SECURE DEDUPLICATION OF ENCRYPTED DATA IN CLOUD

Main Project Report

Submitted by

MANJIMA VARGHESE

Reg no:FIT20MCA-2074

Submitted in partial fulfillment of the requirements for the award of the degree of

Master of Computer Applications

Of

A P J Abdul Kalam Technological University

FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)®

ANGAMALY-683577, ERNAKULAM(DIST)

JULY 2022

DECLARATION

I hereby declare that the report of this project work, submitted to the Department of Computer Applications, Federal Institute of Science and Technology (**FISAT**), Angamaly in partial fulfillment of the award of the degree of Master of Computer Application is an authentic record of my original work.

The report has not been submitted for the award of any degree of this university or any other university.

Date : 8/8/22

Place: Angamaly

Manjima

MANJIMA VARGHESE

FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)®

An ISO 9001:2015 Certified, NAAC ('A' Grade) Accredited Institution with NBA Accredited Programmes
(Approved by AICTE - Affiliated to APJ Abdul Kalam Technological University, Kerala)
Owned and managed by Federal Bank Officers' Association Educational Society

us on Excellence

HORMIS NAGAR, MOOKKANNOOR P.O., ANGAMALY - 683 577, ERNAKULAM DT., KERALA, INDIA.
Tel: (O) 0484-2725272 Fax: 0484-2725250 E-mail: mail@fisat.ac.in Website: www.fisat.ac.in

11TH July 2022

TO WHOMSOEVER IT MAY CONCERN

This is to certify that Mr./Ms. MANJIMA VARGHESE (Reg. No. FIT20MCA-2074) has successfully completed his/her Main Project with the title "SECURE DEDUPLICATION OF ENCRYPTED DATA IN CLOUD", in the Department of Computer Applications, FISAT, during the period from 30th March 2022 to 11th July 2022.

[illegible]

Dr DEEPA MARY MATHEWS

HEAD OF THE DEPARTMENT

FEDERAL INSTITUTE OF SCIENCE & TECHNOLOGY (FISAT) ANGAMALY
DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS

FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)®

ANGAMALY, ERNAKULAM-683577

Focus on Excellence

CERTIFICATE

This is to certify that the project report titled " **SECURE DEDUPLICATION OF ENCRYPTED DATA IN CLOUD** " submitted by **MANJIMA VARGHESE, (Reg No: FIT20MCA-2074**) towards partial fulfillment of the requirements for the award of the degree of Master of Computer Applications is a record of bonafide work carried out by her during the year 2022.

[illegible] 8/8/22

Project Guide

Dr. Sujesh P Lal

FEDERAL INSTITUTE OF SCIENCE & TECHNOLOGY (FISAT) ANGAMALY ★ DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS

Head of the Department [illegible]

Dr. Deepa Mary Mathews

Submitted for the viva-voce held on .... at ....

Examiner :

ACKNOWLEDGEMENT

With pleasure, I am submitting the main project which I did as a part of my curriculum. I am very grateful to almighty God who let me in the right way and showered upon me his blessings throughout the successful completion of this project.

I am very much indebted to **Dr. Manoj George**, Principal, FISAT, Angamaly and **Dr. C Sheela**, Vice Principal, FISAT, Angamaly, for encouragement and co-operation. Our sincere thanks to **Dr. Deepa Mary Mathews**, Head of the department of Computer Applications, FISAT, who gave the source of inspiration for achieving greater height in the pursed of excellence.

The project report titled "Secure deduplication of encrypted data in cloud" has been done under the guidance of scrum master **Ms. Manju Joy** and internal guide **Dr. Sujesh P Lal** With great respect I express my sincere and heartfelt thanks for her guidance for preparing this project.

Last but not the least, I express our heartfelt word of thanks to my parents and friends who have given me inspiration, mental support and lot of help and encouragement for doing this project successfully. My heart flows out to all those who helped me directly or indirectly in the completion of this work.

ABSTRACT

Nowadays, most people are sharing their data on the cloud. Cloud computing has emerged as a popular and effective data management tool for businesses. Each day, around 2.5 quintillion bytes of data is generated on the internet. In order to store this large amount of data, we need servers that can deduplicate data efficiently so as to minimise the use of network bandwidth and storage. In this project, we use the concept of hybrid clouds and hashing techniques to achieve this deduplication. The results show that redundant data is always mapped onto the same hash code and thus it does not get uploaded onto the cloud servers, thus ensuring successful deduplication. It saves storage space as well as bandwidth by eliminating duplicate data.

In this project, data from the data provider gets stored on the cloud server. With effective deduplication, ensuring data confidentiality is also very important. Thus, data is always stored in the cloud in an encrypted format. It is achieved with the help of the advanced SHA1 algorithm. Likewise, I created a private cloud which provides users the ability to store their personal data.Cloud platforms such as Amazon S3 and Microsoft Azure offer cloud storage for files uploaded by both users and providers. Data provider uploaded files will be initially stored on the private cloud on drive for deduplication before being moved to the public cloud, which is created on the aws s3 storage. Likewise,the file uploaded by the user will be stored on the Microsoft Azure platform and will be used for their personal storage. It will not be shared with anyone.

Compared with the prior data deduplication systems, our system has two advantages. Firstly, it can be used to confidentially share data with users by specifying access policies rather than sharing decryption keys. Secondly, it achieves the standard notion of semantic security for data confidentiality while existing systems only achieve it by defining a weaker security notion.

Contents

Chapter 1

Introduction

Data de-duplication is the process of removing multiple copies of the same data. This will happen when a data provider tries to upload a file that has already been uploaded by another data provider. This is why the de-duplication method has been adopted by various providers of cloud storage.Hashing technique SHA1 helps with deduplication and storing data in encrypted form in the cloud.Using the SHA1 method, we can divide the uploaded file into different blocks. It will create a hash value for each block. If the hash value of newly uploaded files matches the old one, then it will restore the index. Otherwise, it will store the data in the public cloud.

In this project, we introduce attribute-based sharing rather than providing the keys to the user. The attribute authority will group the users based on their attributes and share the corresponding file among the users who satisfy the attribute values. And we're introducing the concept of a hybrid cloud for secure data storage and deduplication. Different cloud platforms, such as Amazon S3 and Microsoft Azure, are used to build public and private clouds.Amazon S3 is object storage where we can store and retrieve any amount of data from anywhere. It's a simple storage service.It is used to create a public cloud which is used to store the files uploaded by the data provider after deduplication. The private cloud is implemented by the drive in the computer. Microsoft Azure, also known as Windows Azure, is Microsoft's public cloud computing platform. It provides cloud services, including compute, analytics, storage, and networking.

- In cloud computing, encryption has become common in recent years, but in order to ensure data security, we introduce a new hashing technique which will help to achieve both encryption and deduplication of data in cloud.

In an existing system, the cloud is a platform that is used by different data providers to share their data. On a cloud platform, we store the data in an encrypted form, and the corresponding users can download the data using the decryption key.Users with certain credentials will be able to access the data stored in the cloud.In an existing system, data deduplication is implemented by removing repetitively uploaded files. Each time a new user is requested for a particular file stored in the cloud, it will provide the key to them to access the data if they possess the right certain attributes. The standard ABE system does not support secure deduplication, which is crucial for eliminating duplicate copies of identical data in order to save storage space and network bandwidth.The existing system does not provide an efficient way to group-wise share data.It does not provide a private cloud platform for users to store their personal data.

In this project, we present an attribute-based storage system and attribute-based group sharing.Firstly, we are introducing a hashing algorithm which is used to check the existence of data in the public cloud. If the data is present in the public cloud, then it will not upload the data. Instead, it will restore the index of the already existing data, which is equal to the uploaded data. Otherwise, it will upload the data to the public cloud. In this way, it provides an efficient way of deduplicating data in the cloud.Secondly, we provide a separate private cloud for the registered users to store their data on the private cloud platform.

The main advantage of the proposed system is that we bring into our system a hybrid cloud architecture, which consists of a private cloud responsible for deduplication using hashing techniques and a public cloud for storing the data. It can be used to share data with users by specifying access policies rather than sharing decryption keys. Group shaing will help to share data with more the one users where groups are created by the attribute authority.We also provide a separate private cloud for users to store their personal data

Chapter 2

PROOF OF CONCEPT

This project is the combined work of two IEEE papers.One is attribute-based storage, supporting secure deduplication of encrypted data in the cloud, and the other one is secure encrypted data deduplication using a hashing technique in the cloud. Attribute based storage and attribute based group sharing are taken from the first IEEE paper, and the deduplication of encrypted data in the cloud is taken from the second paper.The other IEEE papers that we used to implement this project are: De-Duplication of Data in the Cloud; weak leakage-resilient client-side de-duplication of encrypted data in cloud storage; and Secure Sharing of Personal Records in the Cloud using Encryption.

The paper ”De-Duplication of Data in the Cloud” presents the ideas of detectable deduplication, file encryption and file uploading, multiple cloud storage, file exchange, and file retrieval. It gives the idea of implementing a hashing technique to avoid duplication.The other paper, weak leakage-resilient client-side de-duplication of encrypted data in cloud storage, gives the concepts of Cloud Storage, Client-side Deduplication, Proofs of Ownership, Privacy, and Universal Hash. It gives the idea of attribute-based encryption, ciphertext, encryption, and decryption keys and their implementation.

Finally, the last paper ”Secure Sharing of Personal Records in the Cloud using Encryption” provides the idea that,checking for the duplicates file while uploading the data and store it in the encrypted format.Provide the keys to authorized users.In This project it takes the cocepts of Attribute based storage,Attribute based sharing,Hashing technique for data deduplication and private cloud service for the users to store their personal data.

2.1 Objectives

- To apply the SHA1 hashing method for deduplication of encrypted data in the cloud.

  The SHA1 hashing method is used to store the hash value of the uploaded image. It works in such a way that when a data provider uploads the file, it will be divided into different blocks, and for each value, there will be a hash value. These hash values will be stored instead of the file. When the same file is uploaded by another user, then like above, the corresponding hash value will be checked. If the hash value is the same, then instead of storing the new hash value, it will just restore the index. Otherwise, it will upload the value.

- To make attribute based storage.

  In this project, we are implementing attribute-based storage. When a new data provider or user makes a registration, there will be a option to choose the attribute. The attributes in this case are doctor, engineer, mechanic, and electrical. When a data provider uploads the file after logging in to their profile, it will upload based on the attribute of the data provider. Similarly, in the attribute authority profile, each user is identified by their attribute too.

- To make an attribute based sharing.

  After attribute-based storage and registration, attribute-based sharing is done on the attribute authority. Attribute authorities will create groups in which the file will be shared with users. Each group should have the same attribute. If a particular file is shared among a group, then it will no longer be their turn to share other files. Only one file will be shared at a time. It's for security purposes. The users of that particular group can only download the corresponding file. No other users outside the group can download the file.

- To apply a hybrid cloud platform where the public cloud is used for storing data and the private cloud is used for checking the data duplication.

In this project, a hybrid cloud architecture is used for attribute-based storage and data deduplication. A hybrid cloud is a combination of public and private clouds. The public cloud is implemented on Amazon S3 and the private cloud is implemented on a computer drive. The file will be uploaded to the private cloud. After the data deduplication check, it will be stored on the public cloud.

- To provide a separate private cloud platform for storing the users' personal files.

In this project, we provide additional storage space for users to upload their personal files. Once the registration process is over, users can see their upload option when they log in to their profile. When the user uploads their file, it will be stored in the storage space created on the Microsoft Azure cloud platform.It will not be shared to any one.It's just used for personal data storage, and the user can download it whenever they need it.

Chapter 3

IMPLEMENTATION

3.1 System Architecture

This project is mainly implemented using a cloud platform for storing files and registration details on a database. The modules and their work are given below.

Database

Database is used to store the registration information of both users and data providers.It consist of all the data including the attribute details of user and data provider.

Data Provider

Data provider will upload the data to share with the users.It will be initially stored in the private cloud. Then it will be checked for deduplication. If the file already exists, then restore the index of the already existing location, or else it will be stored on the public cloud.

User

Users can register on their profiles. After registering, they can access their profile and view the groups to which they belong. They can download the data, which is shared through their groups. User-uploaded data will be stored on the private cloud, which is built on Microsoft Azure.This data is used by the user for their personal use.It is not shared with anyone.

Cloud

Cloud is used for storage of all files uploaded by both user and data provider.Microsoft Azure and Amazon S3 are used for cloud storage platforms.The file uploaded by the data provider will be initially stored in the private cloud, which is stored on the drive, after the deduplication using hashing technique it will then permanently stored on the public cloud, which is the Amazon S3 storage platform. The file that the user uploads for personal use will be stored on the private cloud created on Microsoft Azure.The file uploaded by the user will not be shared with any other users. The combination of these public and private clouds is the hybrid cloud architecture. It will help with better security for the stored data and also with effective data deduplication, which helps reduce network bandwidth.

Attribute authority(Admin)

• The attribute authority will see all the users, data providers, and their corresponding uploaded files. The Admin will create a group of users with similar attributes. Then it will share the data which is matched with its attribute. The users in that corresponding group can download the data. Once a group share of files occurs through a particular group , it will never be there. The users other than that corresponding group will not be able to download the shared data.

The project works like,The data will be shared by the data provider and it will be stored on the private cloud which is created on the drive. After the effective deduplication process, it will be stored on the public cloud, which is created on the aws s3 cloud platform. Then the attribute authority will create groups of users with similar attributes and share the file among them. The users in that corresponding group can download the shared file. The file created by the user will be stored on the cloud platform which is created on Microsoft Azure.

The diagram that describes the operation of the system (figure 3.1).

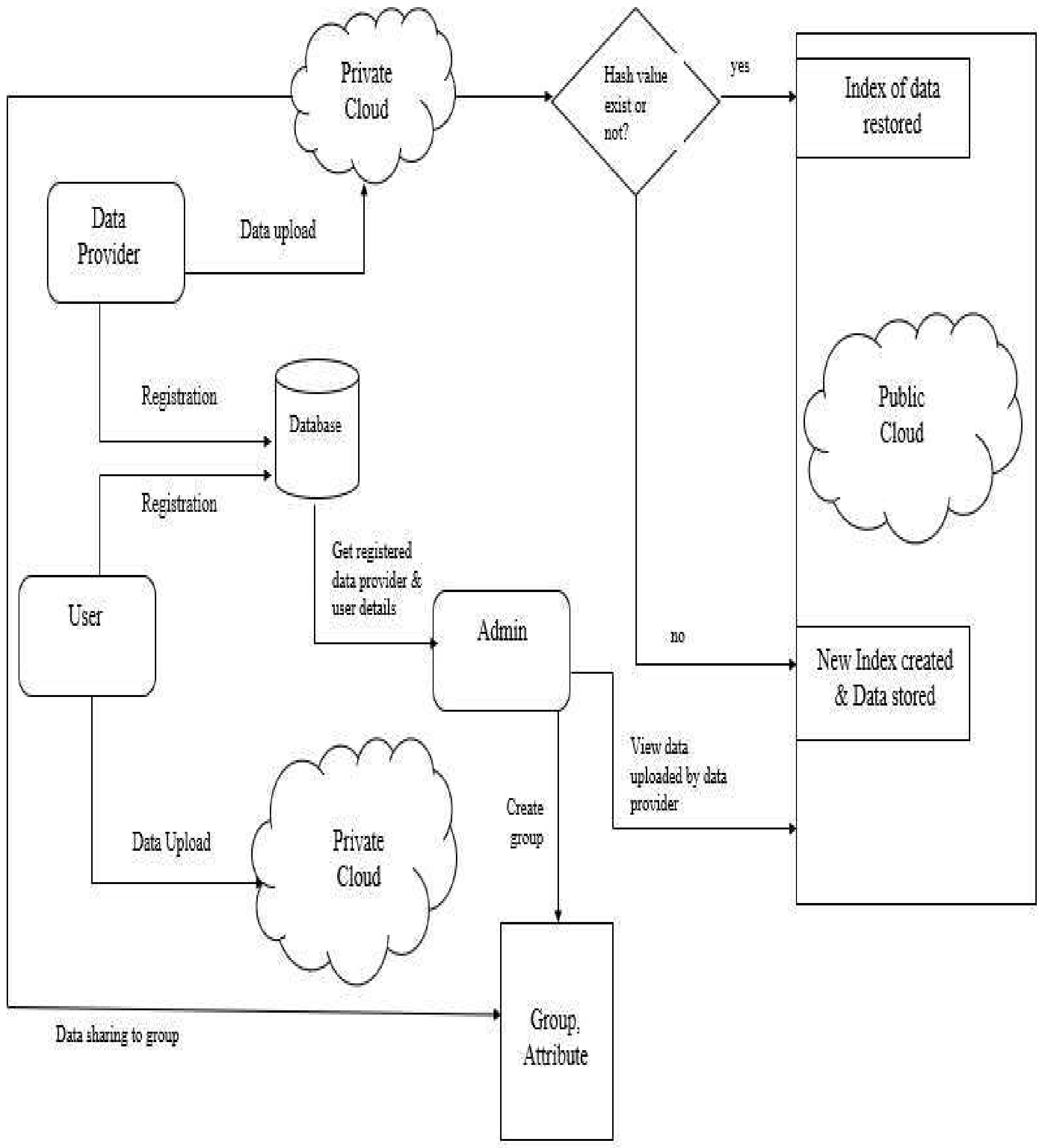
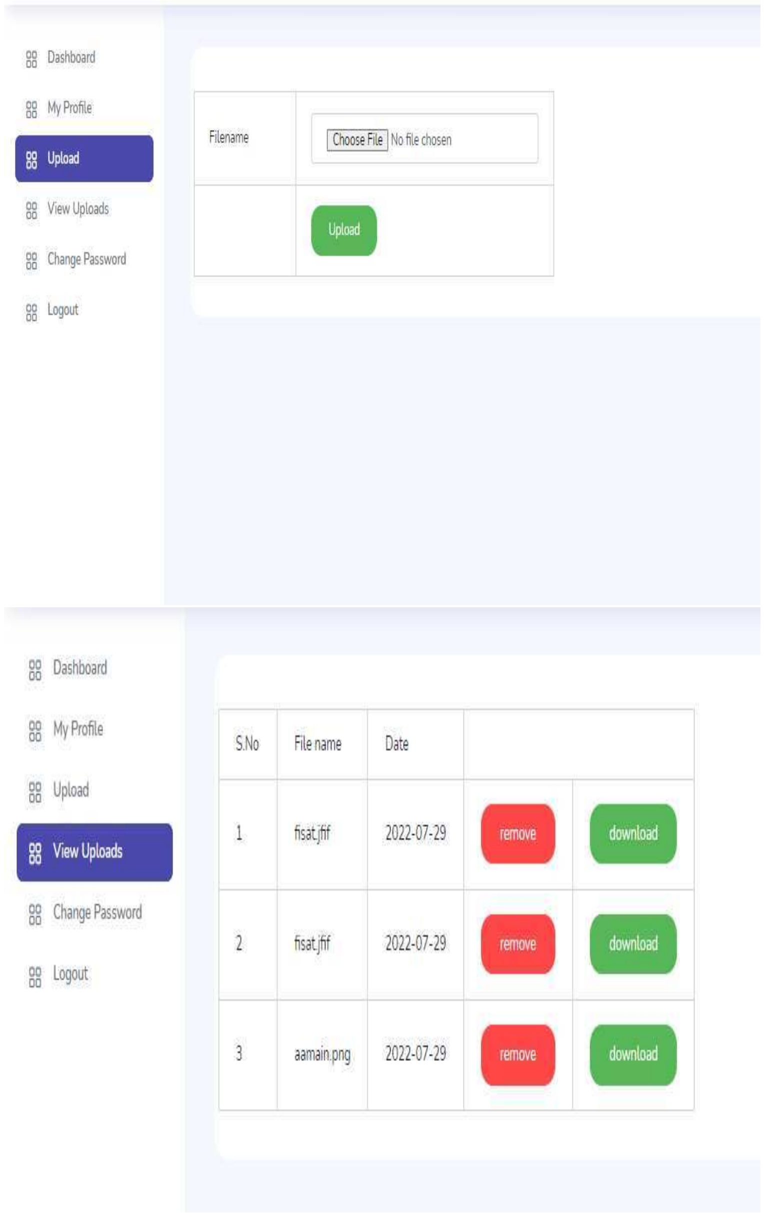


Figure 3.1 : Architecture Diagram

3.2 Dataset

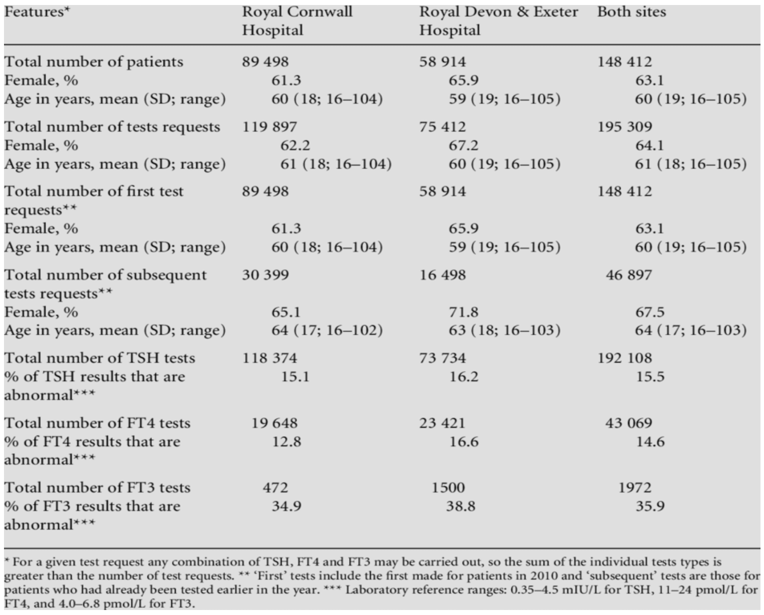
In this project, we will not use any datasets to develop. With the intent of that, we are using the data uploaded by the data provider as well as the user. The data can be an image (jpg, png,jifi etc.) file or be a pdf file.



S.No	File name	Date		
1	fisat.jfif	2022-07-29	remove	download
2	fisat.jfif	2022-07-29	remove	download
3	aamain.png	2022-07-29	remove	download

Data uploading by dataprovider

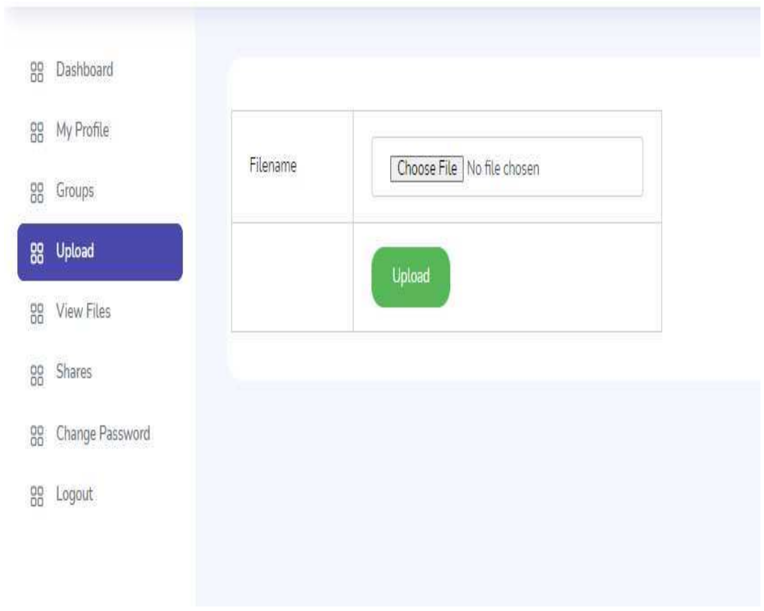
data sample for the doctor's attribute:

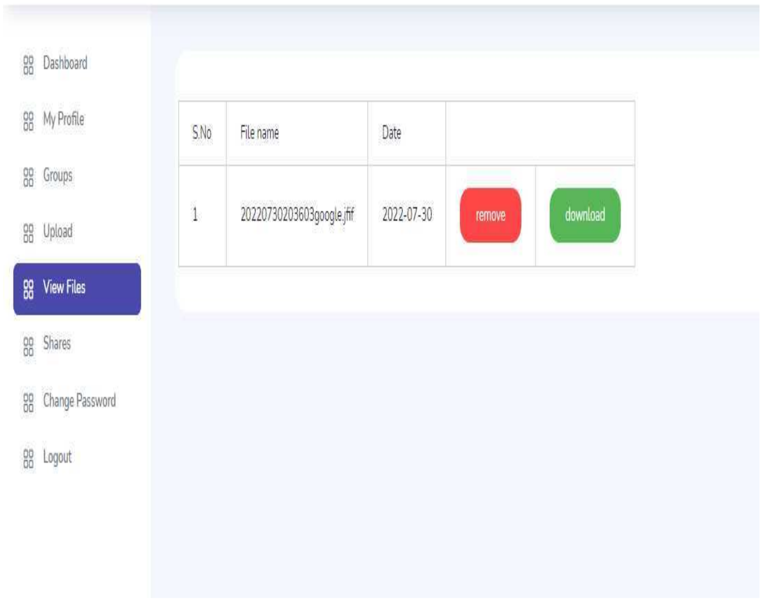


Features*	Royal Cornwall Hospital	Royal Devon & Exeter Hospital	Both sites
Total number of patients	89 498	58 914	148 412
Female, %	61.3	65.9	63.1
Age in years, mean (SD; range)	60 (18; 16–104)	59 (19; 16–105)	60 (19; 16–105)
Total number of tests requests	119 897	75 412	195 309
Female, %	62.2	67.2	64.1
Age in years, mean (SD; range)	61 (18; 16–104)	60 (19; 16–105)	61 (18; 16–105)
Total number of first test requests**	89 498	58 914	148 412
Female, %	61.3	65.9	63.1
Age in years, mean (SD; range)	60 (18; 16–104)	59 (19; 16–105)	60 (19; 16–105)
Total number of subsequent tests requests**	30 399	16 498	46 897
Female, %	65.1	71.8	67.5
Age in years, mean (SD; range)	64 (17; 16–102)	63 (18; 16–103)	64 (17; 16–103)
Total number of TSH tests	118 374	73 734	192 108
% of TSH results that are abnormal***	15.1	16.2	15.5
Total number of FT4 tests	19 648	23 421	43 069
% of FT4 results that are abnormal***	12.8	16.6	14.6
Total number of FT3 tests	472	1500	1972
% of FT3 results that are abnormal***	34.9	38.8	35.9

* For a given test request any combination of TSH, FT4 and FT3 may be carried out, so the sum of the individual tests types is greater than the number of test requests. ** 'First' tests include the first made for patients in 2010 and 'subsequent' tests are those for patients who had already been tested earlier in the year. *** Laboratory reference ranges: 0.35–4.5 mIU/L for TSH, 11–24 pmol/L for FT4, and 4.0–6.8 pmol/L for FT3.

The data uploaded by the user is used to store in the private cloud, which is developed on the Microsoft Azure platform, and the data uploaded by the data provider is used to store in the public cloud, which is developed on the Amazon S3 platform.

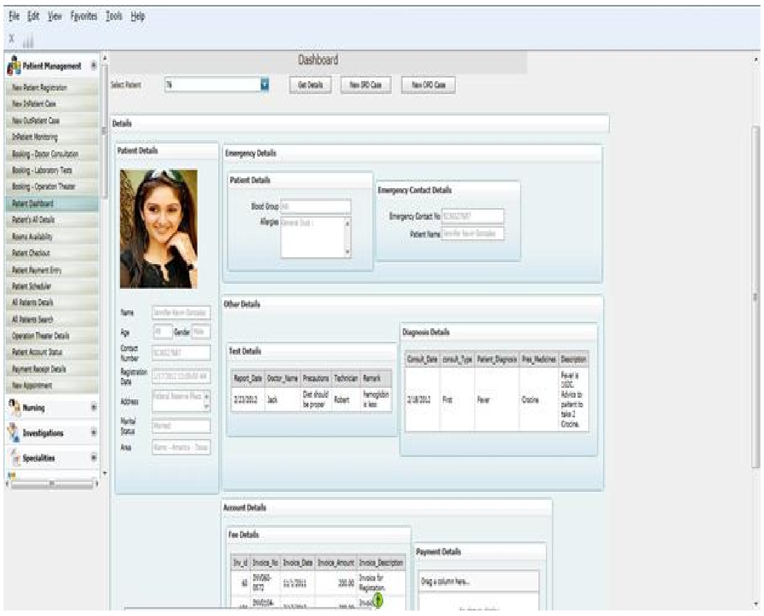




S.No	File name	Date		
1	20220730203603google.jfif	2022-07-30	remove	download

Data uploading by dataprovider

user data sample with doctor attribute:



Select Patient: 76

Blood Group:
Allergies: General Dust :

Emergency Contact No:
Patient Name: Jennifer Kevin Gonzalez

Name: Jennifer Kevin Gonzalez
Age:
Gender: Male
Contact Number:
Registration Date:
Address: Federal Reserve Plaza
Marital Status: Married
Area: Name - America - Texas

Report_Date	Doctor_Name	Precautions	Technician	Remark
2/23/2012	Jack	Diet should be proper	Robert	hemoglobin is less

Consult_Date	consult_Type	Patient_Diagnosis	Pres_Medicines	Description
2/18/2012	First	Fever	Crocine	Fever is 102C. Advice to patient to take 2 Crocine.

Inv_id	Invoice_No	Invoice_Date	Invoice_Amount	Invoice_Description
60	INV060-0072	11/1/2011	200.00	Invoice for Registration.
[illegible]	INV0104-	[illegible]	[illegible]	[illegible]

3.3 Algorithms

3.3.1 SHA-1 Algorithm

SHA-1, or Secure Hash Algorithm 1, is a cryptographic hash function which takes an input and produces a 160-bit (20-byte) hash value. This hash value is called the message digest.

The Hash function is used in cryptography to secure a message by encoding it. It takes input of any length and maps it into a fixed size. Every message should have a unique hash value. A small change in the message should extensively change the hash value. Further, it goes without saying that the same message should always result in the same hash value.

Hashlib library

This module provides access to many hash algorithms like The MD5 which is defined in RFC 1321, is a hash algorithm to turn inputs into a fixed 128-bit (16 bytes) length of the hash value, SHA-1 where SHA, (Secure Hash Algorithms) are set of cryptographic hash functions defined by the language to be used for various applications such as password security etc. Some variants of it are supported by Python in the “hashlib” library. These can be found using algorithms guaranteed function of hashlib etc. Hashlib helps in generating a message digest of the original message. A message digest is a cryptographic hash value of the message produced using hash algorithms.

To use this, simply import it using:

import hashlib

The Python hashlib module is an interface for hashing messages easily. This contains numerous methods which will handle hashing any raw message in an encrypted format.The core purpose of this module is to use a hash function on a string, and encrypt it so that it is very difficult to decrypt it.Typically, an encrypted string is long enough such that it is almost impossible to get back the original string. **hexdigest()** function: This method returns the hashed message in hexadecimal format.

Syntax: **string.hexdigest()**

3.4 Modules

There are three modules in this project.They are Data Provider module,User Module, Attribute Authority module,Cloud Module.

3.4.1 Data Provider Module

In the data provider module, the data provider will upload files with corresponding attributes. These files can be seen in the attribute authority (Admin) page. Attribute authorities will share it with the corresponding users.The files uploaded by the data provider will be stored in the public cloud after deduplication by the private cloud, which is on the drive of the computer.The Amazon S3 platform is used to provide the public cloud to store the files uploaded by the data provider. After deduplication in the private cloud, the data provider's files are stored in the public cloud.The functions in data provider modules are:

- **Data provider registration**

  The new data provider can register with their corresponding attribute.

- **Data provider Upload**

  The new data provider can see their profile after logging in. The Upload option will allow the data provider to upload the data, which is then used to share it among the users. Users of a group that satisfies the corresoping attribute can download this shared file.

- **Data provider View Upload**

  This option will allow the data provider to see all the files you upload. It also consists of a remove and download option. Here, the remove button will help to remove that particular file from uploading and the download button helps to download the data whenever it is needed by the data provider.

3.4.2 User Module

In the user module, users can upload data to be stored in the private cloud. This private cloud is built on Microsoft Azure.Likewise, attribute authority will create a group of users based on the matching attributes and share the corresponding file with the group. The users in that particular group can download the file.The attributes of the uploaded file and those of the users in the group should be equal.The functions in user modules are:

- **User registration**
  The user can register with their corresponding attribute.
- **User Groups**
  This option allows the users to know which groups they are a member of and can access the files.
- **User Upload**
  This allows the user to upload their personal files and store them on the Microsoft Azure cloud platform, where we create the private cloud. This private cloud is provided to store the personal files of users.
- **User ViewFiles**
  This option will allow the users to see all the personal files they have stored on the private cloud.
- **User Shares**
  This allows the users to download the files which are shared among the users in which they are a member of that particular group.

3.4.3 Attribute Authority(Admin) module

In this module, attribute authority can be see all the users and data providers. It can see all the uploaded files by the data providers.It can create groups to share files.Groups are created in such a way that all the users in the group have the same attribute. They can share a particular file with this group if their attributes are equal.The functions in admin module are:

- **Group creation**
  This allows the admin to create groups which consist of members of the same attribute.

- **Providers**
  This will help the admin to see all the data providers in one place. Details of data providers can also be seen there.

- **Users**
  This option will help the attribution authority see all the users.It will consist of all the details among the users.

- **View Uploads**
  This will shows the all all uploaded files name along with the data provider and their attribute.Their is a share option in which it can be used by the attribute authority to share among a group.

- **View Groups**
  This will show all the created groups. More and add members options will be available. The more option is used to see all the members of that group . The add members option is used to add all members to that group.

3.4.4 Cloud module

In this project it uses a hybrid cloud architecture.Hybrid cloud is the combination of public cloud and private cloud.Public cloud is a cloud deployment model where computing resources are owned and operated by a provider and shared across multiple tenants via the Internet and private cloud is defined as computing services offered either over the Internet or a private internal network and only to select users instead of the general public.

Platforms like Microsoft Azure, Amazon S3 and computer drives are used to create public and private clouds. Amazon S3 is a cloud platform that is used to store the data inside the bucket. It is the public cloud used to store the data uploaded by the data provider. Deduplication of data is implemented on the private cloud, which is the drive of the computer. After the deduplication, the uploaded file is either stored in the public cloud or it restores the index of already existing data in the public cloud.Microsoft azure is used to store the data uploaded by the user. It's an additional storage that we provide in the private cloud. It is used by the user to store their personal data and use it whenever they need it.

3.5 Tools and FrameWork

Secure deduplication of encrypted data in the cloud is a cloud-based application. The basic concept of this work is to use hashing techniques to effectively remove the duplication of data. Here we use HTML and CSS to describe the structure of an application and the presentation of Web pages, including colors, layout, and fonts, respectively.The project uses the PyCharm IDE to develop the application.

3.5.1 PyCharm

PyCharm is a Python Integrated Development Environment (IDE) providing a wide range of essential tools for Python developers, and it is tightly integrated to create a convenient environment for productive Python and web development.

- PyCharm offers some of the best features to its users and developers in the following aspects
- Code completion and inspection
- Advanced debugging
- Support for web programming and frameworks such as Django and Flask

3.5.2 Python

The backend is implemented by python , where Python is a computer programming language used to build websites, software, etc. The server used is a python flask and developed in pycharm. Python is an interpreted, interactive, object-oriented programming language. It incorporates modules, exceptions, dynamic typing, very high level dynamic data types, and classes. It supports multiple programming paradigms beyond object-oriented programming, such as procedural and functional programming. Python combines remarkable power with very clear syntax. It has interfaces to many system calls and libraries, as well as to various window systems, and is extensible in C or C++. It is also usable as an extension language for applications that need a programmable interface. Finally, Python is portable: it runs on many Unix variants including Linux and macOS, and on Windows.

3.5.3 Flask

Flask is a micro-web framework written in Python. It was developed by Armin Ronacher, who led a team of international Python enthusiasts called Poocco. Flask is based on the Werkzeg WSGI toolkit and the Jinja2 template engine.Both are Pocco projects. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions.

However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

3.5.4 Amazon S3

Amazon S3 is the platform that we used here to create the public cloud to store the data uploaded by the data provider. After deduplication in the private cloud, data is uploaded to the public cloud.Amazon S3 is a platform that provides cloud services like storage, data protection by encryption techniques, etc.

Here we use Amazon's S3 platform for the storage purpose. The buckets that we create on the Amazon S3 platform are used to store the data. We get a public cloud storage platform from the Amazon S3.

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. Customers of all sizes and industries can use Amazon S3 to store and protect any amount of data for a range of use cases, such as data lakes, websites, mobile applications, backup and restore, archive, enterprise applications, IoT devices, and big data analytics. Amazon S3 provides management features so that you can optimize, organize, and configure access to your data to meet your specific business, organizational, and compliance requirements. By default, S3 buckets and the objects in them are private. You have access only to the S3 resources that you create. To grant granular resource permissions that support your specific use case or to audit the permissions of your Amazon S3 resources,like S3 Block Public Access ,AWS Identity and Access Management (IAM) can use.

3.5.5 Microsoft Azure

Microsoft Azure is a platform that provides cloud services.In this project, we use Microsoft Azure to create a private cloud to store the personal data of users. Users can download and use it later whenever they need.Amazon provides users with the flexibility to use their preferred tools and technologies. In addition, Azure offers 4 different forms of cloud computing: infrastructure as a service (IaaS), platform as a service (PaaS), software as a service (SaaS) and serverless.In this project, we use Azure blob storage. It allows unstructured data to be stored and accessed at a massive scale in block blobs. It also supports Amazon S3 provides features for auditing and managing access to your buckets and objects.

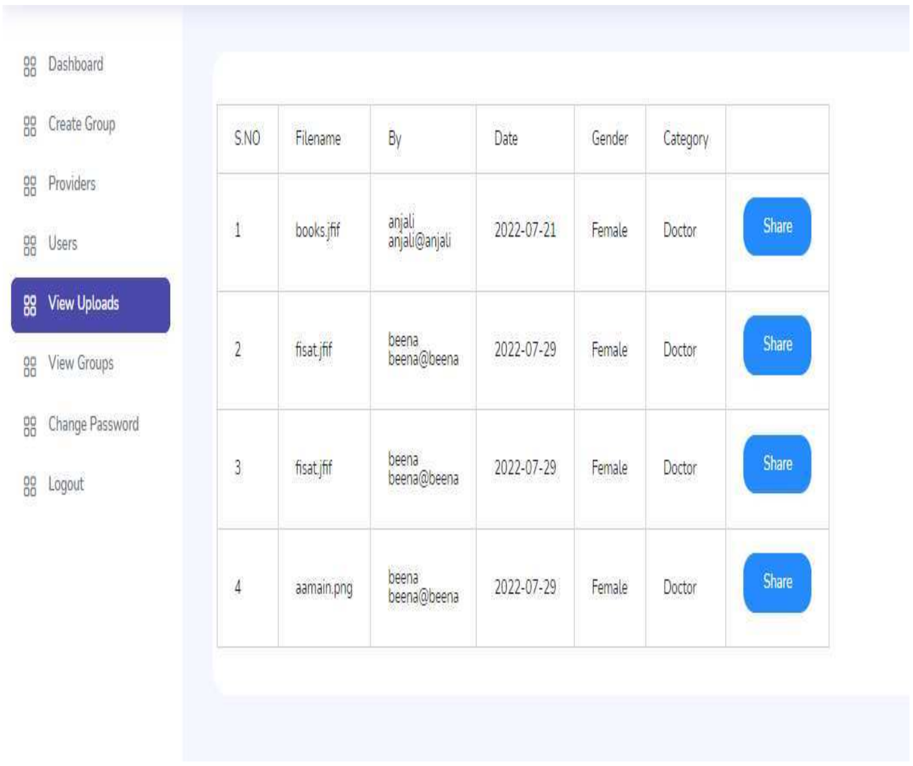
Azure Data Lake Storage Gen2 for enterprise big data analytics solutions. It is mainly used when you want your application to support streaming and random access scenarios and want to be able to access application data from anywhere, and also when you want to build an enterprise data lake on Azure and perform big data analytics.

Chapter 4

RESULT ANALYSIS

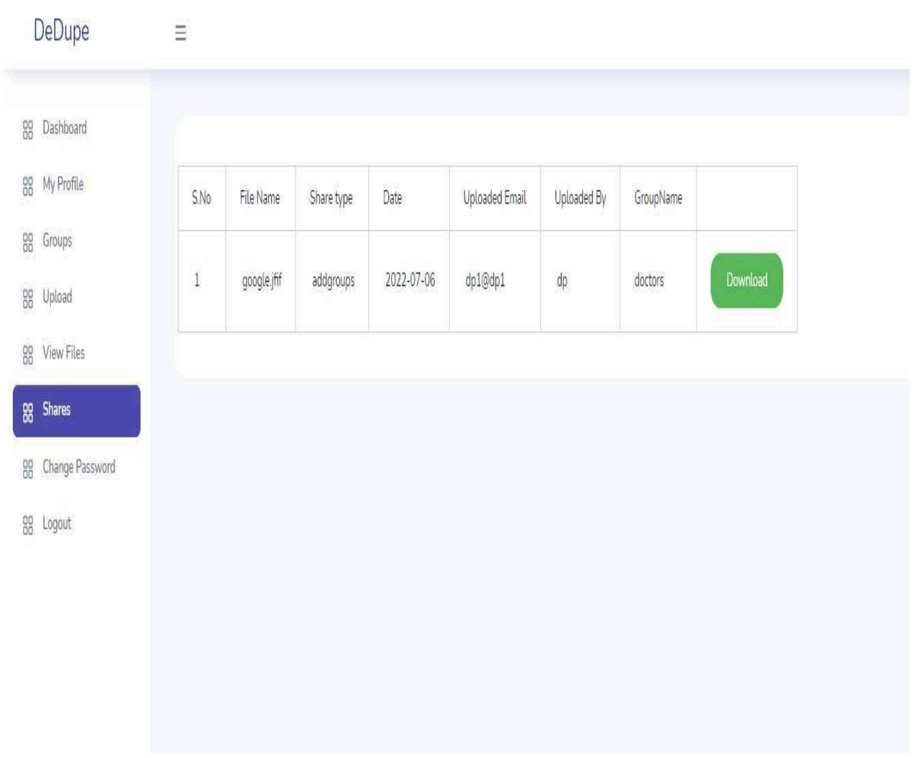
The application was successfully completed on time. The data which is repeatedly uploaded is securely deduplicated in the private cloud and restores the index of already stored files in the public cloud . The data, which is newly uploaded, is securely deduplicated in the private cloud and stored in the public cloud. Data deduplication is done by the hybrid cloud, where a private cloud is used for data deduplication and a public cloud is used for data storage.

The data stored by the data provider is initially stored on the private cloud on drive. It will check the existence of uploaded files in the public cloud using hashing. It restores the index of an already existing file in the public cloud and stores the file that is not present in the public cloud. A separate private cloud is provided for the users to store their personal data. The proposed work helps with the secure deduplication of data using hybrid clouds and attribute-based storage and data sharing among users. Attribute authority, create a group of members, and share the file that matches the attribute successfully. The members of the group successfully downloaded the shared file.They are given below



S.NO	Filename	By	Date	Gender	Category	
1	books.jfif	anjali anjali@anjali	2022-07-21	Female	Doctor	Share
2	fisat.jfif	beena beena@beena	2022-07-29	Female	Doctor	Share
3	fisat.jfif	beena beena@beena	2022-07-29	Female	Doctor	Share
4	aamain.png	beena beena@beena	2022-07-29	Female	Doctor	Share

File sharing by admin



S.No	File Name	Share type	Date	Uploaded Email	Uploaded By	GroupName	
1	google.jfif	addgroups	2022-07-06	dp1@dp1	dp	doctors	Download

File downloading by user

Chapter 5

CONCLUSION AND FUTURE SCOPE

5.1 Conclusion

Attribute-based encryption (ABE) has been widely used in cloud computing where data providers outsource their encrypted data to the cloud and can share the data with users possessing specified credentials. On the other hand, deduplication is an important technique to save the storage space and network bandwidth, which eliminates duplicate copies of identical data. However, the standard ABE systems do not support secure deduplication, which makes them costly to be applied in some commercial storage services. In this paper, we presented a novel approach to realize an attribute-based storage system supporting secure deduplication. Our storage system is built under a hybrid cloud architecture, where a private cloud manipulates the computation and a public cloud manages the storage.

Managing encrypted data with deduplication is essential and significant in practice for achieving a successful cloud storage service, primarily for data storage. In this project, we scheduled a practical scheme to manage the space.When a new file uploaded by the data provider its duplication will check using the hashing technique. Thus by reducing the storage usage, cost will also be reduced automatically. To secure the privacy of sensitive data during deduplication, the encryption technique is used to encrypt the data before outsourcing.A separate private is developing for the users to store their data.

5.2 Future Scope

In the future, to improve the secure storage and deduplication of data, it is clear that the hybrid cloud architecture will help. The combination of public and private clouds will help to overcome the issues of deduplication. This concept will help with secure data storage and deduplication of data in the cloud.

In addition, applying the other hashing techniques and other cloud combining platforms will give better comparison results. This study will help the attribute authorities promote better data sharing among the users.

Chapter 6

APPENDIX

6.1 Source Code

6.1.1 dbconnect.py

```
import mysql.connector
def connection():   cn=mysql.connector.Connect(host='localhost'
,user='root',passwd='',db='dedup', port=3306)
cu=cn.cursor(buffered=True)
// cursor is an inbuild class
in mysql cu is the object
return cu,cn
```

6.1.2 sampleazure.py

```
from azure.storage.blob import BlobServiceClient ,BlobClient ,
ContainerClient ,\_\_version\_\_
connectionstring=”DefaultEndpointsProtocol=https;
AccountName=sample123lik;
AccountKey=jLw4XvM5sS7SWou3mVBqWOYyC
SrrcHin4AqXWkScaLAAC+1u/Dh/i623g0156
6udVavZ0sRSndtk+AStx9AUmA==;Endpoint
Suffix=core.windows.net”
import uuid
blob\_service\_client=BlobServiceClient.from\_connection
\_string(connectionstring)containername=str(uuid.uuid4())
# container\_client=blob\_service\_client.create\_container
(containername)blob\_client=blob\_service\_client.get
\_container\_client(container=”sample”)
with open(”aaanew.sql”,”wb”) as h:k=blob\_client.download
\_blob (”a.sql”).readall()
h.write(k)
print(k)
```

6.1.3 log.py

from dbconnect import connection

from flask import Flask, redirect,request,session, url_for

from flask.templating import render_template

import time

from pip._vendor.requests.packages.urllib3.util.connection import select from builtins import len import hashlib

import os

import boto

import boto.s3

import sys

from boto.s3.key import Key

from fileinput import filename

import array

from flask.helpers import make$_{r}esponse, send_{f}ile$

$app = Flask(\_\_name\_\_)$

$app.secret\_key = "hai"$

$@app.route('/')$

$@app.route('/login')$

$defmain():$

$returnrender\_template('login.html')$

$staticpath = "G : \_project\_project "$

@app.route('/log')

def lg():

session.clear()

return redirect(url_for('main'))

@app.route('/regi')

def rg():

return render_template('registration.html')

@app.route('/frg',methods=['POST'])

def registr():

c=request.form['name']

e=request.form['email']

f=request.form['radio']

g=request.form['date']

h=request.form['address']

i=request.form['mobile']

j=request.form['password']

l=request.form['select']

photo=request.files['image']

exten = time.strftime("

photo.save(staticpath +"

photos

" +exten+ ".jpg")

path = "/static/photos/" +exten+ ".jpg"

qry="insert into registration(name,image,email,gender,

date$_o f_b irth, address, mob, password,$

$category, type) value('" + c + "',' " + path + "','$

$" + e + "',' " + f + "',' " + g + "',' " + h + "',' " + i + "',$

$'" + j + "',' " + l + "',' user')"$

print(qry)

cu,cn= connection()

cu.execute(qry)

cn.commit()

return render_template('login.html')

@app.route('/get',methods=['POST'])

def check():

```
a=request.form['name']
b=request.form['pass']
cu,cn=connection()
qry="select *  from registration where email='"+a+"' and password='"+b+"'"
cm=cu.execute
res=cu.fetchone()
type=session['type']
if res is not None:
session['uid']=str(res[0])
session['type']=str(res[10])
type=session['type']
if type=='admin':
return admin()
elif type=='provider':
return provider()
elif type=='user':
return render_template('userhome.html')
else:
return "¡script¿alert('Invalid username or password');window.location='/'¡/script¿"


    @app.route('/uplod')
    def up():
return render_template('fileupload.html')
    @app.route('/uplnew′,methods = [′POST′])
defupl_new() :
fromdatetimeimportdatetime
filename = datetime.now().strftime("
cu,cn = connection()
ud = session[′uid′]
```

f = request.files['filename']

print(type(f))

filedir =' /static/fileupload/' + f.filenameuploadfilefolderinserted

p = "G :

deduplication_project

deduplication$_p$roject

static

fileupload

" + filename + f.filename

f.save(p)

fromazure.storage.blobimportBlobServiceClient, BlobClient, ContainerClient,

__version__connectionstring = "DefaultEndpointsProtocol = https;

AccountName = sample123lik;

AccountKey = jAAC + lu/Dh/i6x9AUmA ==; EndpointSuffix = core.windows.net"

importuuid

blob_service$_c$lient = BlobServiceClient.from_connection_string(connectionstring)

containername = str(uuid.uuid4())

container_client = blob_service_client.create_container(containername)

blob_client = blob_service_client.get_blob_client

(container = "sample", blob = filename + f.filename)

withopen(p, "rb")ash : k = blob_client.upload_blob(h)print(k)

cu, cn = connection()

hash = "j"

qry = "insertintoupload(filename, uid, hashvalue, date)

values('" + filename + f.filename + "',' " + ud + "',' " + hash + "', CURDATE())"

print(qry)

cu.execute(qry)

cn.commit()

return" < script > alert('Fileuploadedsuccessfully');

$window.location =' /uplod' < /script >$”

```
    import boto3
sessionmk = boto3.Session( aws_access_key_id=’AKIA23KD6ZZB376VCBAB’,
aws_secret_access_key=’LgjzHEdPc3WkTV8LvRSlaIDr3SW724znQUt4fdBE’, )
s3 = sessionmk.resource(’s3’)
s3.meta.client.upload_file(Filename=x, Bucket=’testabcdcrypto’, Key=str(blockid))
cu.execute(qry)
cn.commit()
ids=ids+1
return ”¡script¿alert(’FIle uploaded successfully’);window.location=’/uplod’¡/script¿”
    @app.route(”/sample”)
def sample():
return render_template(”sample.html”)
    @app.route(’/adm’)
def assswa():
return render_template(’index/index.html’)
@app.route(’/srw’)
def sert():
import boto3
AWS_ACCESS_KEY_ID = ”
AWS_SECRET_ACCESS_KEY = ”
bucket_name = AWS_ACCESS_KEY_ID.lower() + ’-dump’
conn = boto3.connect_s3(AWS_ACCESS_KEY_ID,AWS_SECRET_ACCESS_KEY)
bucket = conn.create_bucket(bucket_name,
location=boto.s3.connection.Location.DEFAULT)
testfile = ”replace this with an actual filename”
print (’Uploading (testfile, bucket_name))
k = Key(bucket)
```

```
k.key = 'my test file'
k.set_contents_from_filename(testfile,cb=percent_cb, num_cb=10)
def percent_cb(complete, total):
sys.stdout.write('.')
sys.stdout.flush()
if __name__ ==' __main__':
app.run(debug=True)
```

6.1.4 fileupload.html

```
{% extends ”index/index.html” %}
{% block body %}
<form id=”form1” name=”form1”
enctype=”multipart/form−data”
method=”post” action=”/upl_new”>
<table class=”table table−bordered” >
<tbody>
<tr>
<td width=”150px”>Filename</td>
<td><input type=”file” class=”form−control”
name=”filename” id=”fileField” required></td>
  </tr>
 <tr>
 <td></td>
<td><input class=”btn btn−success”
type=”submit” name=”submit” =”submit”
value=”Upload”></td>
  </tr>
</tbody>
  </table>
</form>
{% endblock %}
```

6.1.5 fileupload.html

```
<html lang="zxx">
<head>
<title>Latest Login Form Responsive
Widget Template :: w3layouts</title>
<!-- Meta tag Keywords -->
<meta name="viewport"
content="width=device-width, initial-scale=1">
<meta charset="UTF-8" />
<meta name="keywords"
content="Latest Login Form Responsive Widget,
Login form widgets, Sign up Web forms ,
Login signup Responsive web form,
Flat Pricing table,Flat Drop downs,
Registration Forms,News letter Forms,Elements"
/>
<script>
addEventListener("load", function () {
setTimeout(hideURLbar, 0);
}, false);
function hideURLbar() {
window.scrollTo(0, 1);
}
</script>
<!-- Meta tag Keywords -->
<!-- css files -->
<link rel="stylesheet"
href="/static/web/css/style.css"
```

```
type="text/css" media="all" />
<!-- Style-CSS -->
<link href="/static/web/css/font-awesome.min.css"
rel="stylesheet">
<!-- Font-Awesome-Icons-CSS -->
<!-- //css files -->
<!-- web-fonts -->
<link href="//fonts.googleapis.com/css?
family=Source+Sans+Pro:
200,200i,300,300i,400,400i,600,600i,700,700i,900,900i
&amp;subset=cyrillic,
cyrillic-ext,greek,greek-ext,latin-ext,vietnamese"
 rel="stylesheet">
<!-- //web-fonts -->
</head>
<body>
<div class="main-bg">
<!-- title -->
<h1>DEDUPLICATION</h1>
<!-- //title -->
<!-- content -->
<div class="sub-main-w3">
<div class="bg-content-w3pvt">
<div class="top-content-style">
<img src="/static/web/images/user.jpg" alt="" />
</div>
<form action="/get" method="post">
<p class="legend">Login Here
<span class="fa fa-hand-o-down">
```

```
</span></p>
<div class="input">
<input type="email" placeholder="Email"
name="name" required />
<span class="fa fa-envelope"></span>
</div>
<div class="input">
<input type="password" placeholder="Password"
name="pass" required />
<span class="fa fa-unlock"></span>
</div>
<button type="submit" class="btn submit">
<span class="fa fa-sign-in"></span>
</button>
<br>
<ul>
<a href="regi">New User? Signup </a>
</ul>
<ul>
<a href="provider_reg">New Provider? Signup</a>
</ul>
</form>
</div>
</div>
</div>
</body>
</html>
```

6.2 Screen Shots

Figure 6.1: Home Screen

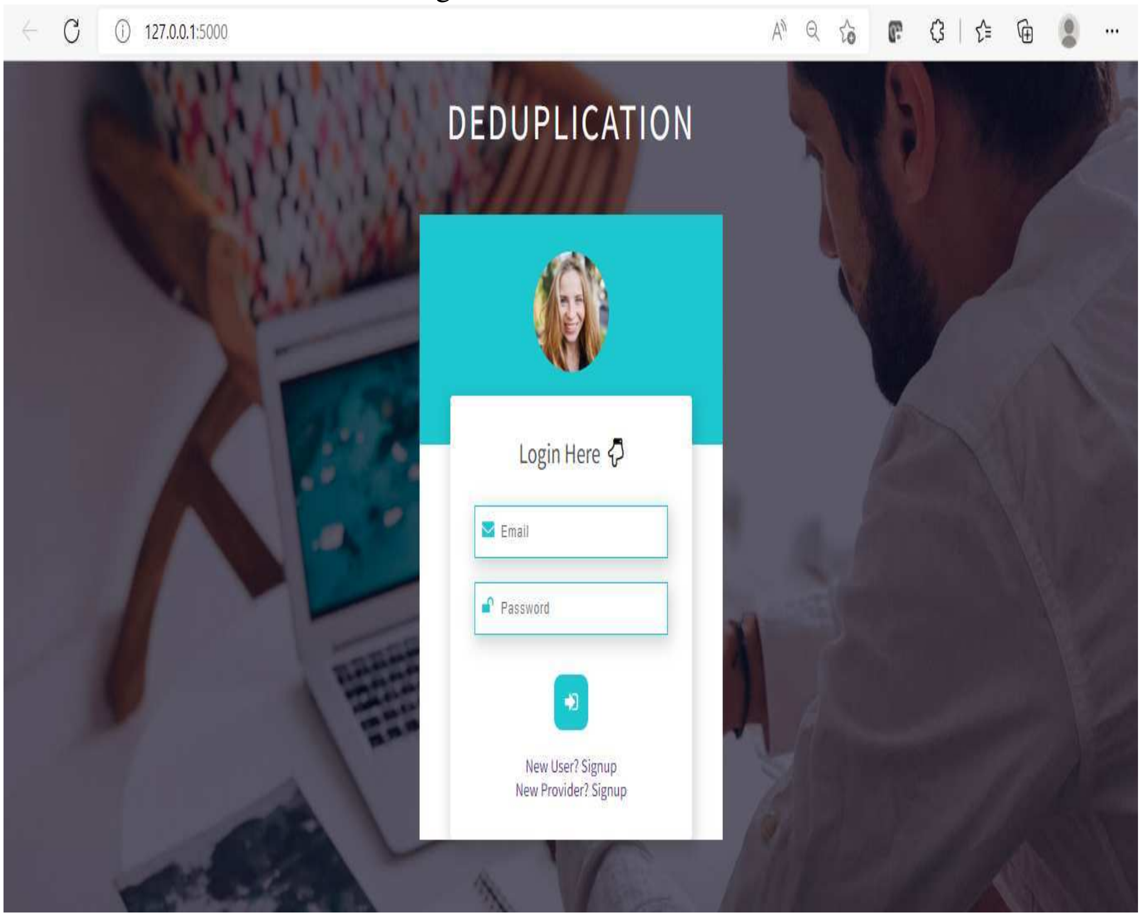
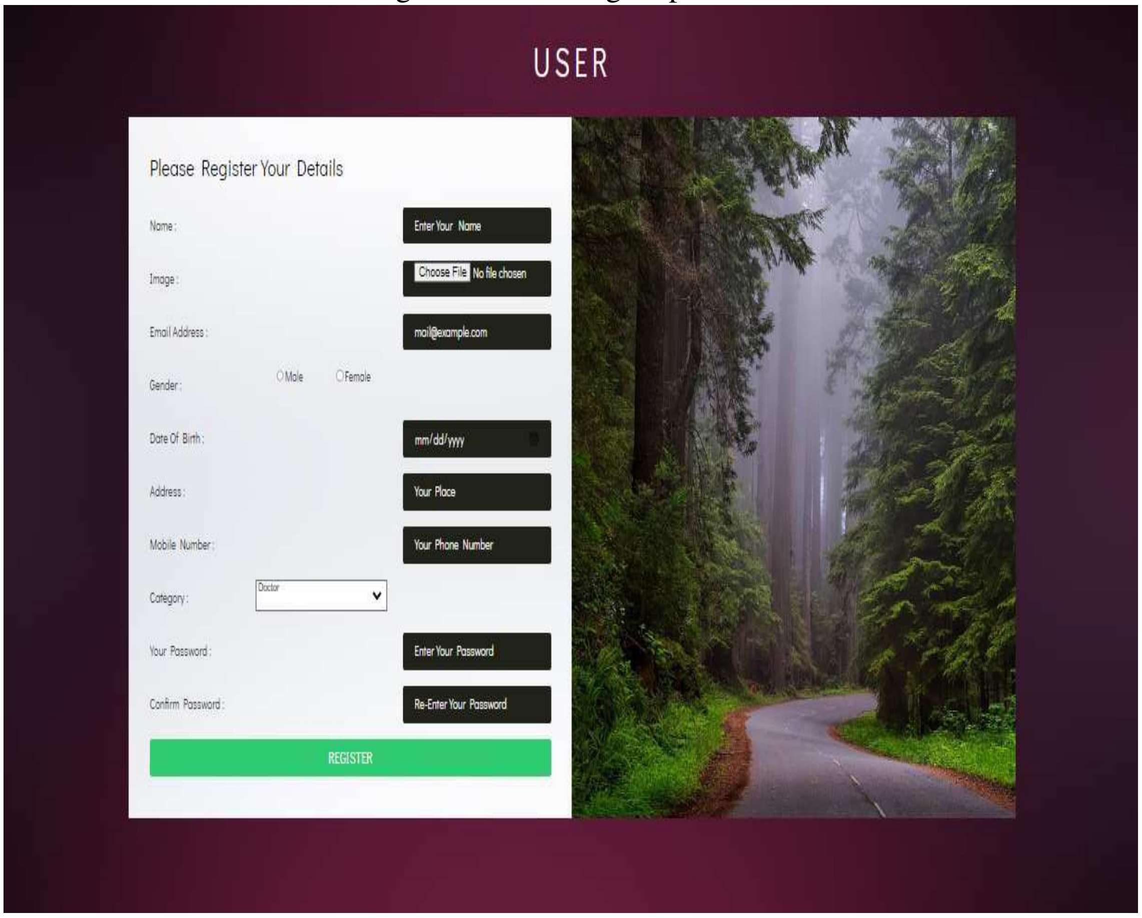


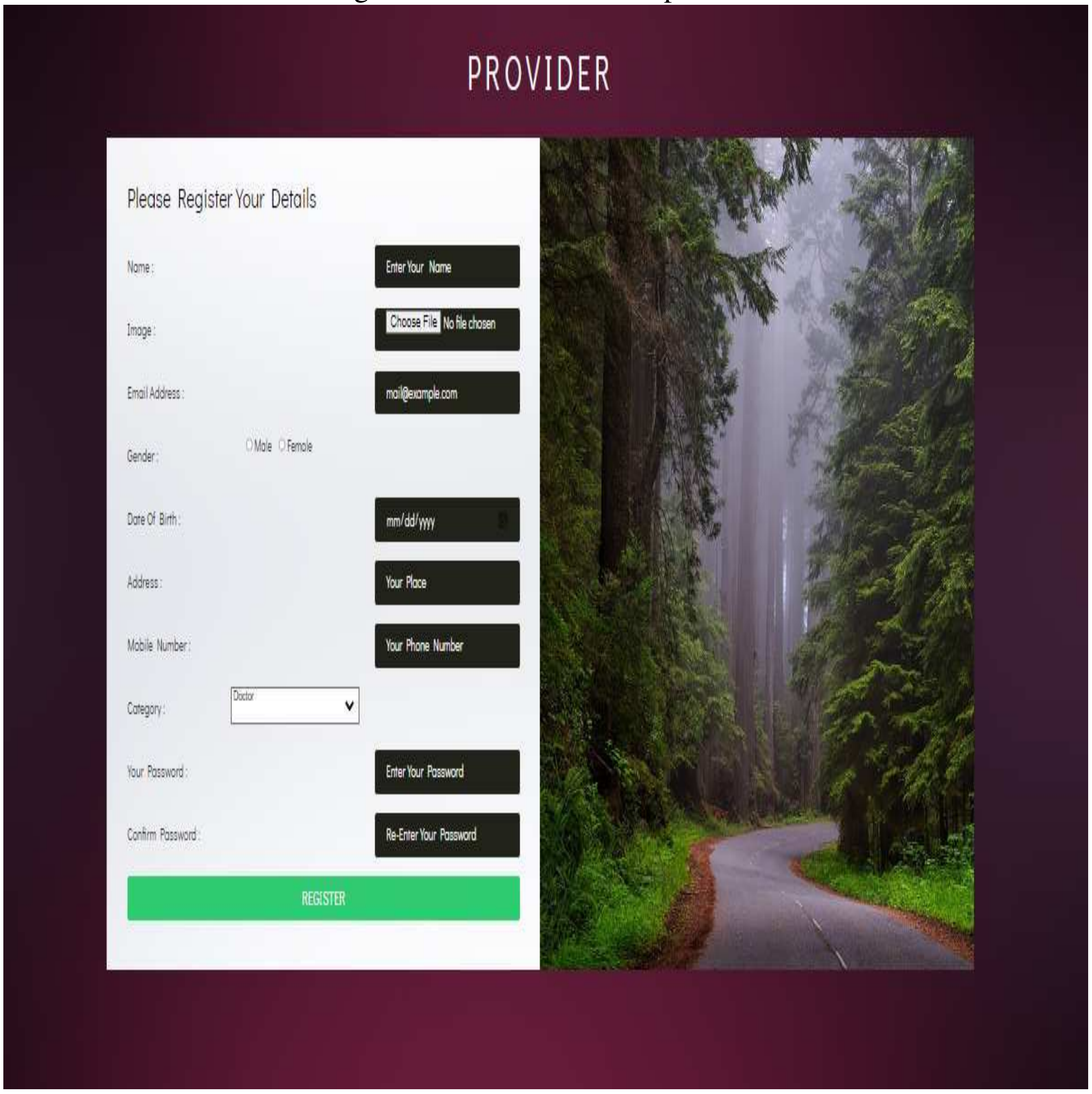
Figure 6.2: UserSign UpScreen


USER

Please Register Your Details

Name :	Enter Your Name
Image :	Choose File No file chosen
Email Address :	mail@example.com
Gender :	○Male ○Female
Date Of Birth :	mm/dd/yyyy
Address :	Your Place
Mobile Number :	Your Phone Number
Category :	Doctor
Your Password :	Enter Your Password
Confirm Password :	Re-Enter Your Password

Figure 6.3: DataProvider UpScreen



PROVIDER

Please Register Your Details

Name :	Enter Your Name
Image :	Choose File No file chosen
Email Address :	mail@example.com
Gender :	○ Male ○ Female
Date Of Birth :	mm/dd/yyyy
Address :	Your Place
Mobile Number :	Your Phone Number
Category :	Doctor
Your Password :	Enter Your Password
Confirm Password :	Re-Enter Your Password

REGISTER

Figure 6.4: Admin Group Creation

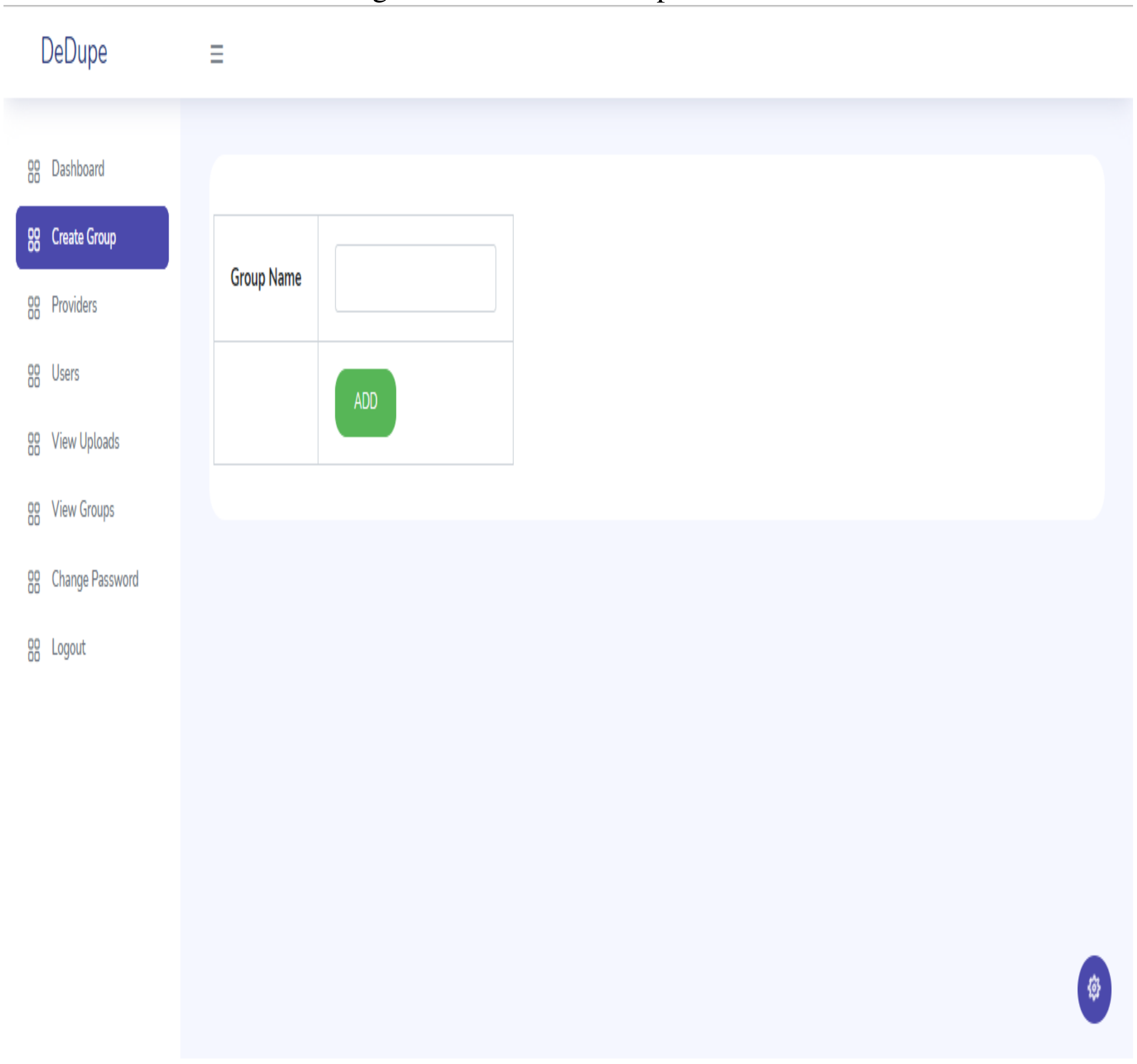
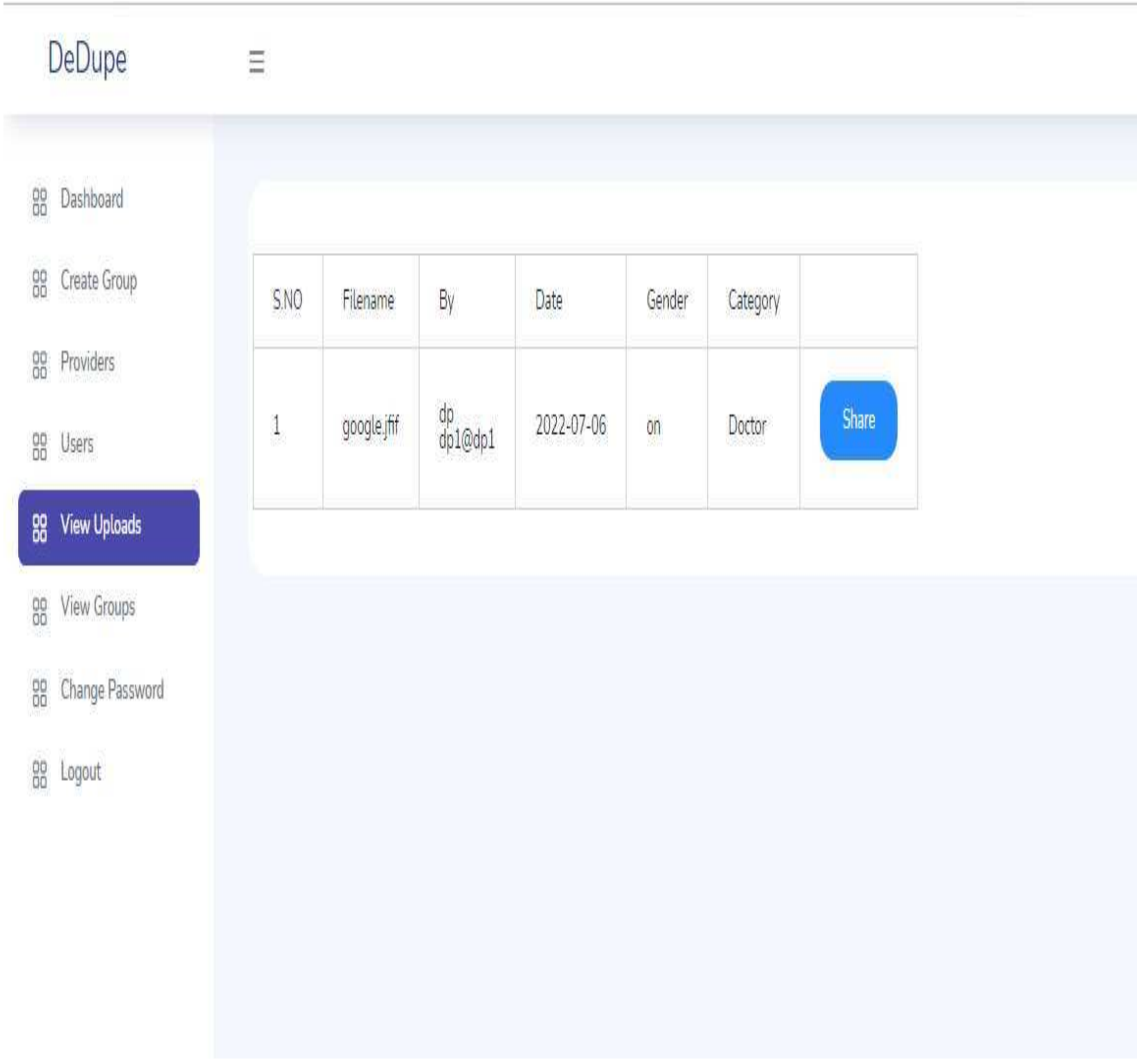
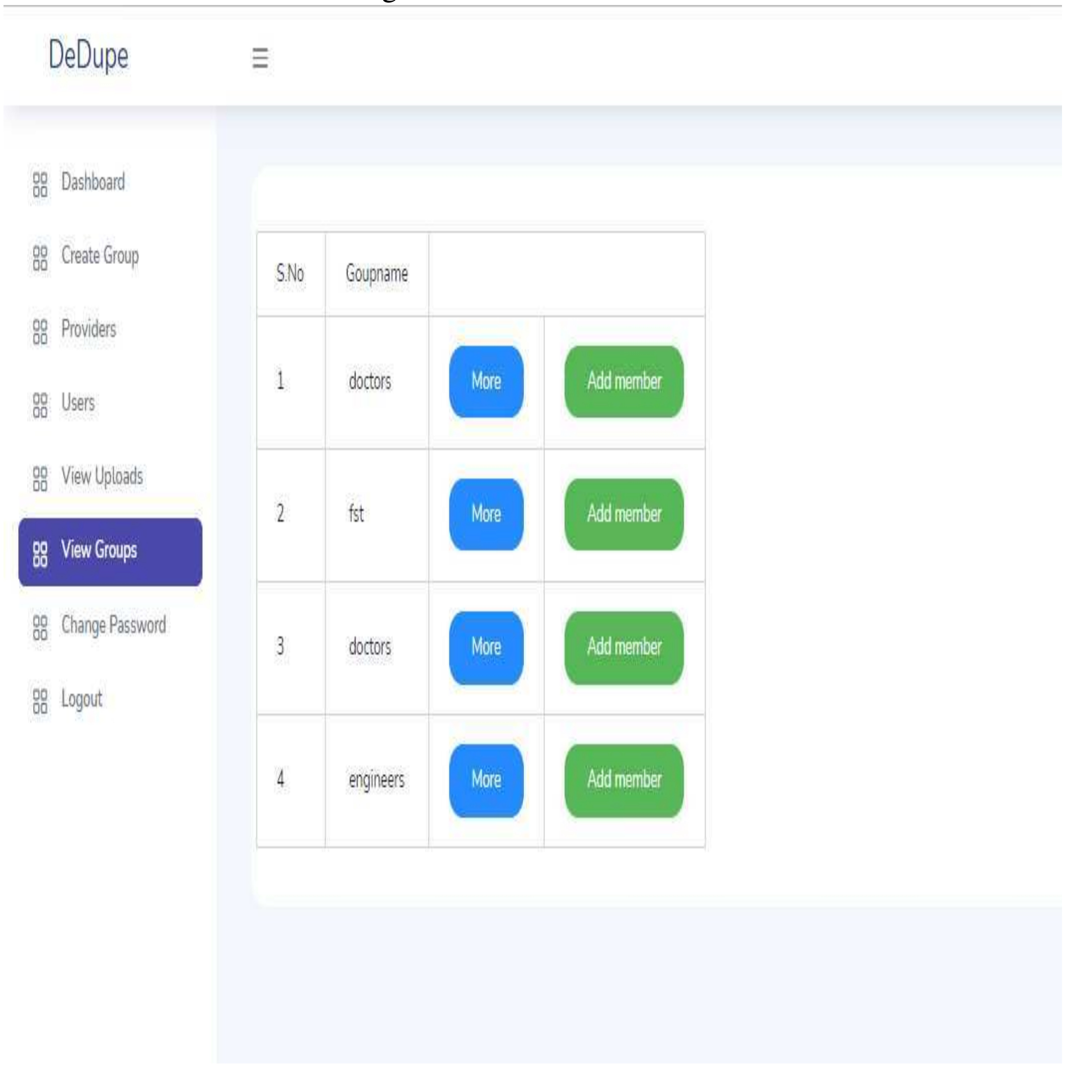


Figure 6.5: Admin Share



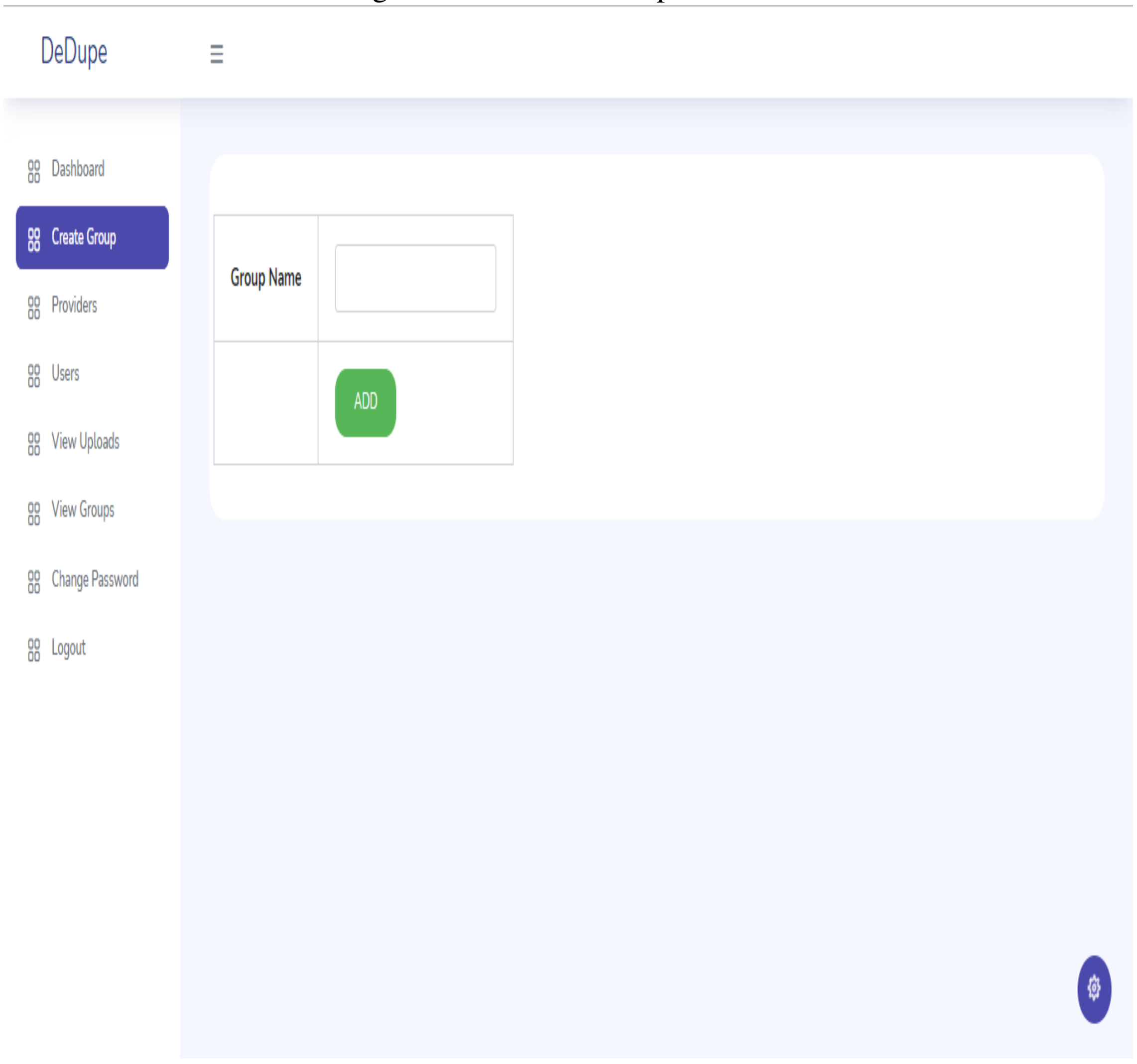
S.NO	Filename	By	Date	Gender	Category	
1	google.jfif	dp dp1@dp1	2022-07-06	on	Doctor	Share

Figure 6.6: Admin Add Members



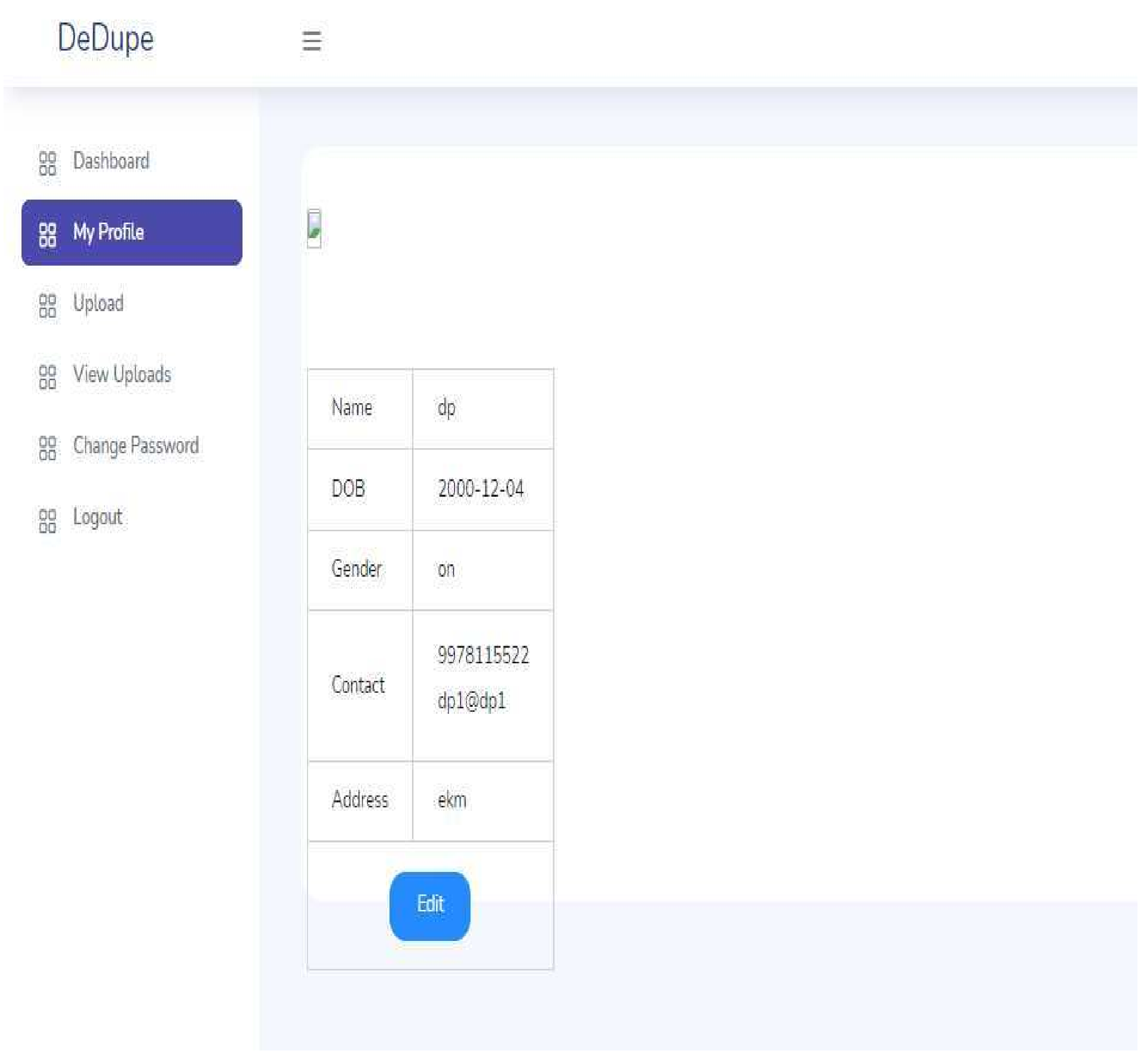
S.No	Goupname		
1	doctors	More	Add member
2	fst	More	Add member
3	doctors	More	Add member
4	engineers	More	Add member

Figure 6.7: Admin Group Creation



Group Name	
	ADD

Figure 6.8: Data Provider View Profile



Name	dp
DOB	2000-12-04
Gender	on
Contact	9978115522 dp1@dp1
Address	ekm

Figure 6.9: Data Provider Upload File

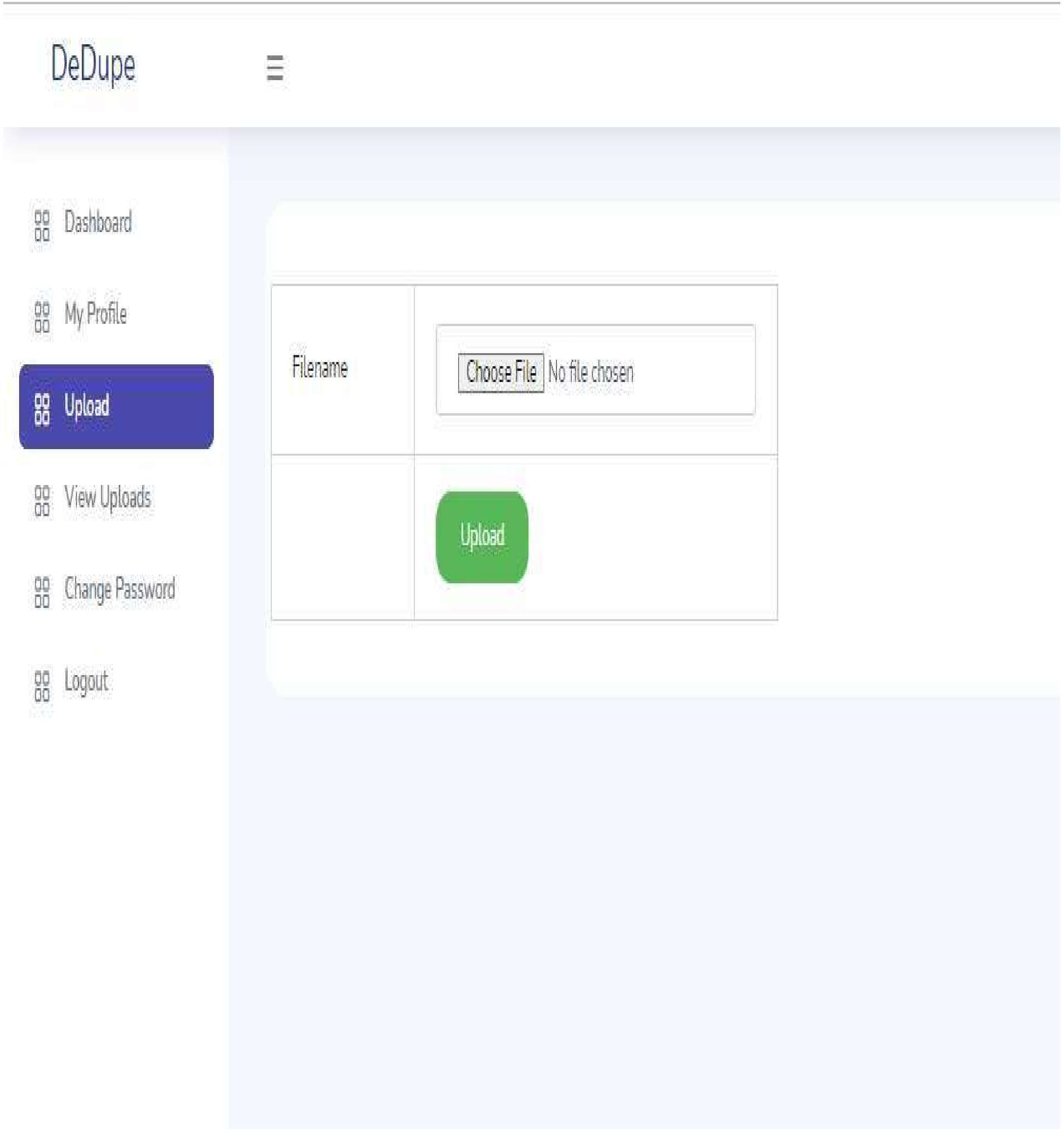
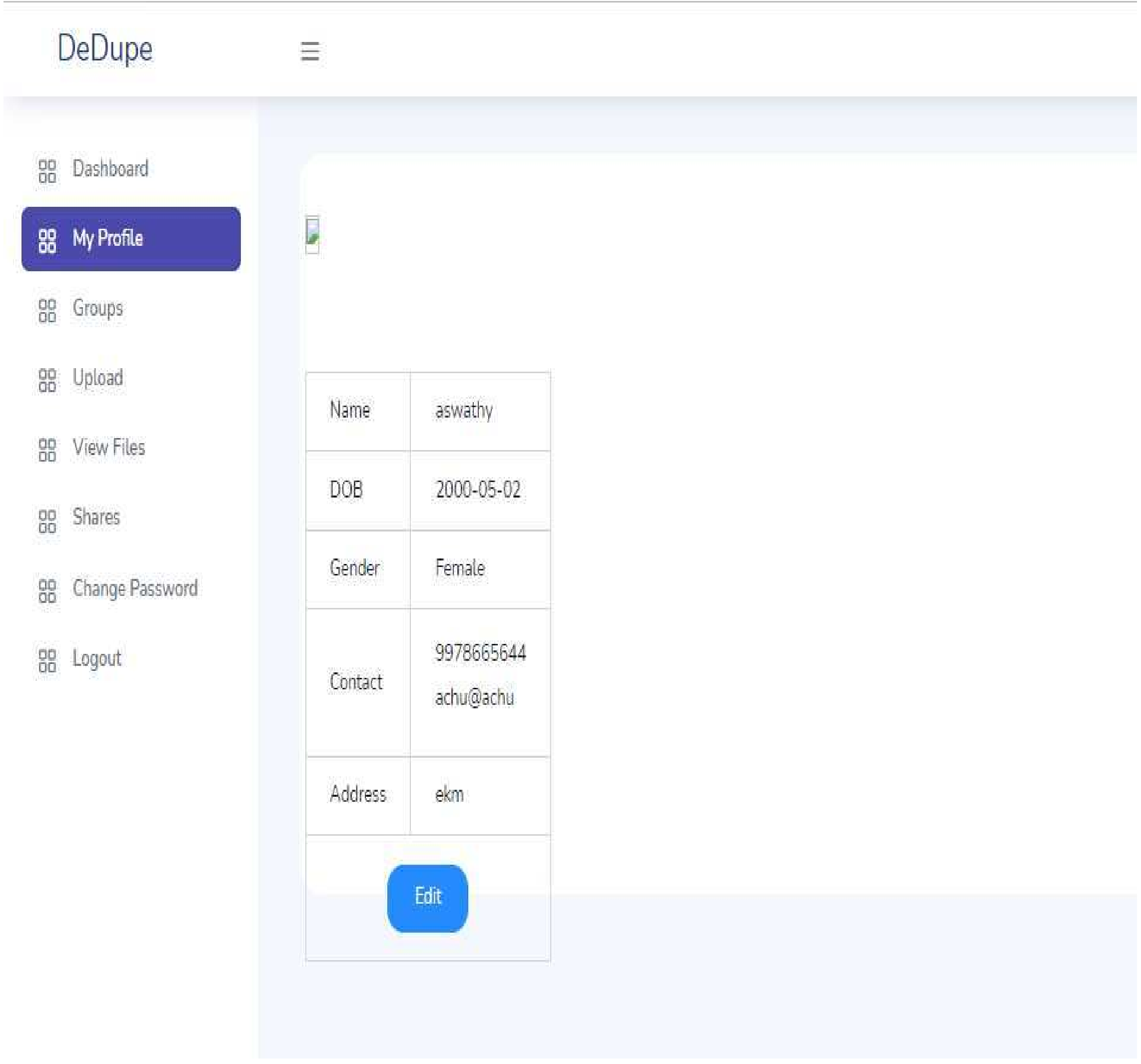
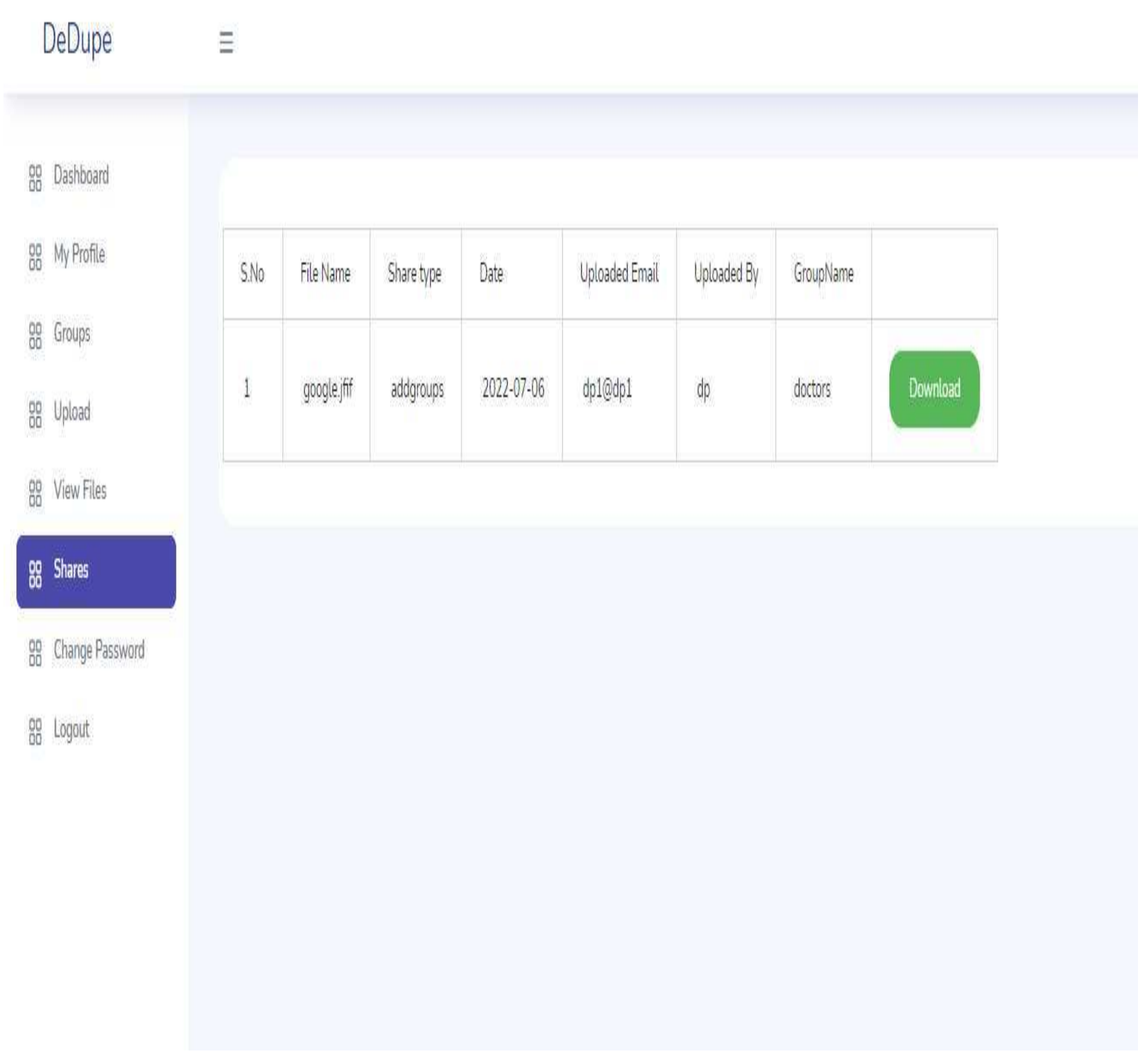


Figure 6.10: User View Profile



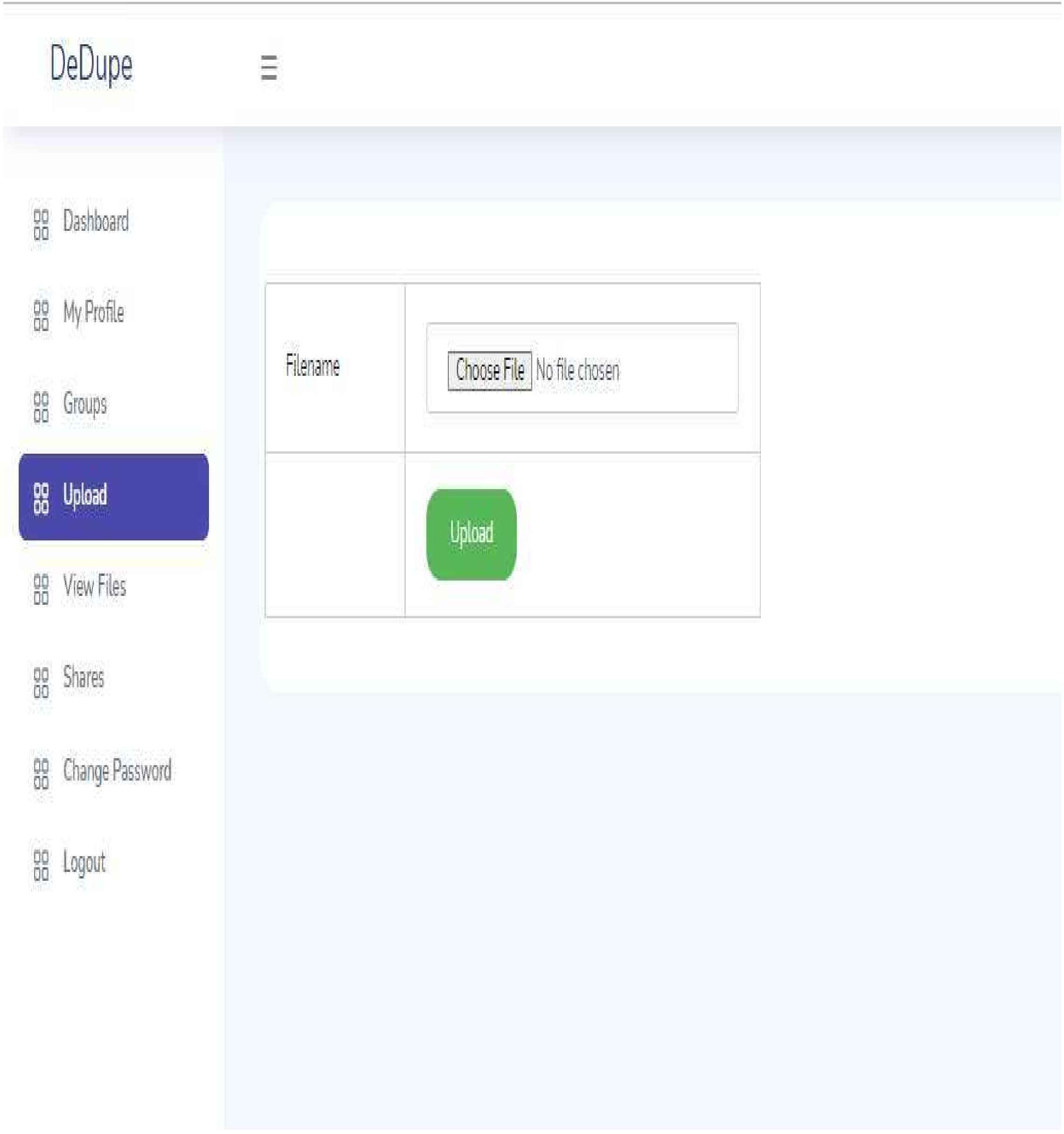
Name	aswathy
DOB	2000-05-02
Gender	Female
Contact	9978665644 achu@achu
Address	ekm

Figure 6.11: User File Download



S.No	File Name	Share type	Date	Uploaded Email	Uploaded By	GroupName	
1	google.jfif	addgroups	2022-07-06	dp1@dp1	dp	doctors	Download

Figure 6.12: User File Upload



Chapter 7

REFERENCES

[1] Hui Cui, Robert H. Deng, Yingjiu Li, and Guowei Wu, “Attribute-Based Storage Supporting Secure Deduplication of Encrypted Data in Cloud”.2017.

[2] S. G. Pundkar, G. R. Bamnote ”Secure Sharing of Personal Records in Cloud using Encryption” Global Journal of Engineering Science and Researches May 2015

[3] J. Li, C. Qin, P. P. C. Lee, and X. Zhang, “Information leakage in encrypted deduplication via frequency analysis,” in Proc. 47th Annu. IEEE/IFIP Int conf. Dependable Syst. Netw., Jun. 2017.

[4] Y. Yang, H. Zhu, H. Lu, J.Weng, Y. Zhang, and K. R. Choo, “Cloud based data sharing with fine-grained proxy re-encryption,” Pervasive and Mobile Computing, vol. 28, pp. 122–134, 2016.

[5] M. Bellare, S. Keelveedhi, and T. Ristenpart, “Message-locked encryption and secure deduplication,” in Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26- 30, 2013. Proceedings, ser. Lecture Notes in Computer Science, vol. 7881. Springer, 2013.

[6] Jiawei Yuan, Shucheng Yu, Secure and Constant Cost Public Cloud Storage Auditing with deduplication IEEE conf. Oct 2013.

[7] Kameswari Bhaskar, R. Sathiyavathi ,Jayashree R, L.Mary Gladence, V. Maria Anu, “A NOVEL APPROACH FOR SECURING DATA DE-DUPLICATION METHODOLOGY IN HYBRID CLOUD STORAGE”

[8] M. Bellare and S. Keelveedhi, “Interactive message-locked encryption and secure deduplication,” in Public-Key Cryptography – PKC 2015