

# File Handling System Using Google Drive Integration (FHS)

Sahil Shahane<sup>1</sup>, Uttkarsh Patel<sup>2</sup>, Prathmesh Satpute<sup>3</sup>, Mrunal Aware<sup>4</sup>

<sup>1-3</sup>Student, Dept. of Computer Engineering, MIT Polytechnic, Maharashtra, India

<sup>4</sup>Professor, Dept. of Computer Engineering, MIT Polytechnic, Maharashtra, India

\*\*\*

**Abstract** - We should not rely completely on data stored on our daily device. Be it a phone or a pc you should have a backup of your system. Cloud storage offers some advantages over local storage. In the proposed project, we have tried to overcome this problem by not only providing a user with a backup system but an automated one. Users can back up their photos and documents to their Google Drive. The system only updates the file on the cloud that is changed on the local machine and vice versa. The proposed system eliminates the headache of deleting individual files on the cloud storage and saves time.

**Key Words:** Python, Google Drive, JavaScript, Electron, File Management

## 1. INTRODUCTION

Every person should have their important files and folders backed up to avoid loss of data due to an accident. Many people prefer having offline backups of their important documents and photos to avoid the additional cost of an online backup. What they don't realize is that having an offline device is still prone to damage and corruption. Having a cloud-based backup seems a good option.

FHS is aimed to resolve the problem of syncing different files in different devices by storing them on a remote server. It automates the long and cumbersome process of uploading and downloading individual files to and from the server, which was earlier done manually. The proposed system also takes care of the situations where some files were not uploaded or downloaded due to poor internet connection or any other reason.

By using a cloud, we can reduce the risk of data loss, security and cost of an external backup device. The proposed project is not just targeted to solve the problem of backing data but also to manage it in a smart and sophisticated way. There are plenty of options to choose from but a major problem with most of them is that the users need to purchase a subscription which may not be much for western countries but is a lot more for Indians. The proposed project

overcomes that problem by making use of the existing storage provided to them.

Indians generally don't think of spending on cloud storage and store their important data either on their daily device or a small hardisk. Storing on Google Drive seems like a nice option. The proposed application makes the use of that 15 GB to store the user's important data. Yes, 15 GB is comparatively less, but for all the important data that the user can access from anywhere in the world it seems just enough. If in case the user needs more storage, he/she can easily expand the users limit with a low-priced subscription. Users should trust Google for the security of their data.

## 2. LITERATURE SURVEY

To achieve safe policy-based file access control, file assured deletion, and policy-based renewal of a file stored in a cloud environment, a suitable encryption technique with key management should be applied before outsourcing the data. The proposed project used a remote backup system which is an advanced concept which reduces the cost of implementing more memory in an organization. It helps enterprises and government agencies reduce their financial overhead and to achieve secure data transactions in the cloud. To achieve secure data transactions in the cloud, a suitable cryptography method is used. [1]

Here the primary goal is to make both version control and assured deletion compatible with each other in a single design. For this, a secure cloud-based backup system that supports both version control and assured deletion is used, such that cloud clients can specify particular versions or files on the cloud to be assuredly deleted, while other versions that share the common data of the deleted versions or files will remain unaffected. [2]

Proposed project seeks to address the access control and assured deletion problems from a practical perspective. The system can now outsource data backups off-site to third-party cloud storage services to reduce data management costs. However, it must provide security guarantees for the outsourced data, which is now maintained by third parties.

The proposed system, designed and implemented FADE, a secure overlay cloud storage system that achieves fine-grained, policy-based access control, and file assured deletion.[3]

### 3. APPROACH

A file management system is used for file maintenance (or management) operations. It is a type of software that manages data files in a computer system. As technology has advanced a lot, more data is being generated, and managing this data manually is a hard and complex job (depending upon the amount of data). To manage this data, we have created software which is capable of automatically backing up data with synchronization support with the help of python. To gather user data, we use Python’s inbuilt libraries which are already optimized, and to generate the metadata from the user’s data we use advanced hashing algorithms such as MD5. After generating the required data for the software, we automatically backup user data with the appropriate cloud storage services which provide the 128-bit AES encryption and for storing the generated data locally we have used “feather”. The software has advanced features like detecting changes in files that can be used to back up only changed files after the first backup which reduces the backup time to a great extent. The software can automatically detect changes in files in cloud storage and automatically update it in local storage so that the user stays up-to-date with their files which reduces the time to manually download and move the data.

### 4. PROPOSED METHODOLOGY

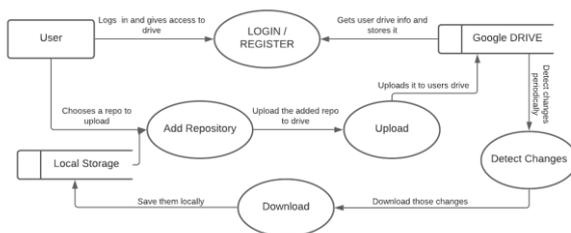


Fig-1 Data Flow Diagram (Level 0) of the system

#### 4.1 Backend

Before the user can start uploading the data it needs to specify the files and folders that it wishes to backup. Once all the files and folders are specified, they are traversed one by one to save their current state. The targeted files are specified, all the metadata regarding those files are saved using the Pandas library in python and stored using another

library called Feather as it gives very promising results when compared according to their memory efficiency, I/O Speeds, indexing. This process doesn't take much time and is extremely efficient.

#### 4.2 Google Drive

To interact with the Drive that is to upload, download and detect changes to and from the drive, the Google Drive API and Google Drive Activity API are used with python.

Google Drive API is used for uploading and downloading the necessary files and folders. Files and folders are referenced with their respective drive id which are automatically assigned to them on uploading or can be pre-generated.

The whole process of uploading and downloading the files is multithreaded to increase the efficiency and decrease the waiting time, and utilize the system resources in a better manner. Python’s multithreading module is used to implement and manage the threads. A scheduler gets instructions from the front-end about the type of operations to be performed by the scripts along with their data. The scheduler then schedules multiple threads for the operation which performs the required task.

#### 4.3 Frontend

The GUI for the application is created using React JS along with Electron JS. “Electron (formerly known as Atom Shell) is an open-source software framework developed and maintained by GitHub. It allows for the development of desktop GUI applications using web technologies: it combines the Chromium rendering engine and the Node.js runtime [5]”. It also supports multiple operating systems, making the application platform independent. React is a front-end library mostly used for single page applications. It promotes fast development of websites and easy to manage functionalities without negotiating with speed.

To make the development easy and provide a better interface Ant Design is used. Ant Design is a React UI library that contains components which are helpful for building elegant user interfaces.

The interaction between the frontend in JavaScript and the backend in Python is implemented with the help of a node library called python-shell. It uses IPC to communicate with python scripts by using the standard I/O streams. And on top of that it also supports JSON format for communication.

## 5. WORKING

The user chooses a folder to be synced and backed up with the cloud in this case Google Drive. In the selected folder a new hidden file is generated with all the information needed to track the local changes of the file and the syncing information with the Drive, just like what happens when we initialize git in a repository. File and folder information along with their hashes are stored in JSON format, and all the other metadata needed to track activities are stored in an SQL database inside the same folder.

Whenever the user changes some files and opens the software the python scripts detect whether anything has changed. If anything is changed it updates the current metadata with the new metadata and sends a signal to the UI about the changes. The UI in return prompts the user about the changes, the user can then choose whether to upload and not to sync the data according to their convenience. A similar process takes place for downloading.

### 5.1 Uploading

For uploading the files for the first time or for the changes, the specific files and folders are selected from the SQL database and are sent to the python scheduler to upload. Google Drive API is used to upload the files. The scheduler schedules and creates threads for every upload. As the scheduler is scheduling the uploads the UI is updated, depicting the files and folders scheduled for uploading. On the completion of upload the uploading function returns a response through the standard output stream. The frontend receives this message and updates the completion of the specified file in the UI.

### 5.2 Downloading

To download files is to check the changed files in the Drive and then to download them in order to update the old versioned file present in the local storage. Required metadata of the files are fetched from the SQL database and sent to the scheduler. The scheduler creates threads for downloading the files and then starts updating them with the local files.

### 5.3 Changes

The Google Drive Activity API is used along with the Google Drive API to get the changes from the Drive. The changes are checked from time to time and not always in order to save bandwidth. As soon as the changes are checked, the

repository specific SQL database is updated with the latest information, and the user is asked to approve making changes to the local files, if yes then the files are scheduled to download.

### 5.4 Detecting Changes Locally

All the files in the selected repository are traversed and their last modified time is recorded and stored in a hidden folder inside that repository. When the user syncs the files with the drive all the files in the repository are again traversed and compared with the previously saved data. In case there are changes in the existing files or folders then appropriate actions like uploading or downloading to and from the google drive are performed, in order to sync the files, present in both the locations.

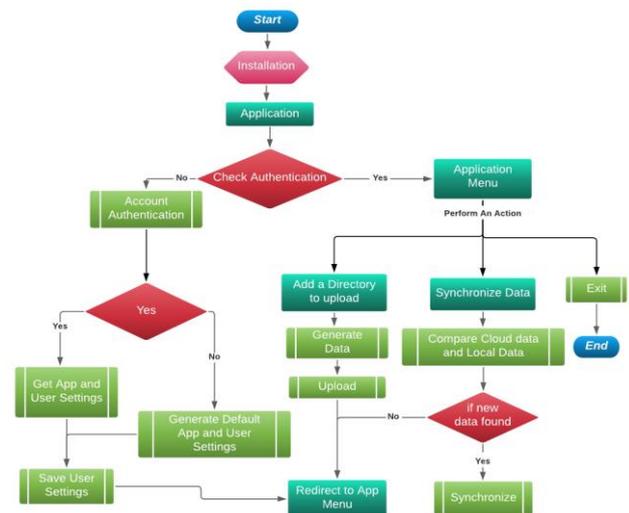


Fig - 2 Activity Diagram of the application

## 6. CONCLUSION

The system being developed is completely feasible as it makes the use of existing technology in a better manner. It not only automates the existing procedure of uploading new files one by one but also gives the user the ability to keep track of it. Security is not one of the concerns as the data stored is completely safe and only accessible by the user. Since no extra device is needed, there are no additional expenses for the functioning of the project, other than the regular internet charges occurring due to upload or download of the data, unless the user chooses to opt for more storage space.

## 7. REFERENCES

[1] Secure Cloud Storage using Decentralized Access Control with Anonymous Authentication R.Ranjith, D.Kayathri Devi #International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 11 (November 2013)

[2] Secure Overlay Cloud Storage with Access Control and Assured Deletion by Yang Tang, Patrick P. C. Lee, John C. S. Lui, Radia Perlman # The chinese university of Hong Kong | CUHK - Department of computer science and engineering, (November 2012)

[3] A Secure Cloud Backup System with Assured Deletion and Version Control Arthur Rahumed, Henry C. H. Chen, Yang Tang, Patrick P. C. Lee, and John C. S. Lui The Chinese University of Hong Kong, Hong Kong @cse.cuhk.edu.hk (September 2011)

[4] D. Boneh and R. Lipton. A Revocable Backup System. In Proc. of USENIX Security Symposium, 1996.

[5]Wikipedia Electron (software framework),

[https://en.wikipedia.org/wiki/Electron\\_\(software\\_framework\)](https://en.wikipedia.org/wiki/Electron_(software_framework)), [Accessed 25 March 2021, at 18:14 (UTC)]

[6] Google Drive API for Developers, <https://developers.google.com/drive/api/v3/about-sdk>, [Accessed 25 March 2021, at 19:01 (UTC)]