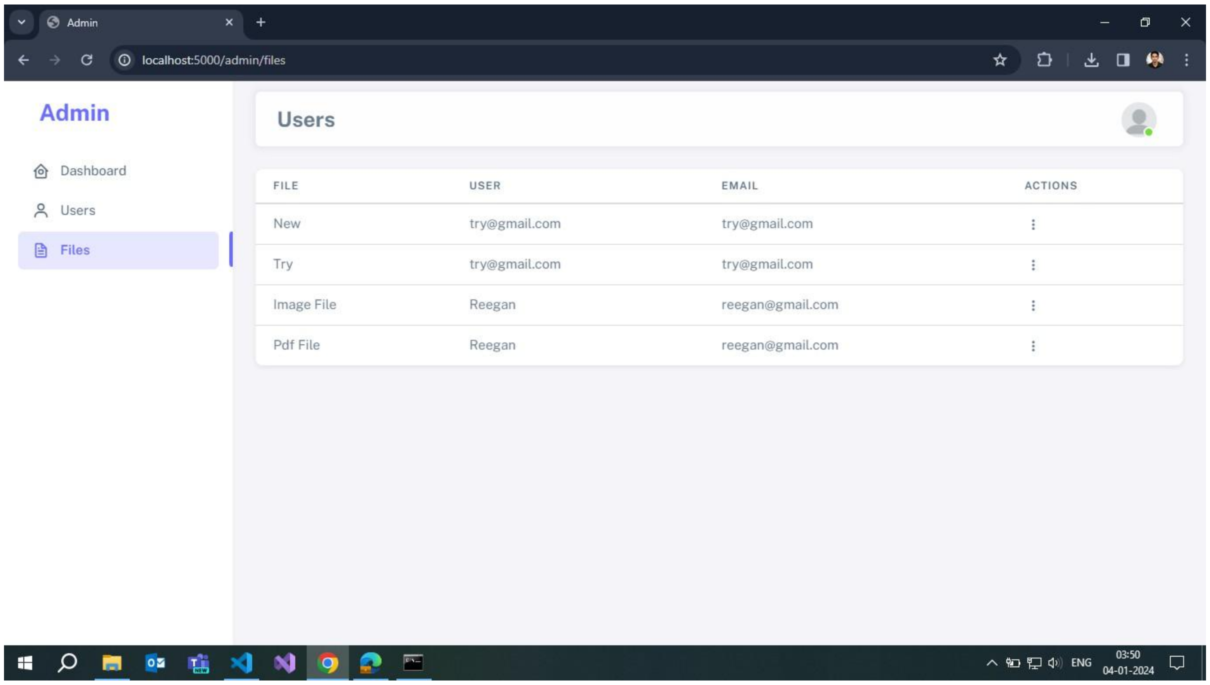
Files

Users

NAME	EMAIL	ACTIONS
try@gmail.com	try@gmail.com	⋮
Reegan	reegan@gmail.com	

Delete

03:50
04-01-2024


Admin

localhost:5000/admin/files

Admin

Dashboard

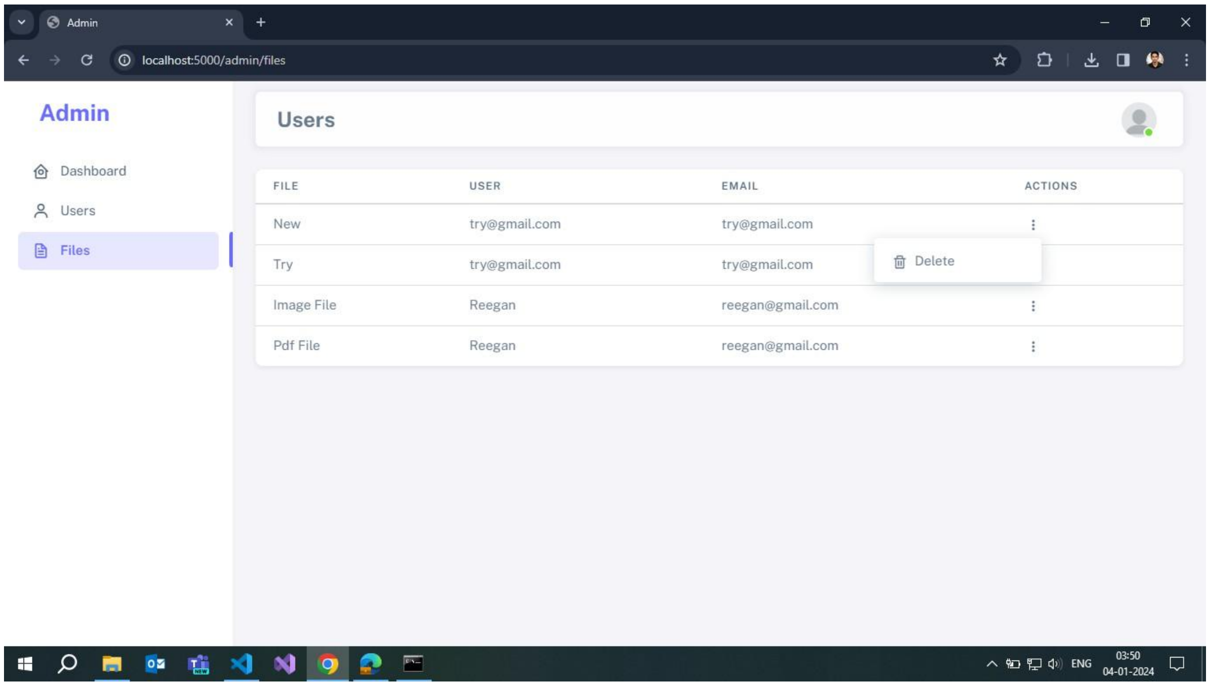
Users

Files

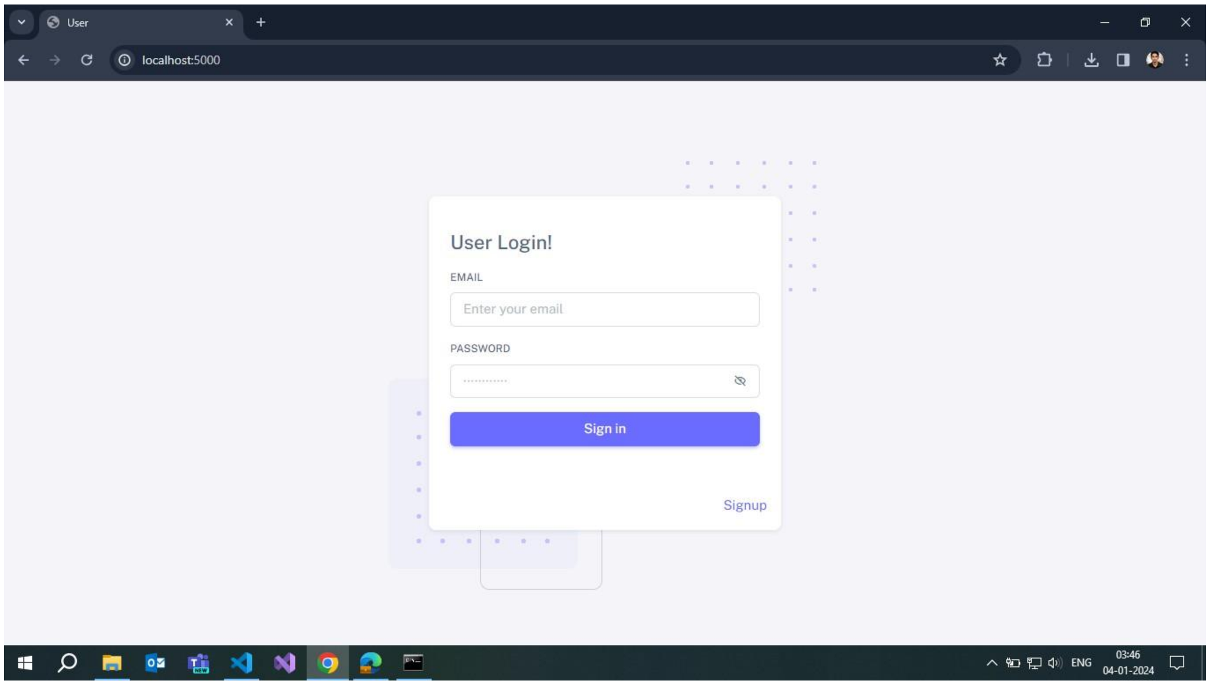
Users

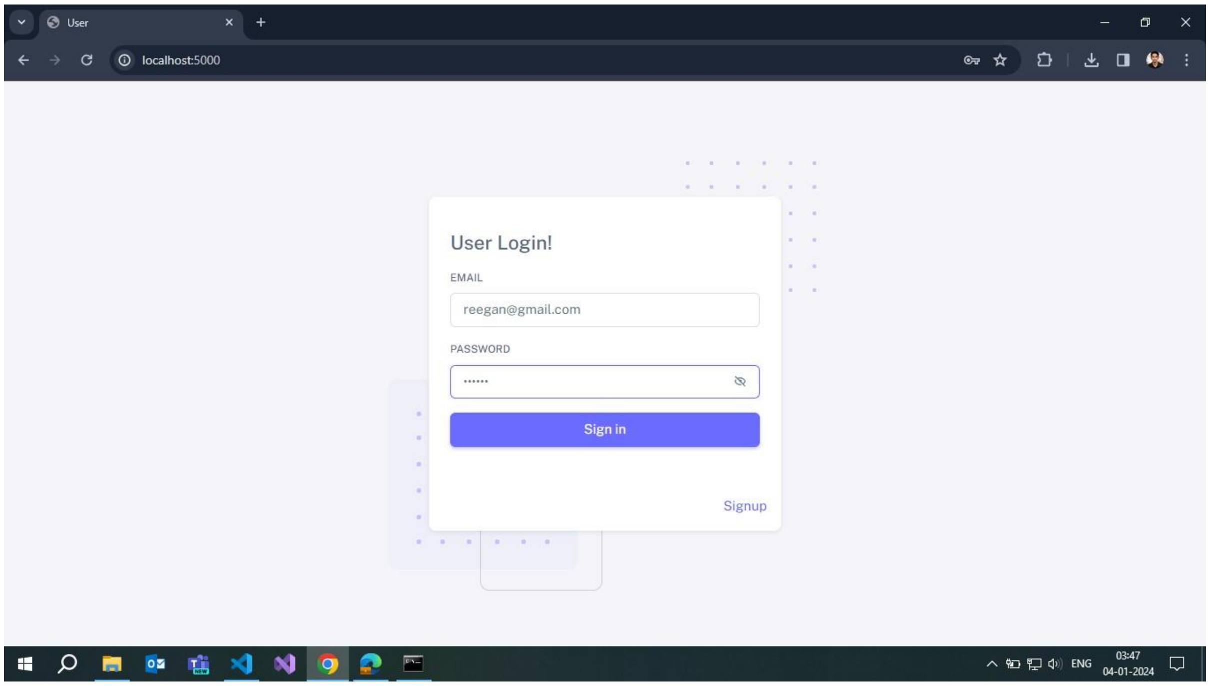
FILE	USER	EMAIL	ACTIONS
New	try@gmail.com	try@gmail.com	⋮
Try	try@gmail.com	try@gmail.com	⋮
Image File	Reegan	reegan@gmail.com	⋮
Pdf File	Reegan	reegan@gmail.com	⋮

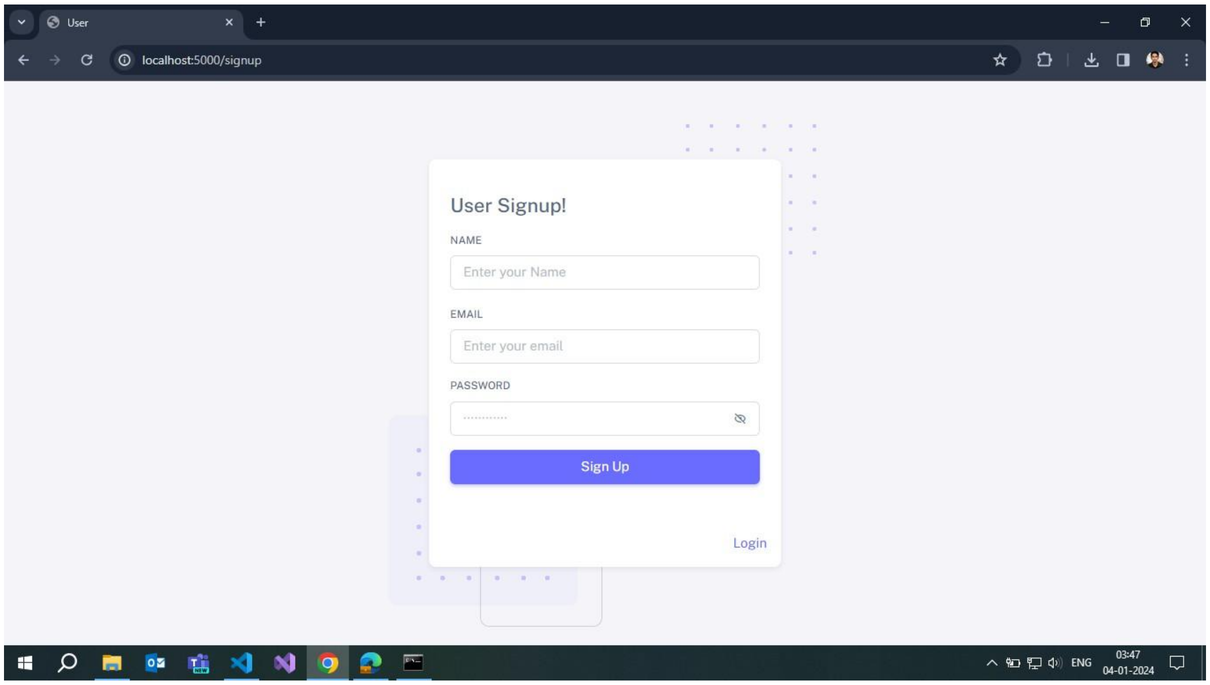
03:50
04-01-2024



FILE	USER	EMAIL	ACTIONS
New	try@gmail.com	try@gmail.com	⋮
Try	try@gmail.com	try@gmail.com	
Image File	Reegan	reegan@gmail.com	⋮
Pdf File	Reegan	reegan@gmail.com	⋮




User
localhost:5000
User Login!
EMAIL
reegan@gmail.com
PASSWORD
Sign in
Signup
03:47
04-01-2024


User
localhost:5000/signup
User Signup!
NAME
Enter your Name
EMAIL
Enter your email
PASSWORD
Sign Up
Login
03:47
04-01-2024

CHAPTER 9

CONCLUSION

In an era where data is the lifeblood of digital operations, ensuring its security and accessibility in the cloud is non-negotiable. Smart data distribution and duplication emerge as key strategies in achieving these goals. By intelligently segmenting and replicating data across geographically and logically diverse nodes, organizations can significantly reduce the risk of data loss, enhance disaster recovery capabilities, and optimize system performance. Furthermore, these techniques bolster resilience against cyber threats while enabling faster access and improved load balancing.

As cloud environments continue to evolve, the integration of machine learning and real-time analytics into data distribution frameworks will further refine these processes, making them more adaptive and predictive. Ultimately, the fusion of smart data strategies with robust cloud architectures not only strengthens security and performance but also lays the groundwork for a more reliable and scalable digital

CHAPTER 10

REFERENCES

[1] Mazhar Ali, Samee U. Khan, “DROPS: Division and Replication of Data in Cloud for Optimal Performance and Security”, IEEE 2023.

[2] J. J. Wylie, M. Bakkaloglu, V. Pandurangan, M. W. Bigrigg, S. Oguz, K. Tew, C. Williams, G. R. Ganger, and P. K. Khosla,“Selecting the right data distribution scheme for a survivable storage system”, Carnegie Mellon University, Technical Report CMU-CS-01-120, May 2023.

[3] S. U. Khan, and I. Ahmad,“Comparison and analysis of ten static heuristics-based Internet data replication techniques”, Journal of Parallel and Distributed Computing, Vol. 68, No. 2,pp. 113-136,2022.

[4] D. Boru, D. Kliazovich, F. Granelli, P. Bouvry, and A. Y. Zomaya,“Energy-efficient data replication in cloud computing datacenters”, In IEEE GlobecomWorkshops,pp. 446-451,2022.

[5] Loukopoulos and I. Ahmad, “Static and adaptive distributed data repli-cation using genetic algorithms”, Journal of Parallel and Distributed Computing, Vol. 64, No. 11,pp. 1270-1285,2021.

[6] K. Bilal, S. U. Khan, L. Zhang, H. Li, K. Hayat, S. A. Madani, N. Min-Allah, L. Wang, D. Chen, M. Iqbal, C. Z. Xu, and A. Y. Zomaya, “Quantitative comparisons of the state of the art data center architectures”, Concurrency and Computation: Practice and Experience, Vol. 25, No. 12,pp. 1771-1783,2021.

[7] K. Hashizume, D. G. Rosado, E. Fernndez-Medina, and E. B. Fernandez, “An analysis of security issues for cloud computing”, Journal of Internet Services and Applications, Vol. 4, No. 1,pp. 1-13,2020.

[8] K. Lai, M. Feldman, I. Stoica, and J. Chuang, “Incentives for cooperation in peer-to-peer networks”, in Proc. 1st Workshop Economics Peer-toPeer Syst.,pp. 631660,2020.
[9] ManishaKalkal, SonaMalhotra, “Replication for Improving Availability and Balancing Load in Cloud Data Centres”, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 5, Issue 4, 2019.
[10]S. M. Khan and K. W. Hamlen, Hatman,“Intra-cloud trust management for Hadoop”, in Proc. 5th Int. Conf. Cloud Comput., 2019
[11]S. Pearson and A. Benameur, Privacy, “security and trust issues arising from cloud computing”, in Proc. 2nd Int. Conf. Cloud Comput.,pp. 693702,2018.
[12]E. Bertino, F. Paci, R. Ferrini, and N. Shang, “Privacy-preserving digital identity management for cloud computing”, IEEE Data Eng. Bull, vol. 32, no. 1, pp. 2127, Mar. 2018.
[13]F. Skopik, D. Schall, and S. Dustdar, “Start trusting strangers bootstrapping and prediction of trust”, in Proc. 10th Int. Conf. Web Inf. Syst. Eng., pp. 275289,2017.
[14]H. Guo, J. Huai, Y. Li, and T. Deng, “KAF: Kalman filter based adaptive maintenance for dependability of composite services”, in Proc. 20th Int. Conf. Adv. Inf. Syst. Eng.,pp. 328342,2016.
[15]Y. Wei and M. B. Blake, “Service-oriented computing and cloud computing: Challenges and opportunities”, IEEE Internet Comput., vol. 14, no. 6, pp. 7275, Nov./Dec. 2015.
[16]B. Fung, K. Wang, R. Chen, and P. Yu, “Privacy-preserving data publishing: A survey of recent developments”, ACM Comput. Surv., vol. 42, no. 4, pp. 153, 2014