



NotTy : Notes with Added Security

Secure your notes with added security

SCHOOL OF COMPUTER SCIENCE

UNIVERSITY OF PETROLEUM & ENERGY STUDIES

Bidholi Campus, Energy Acres, Dehradun – 248007

Kashish Srivastava

(Enroll No. R134218079 & Sap id 500067405)

Yash Kumar

(Enroll. No. R134218192 & Sap id 500068736)

Vishal Patwal

(Enroll. No. R134218189 & Sap id 500068529)

Abhishek Gupta

(Enroll. No. R134218008 & Sap id 500067804)

Under the guidance of

Ms. Tripti Mishra

Department of Systemics



About Us

NotTy : Notes with Added Security

CONTENTS

- ❖ INTRODUCTION
- ❖ PROBLEM STATEMENT
- ❖ LITERATURE REVIEW
 - ❖ OBJECTIVE
 - ❖ METHODOLOGY
- ❖ SYSTEM REQUIREMENTS
- ❖ ENVIRONMENT FOUNDATION
- ❖ DESIGN AND ALGORITHM
 - ❖ RESULTS
 - ❖ CONCLUSION



Data with low confidentiality concerns might be considered "public" or can cause compromising access to unauthorized users. Data with high privacy concern is viewed as secret and should be kept private to forestall fraud, bargain of records and frameworks, lawful or reputational harm, and other severe consequences. These days clients store their personal information in notes for brisk access, but they are quite prone to be exploited.

NotTy (Notes Security) application with two-factor authentication for enhanced security. It generates an OTP everytime the user opens NotTy along with OTP authentication and hashing will be applied for additional security.



INTRODUCTION

DATA INTEGRITY | CONFIDENTIALITY | AUTHENTIC

PROBLEM STATEMENT

These days people generally tend to write any information or reminders in the form of notes. Sometimes sensitive information such as login credentials are also stored on such applications. Most of the notes applications present on the market offer little to no sense of security to the data stored in them.

This project aims to develop a text editor with simplistic features where the data stored can be verified for its integrity along with two-factor user authentication (User password and OTP)

LITERATURE REVIEW


Predicting user concerns about online privacy

A comparative study of Message Digest 5(MD5) and SHA256 algorithm

Encrypting Data at Rest

IEEE standards for encrypted storage

Discussed these papers theories on concepts of hashing, algorithm, data privacy, encryption etc.



OBJECTIVE & METHODOLOGY

We would like to define an environment. Based on this environment our project presentation is being followed

OBJECTIVE



Working on creating the java application which would keep the confidential data safe as well as integral in nature. Implementation of two-factor authentication along with encryption algorithm would be the most important aspect for the following.

- **Sub Objectives**

- Text editor application in Java
- Implementation of two-factor authentication i.e., Password inquiry and One Time Password inquiry
- Implementing the concepts of hashing to maintain the data integrity and check for breaches if any(MD5).
- Working on the encryption algorithm to provide the user a secure environment.

METHODOLOGY



For this project, the major focus is on developing a secure application for the user for storing their confidential data which would be secured via two-factor authentication and hashing. The entire implementation of this project can be explained in the following points:

- The application has to be developed as in for opening, appending and reading the notes via file handling concept.
- The application after basic development of text editor has to be followed by 2FA implementation.
- The first implementation for authentication is to allow the user to set a password for the user.

- The second implementation for authentication is of the OTP concept.
- For the notes and confidential data, integrity has to be checked of the data stored with the help of hashing concepts.
- At the end for a secure environment an encryption algorithm can be implemented.



SYSTEM REQUIREMENTS

- **Hardware:**

Laptop with i3/i5/i7 processor with 8Gb Ram, 1 Tb Hard-disk

- **Language:**

Java

- **Operating System:**

Ubuntu/ Linux /Windows

- **Resources:**

Packages in Java, Notepad ++

ENVIRONMENT FOUNDATION

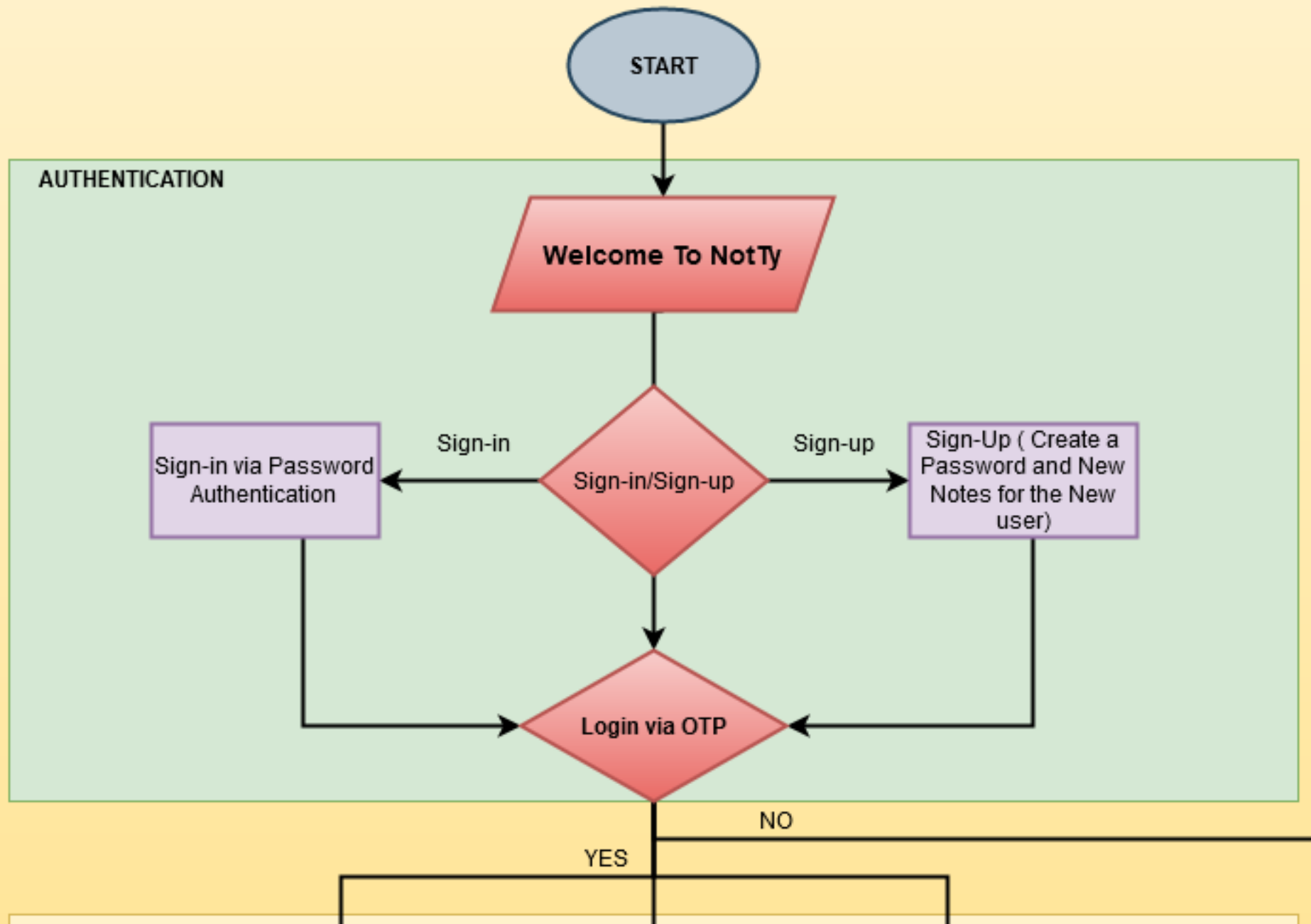
- CLI-Based
- Limited users (5 users as a test case)
- Password generation is user/admin dependent
- For more number of users the password generation will be automatic
- Client Server Connection will be done for the GUI model via Azure cloud

Conditions only for project prototype

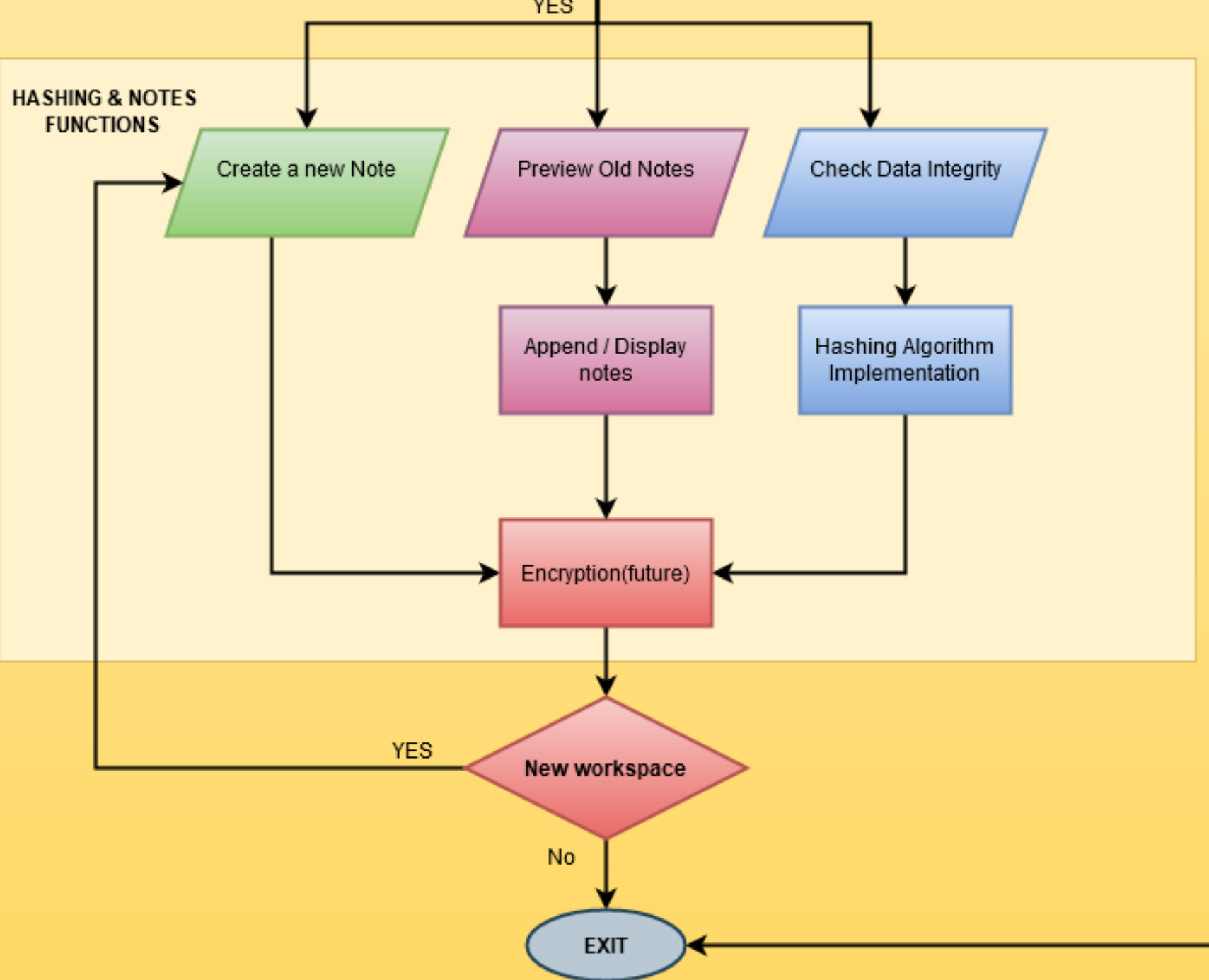
DESIGN AND ALGORITHM

Working and Practical Implementation of the
project





Divided in 3 parts:
AUTHENTICATION,
HASHING
AND
NOTES FUNCTIONS



MidSem Completion

- CLI based Notes functions
- Password Implementation

EndSem Planning

- OTP Implementation
- Hashing Algorithm Implementation

Future Scope

- Encryption
- Cloud based environment

Algorithms : User Input

For w,r and a : write, read and append notes

```
String sx=sc.nextLine();
if(sx.equalsIgnoreCase("r"))
{
    new FReading();
}
else if(sx.equalsIgnoreCase("w") || sx.equalsIgnoreCase("a"))
{
    writingToFile(sx);
}
else
{
    System.out.println("Sorry you try to do unexpected ,betterluck next time ");
}
sc.close();
```

- Step1- Asks user for input r(read), w(write), a(append)
- Step2- If 'a' input from user read file function is called.
- Step3- If 'w' or 'a' input from user write file function is called.
- Step4- Else it displays unexpected input.

Algorithms: Writing content in notes

Writing content in notes. Using FileWriter

```
FileWriter f0 =null;
if(s.equalsIgnoreCase("w"))
{
    f0 = new FileWriter(f,false);
    System.out.println("CAUTION >> Please understand it will overwrite the content of the file ");
    System.out.println("Type 'no' to exit");
    System.out.println("Do you want to proceed :type 'yes' ");
    in=new Scanner(System.in);
    String sl=in.nextLine();
    if(sl.equals("no"))
    System.exit(0);
    System.out.println("Write 'stop' when you finish writing file ");
    //f.delete(); //optional karna h
    f.createNewFile();
    while(!(source=bf.readLine()).equalsIgnoreCase("stop")){
        f0.write(source + System.getProperty("line.separator"));
    }

    in.close();
}
```

- Step1- User enter 'w' to write to a file
- Step2- If file exists, warns about overwriting
- Step3- If no, exit
- Step4- If yes, proceed
- Step5- User writes to the file
- Step6- User enters 'Stop' to finish writing

Algorithms: Appending changes to previous notes

```
//For appending the content
else
{
f0 = new FileWriter(f,true);
    System.out.println("Write 'stop' when you finish appending file ");
    while(!(source=bf.readLine()).equalsIgnoreCase("stop")){
        f0.append(source+ System.getProperty("line.separator"));
    }

f0.close();
```

- Step1- User enter appending mode
- Step2- User enter data
- Step3- User enters 'Stop' to finish appending file

Algorithms: Reading notes

Step1- User enter reading mode

Step2- User enters file name which he wants to read

Step3- If file exists then user reads file

Step4- If file doesn't exist, error displayed

```
try{
    Scanner s3=new Scanner(System.in);
    System.out.println("Enter file name u want to read");
    String fq=s3.nextLine();
    File f5=new File(fq);           //choose option which file to be read
    if(! f5.exists())
        f5.createNewFile();
    FileReader fl=new FileReader(f5);
    BufferedReader bf=new BufferedReader(fl);
    //For reading till end
    while((str=bf.readLine())!=null){
        System.out.println(str);
    }
    fl.close();
} catch (Exception e) {
```

Algorithms: MD5 Hashing for Data Integrity

```
switch(ch)
{
    case 1:try{
        String password = bf.readLine();
        String hash= String.format("%032x", // produces lower case 32 char wide hexa left-padded with 0
        new BigInteger(1, // handles large POSITIVE numbers
        MessageDigest.getInstance("MD5").digest(password.getBytes())));
        System.out.println("Here is the hash for your file for integrity check : " + hash);
    }
    catch (NoSuchAlgorithmException e) {
        e.printStackTrace();// do whatever seems relevant
    }
    case 2: break;
    default: System.out.println("Thannkyou, you will exit from the editor" );
}
```

Step1- All the contents of the file are passed on to the string

Step2- The hash that has to be generated will be of lowercase 32 characters

Step3- BigInteger helps in mathematical operation which involves very *big integer* calculations that are outside the limit

Step4- Message Digest for MD5

RESULTS

Reading Mode - A user can view any pre-created file in this mode but cannot edit its components.

Writing Mode - A user can create or edit the contents of the file in this mode but the changes will be overwritten means the previous data will not be saved.

Append Mode - A user can create or edit the contents of the file in this mode and previous data will not be erased from the memory.

It will show an error everytime user requests an operation other than read, write or append.

Password Creation - A new user is able to set a password on the application to secure their data. Everytime user opens the application they have to prove their authenticity by entering correct password.

Data Integrity – MD5 hashing implementation to maintain the hashing of the algorithm.

CLI based notes editor with 2-factor-authentication and Data Integrity

```
@@@|
!!! Welcome to NotTy: Notes with Added Security !!!
@@@|

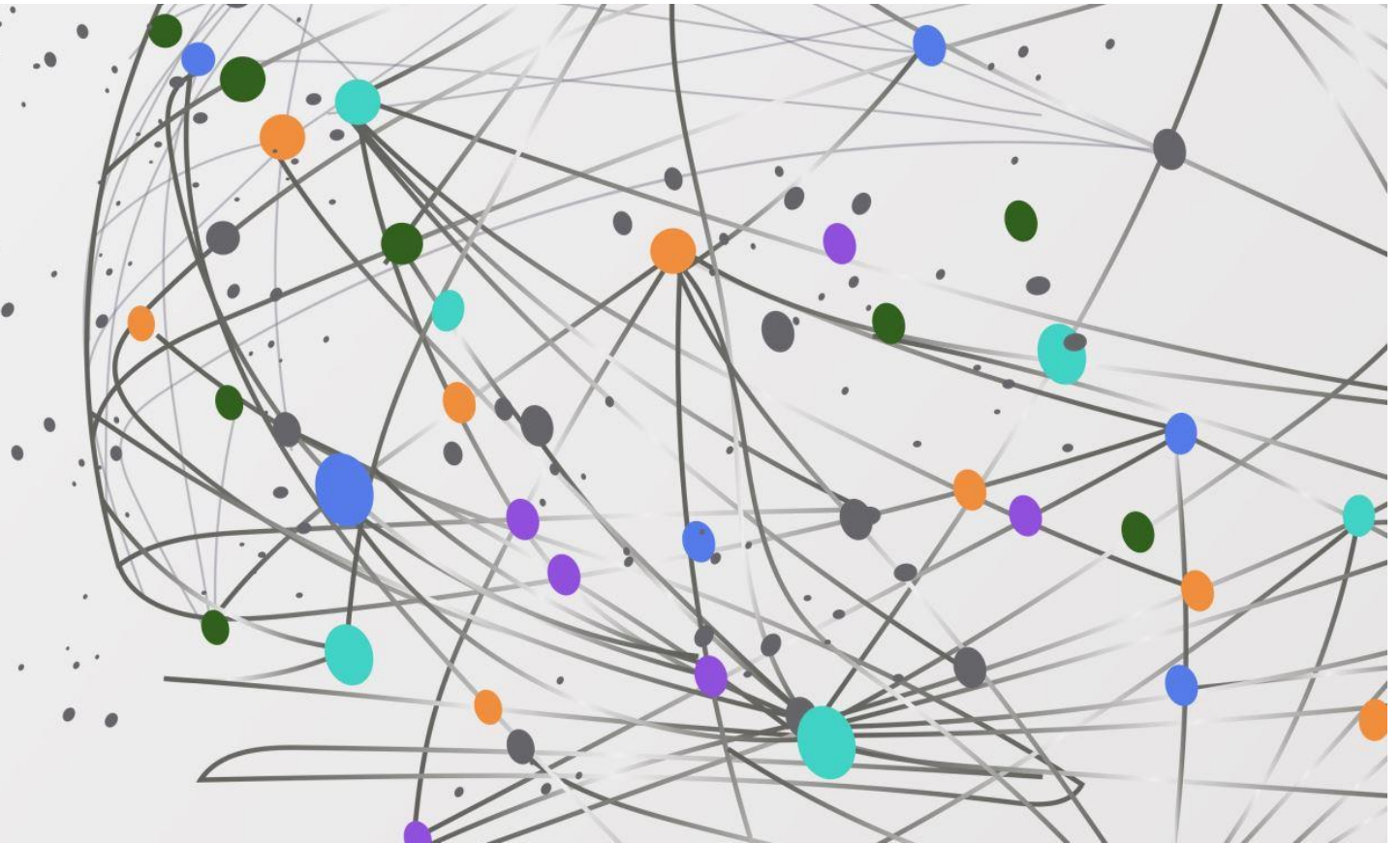
If you are a new user press 1 else if you are an existing user press 2
2
Enter your password
K@sh

*****Authenticate via One-Time-Password*****
Enter the OTP provided by our authenticator
0057
!** Successfullly Logged-In your account! **!

Please select one of the below operations
w for write mode
r for read mode
a for append mode
r
Enter file name u want to read
Kashish.txt
Part 2
Do you also want to check the integrity of the file you are reading? If yes press 1 else press 2
1
Enter file name u want to read
Kashish.txt
Here is the hash for your file for integrity check : 6417c68d02736b9d6bbccd60d0e3bd39
```

CONCLUSION

Through this application, end-users will get a platform where no intruders can access their data which is updated on NotTy without their consent as it is secured with two-factor authentication. For the future scope we would prefer shifting the prototype to azure cloud for better scalability and provide encryption implementation for passwords and maintaining high security standards.





Thank You

KASHISH SRIVASTAVA



YASH KUMAR



VISHAL PATWAL



ABHISHEK GUPTA

