A MINI PROJECT REPORT

On

Cloud Drive

Submitted by

Shalabh Chaudhary
Roll No: 151500470
Prateek Agarwal
Roll No: 151200276

Department of Computer Engineering & Applications
Institute of Engineering & Technology

GLA University
Mathura- 281406, INDIA
December, 2017

Department of Computer Engineering and Applications
GLA University, Mathura
17 km. Stone NH#2, Mathura-Delhi Road, P.O. – Chaumuha,
Mathura – 281406

DECLARATION

We hereby declare that the work which is being presented in the Mini Project "**Cloud Drive",** in partial fulfillment of the requirements for Mini-Project LAB, is an authentic record of our own work carried under the supervision of **Mr. Saurabh Singhal, Assistant Professor, GLA University, Mathura**.

Shalabh Chaudhary
Roll. No. : 151500470
Course: B.Tech
Year: III
Semester: V

Prateek Agarwal
Roll. No. : 151200276
Course: B.Tech
Year: III
Semester: V

Department of Computer Engineering and Applications
GLA University, Mathura
17 km. Stone NH#2, Mathura-Delhi Road, P.O. – Chaumuha,
Mathura – 281406

CERTIFICATE

This is to certify that the project entitled **"*Cloud Drive*"** carried out in Mini Project- I Lab is a bonafide work done by ***Shalabh Chaudhary (151500470) and Prateek Agrawal (151200276)*** and is submitted in partial fulfillment of the requirements for the award of the degree Bachelor of Technology (Computer Science & Engineering).

Signature of Supervisor:

Name of Supervisor: Mr. Saurabh Singhal

Date: 15/12/2017

ACKNOWLEDGEMENT

It gives us a great sense of pleasure to present the report of the B. Tech Mini Project undertaken during B. Tech. Third Year. This project in itself is an acknowledgement to the inspiration, drive and technical assistance contributed to it by many individuals. This project would never have seen the light of the day without the help and guidance that we have received.

Our heartiest thanks to ***Dr. (Prof). Anand Singh Jalal,*** *Head of Dept., Department of CEA for providing us with an encouraging platform to develop this project, which thus helped us in shaping our abilities towards a constructive goal.*

We owe special debt of gratitude to ***Mr. Saurabh Singhal,*** *Assistant Professor, Department of CEA, for his constant support and guidance throughout the course of our work. His sincerity, thoroughness and perseverance have been a constant source of inspiration for us. He has showered us with all his extensively experienced ideas and insightful comments at virtually all stages of the project & has also taught us about the latest industry-oriented technologies.*

We also do not like to miss the opportunity to acknowledge the contribution of all faculty members of the department for their kind guidance and cooperation during the development of our project.

ABSTRACT

Mini Project - I named as *Cloud drive*, which is done under the guidance of our respected HOD of Computer Science Engineering And Application Department and Mr. Saurabh Singhal, Assistant Professor, our guide, at GLA University, Mathura. The objective of this project is to develop a desktop application which can handle, manage, or maintain clients and users data to cloud(like Google Drive) easily from their own desktop that can be either Linux or Windows.

TABLE OF CONTENT

1. Introduction

The project(Cloud Drive) is a software application which runs on Linux and Windows desktop.
The application works with Oauth2 authentication mechanism and would prompt to the user to grant access the Files and about mechanism. It uses Google Drive API that allow clients to access the resources from Google Drive.
This project is all about to develop or build an desktop application that can manage users data and file on cloud i.e., Drive of their Gmail account.

1.1 Motivation & Overview

The motivation for this project came in the mind through study of Google web products and its services. Google provides a plethora of APIs which can be easily integrated with your web or desktop applications. It was also found that to make things more flexible or professional Google provide free development APIs. Such APIs not only make private applications integrate-able but also brings in possibilities for seamlessly consuming cloud services with just one click of a mouse for the end-user.
Google has a number of cloud tools which are useful including - google drive, calendar, fusion tables (database with google maps), sites (web pages), etc. and as we know that Google.lnc has not provided any client for Google Drive for Linux platform till now.
Therefore, the idea for developing a project called *Cloud Drive* came into role.

1.2 Objective

The main objective is to demonstrate and develop an application that can offer the user the capability to manipulate the files and data from desktop, the application will be a cross platform that use Google APIs in Python and able to Create, Delete, Modify, Share, Access, Contact to owner and even files will be synced automatically to cloud whenever system get connected to internet.
Cloud Drive provides a UI(user interface) to its consumer so that they able to manage their files accordingly as per their choice and even can access anywhere wherever they want but with a single condition that they should have access of the internet.

2. Software Requirement Analysis

"*Cloud Drive*" rely on very little in the way of external software interfaces.However, the system will require interfaces with the installed computer's hardware. The system is a web-enabled system i.e., all user interaction is done through a web browser.

The "*Cloud Drive*" needs PyQt5. PyQt5 doesn't come preinstalled on Windows and Linux Platform. So it need to be installed while user install the application.

Technologies Used:

- Python3
- Shell

Libraries Used:

- PyQt5
- Pyuic5
- Httplib2
- Google api Python client

Broad Requirements:
OS: Linux
IDE: Jetbrains Pycharm

Dependencies →

- Application will be installed on a machine running either Windows or Linux OS.
- Pip and Python3 must be preinstalled on the system.
- Network adapter must be available and enabled in the system.
- The system must be supported to established HTTPS connections.
- The user must have a Google account and storage available in Google Drive.
- The application must be non-blocked by firewall (if available).
- A secure internet connection must be available in system that user can access.

The interfaces required on the system server are :

→ Network interface to a network with an internet connection.

→ All user interfaces other than initial installation occur at time of installation.

→ There are no hardware interfaces to this system.

2.1 Define the Problem

The application "*Cloud Drive*" will be cross platform i.e., can be run on all platforms that can be Linux, Windows etc. It is a client for Google Drive. As, Google.lnc has not provided any client for Google Drive for Linux platform. Therefore, this application helps the user to have full control to manage their files and folders on cloud(eg., Google Drive) even in Linux system.

2.2 Define the Modules and their Functionalities (SRS)

A package will be provided to user and need to be installed. After installing, a folder named Cloud Drive will be created in home folder(In Linux), a folder will be added to quick access (in Windows). User needs to first Signup then Sign In to his Google account and application will have access of his Google drive to update, delete, add the files on cloud.

Python3 and Pip must be preinstalled on the operating system. The user needs to give exception in firewall (if any) and the system must support to network connections and HTTPS protocol.

All offline available files will be automatically synchronized to his Cloud Drive folder and the newly created files or folder will be uploaded to cloud. Synchronization will be done automatically when the device will be connected to internet.

<u>Functionalities of Cloud Drive</u>→

1. Create files - Creates a new file. This method supports an **/upload** URI and accepts uploaded media with the following characteristics:
   (i) Maximum file size: 5120GB
   (ii) Accepted Media MIME types: */*
   If successful, this method returns a Files resource in the response body.

2. Delete files - Permanently deletes a file owned by the user without moving it to the trash. If the file belongs to a Team Drive the user must be an organizer on the parent. If the target is a folder, all descendants owned by the user are also deleted.
   If successful, this method returns an empty response body.

3. Empty Trash - Permanently deletes all of the user's trashed files. If successful, this method returns an empty response body.
   If successful, this method returns an empty response body.

4. Update/Modify files - Updates a file's metadata and/or content with patch semantics.
   If successful, this method returns a Files resource in the response body.

5. Share files with others - This app can manage permissions and sharing settings of files. Also, apps can provide commenting and discussions on files, allowing users to work collaboratively on the documents they share in Drive.
   Access to files & folders is determined by an access control list (ACL) that determine whether or not users can perform actions on a file such as read or write.

6. Contact to owner - User can contact to owner if they face any type of problem or error. Every authenticated user can easily mail their problems to owner using mail id of their personal google account.

7. Login/Logout - User can login from their google account to get access to all feature of Cloud Drive application and when they don’t need it further they can simply logout.

3. Software Design

- Data Flow Diagram [DFD]

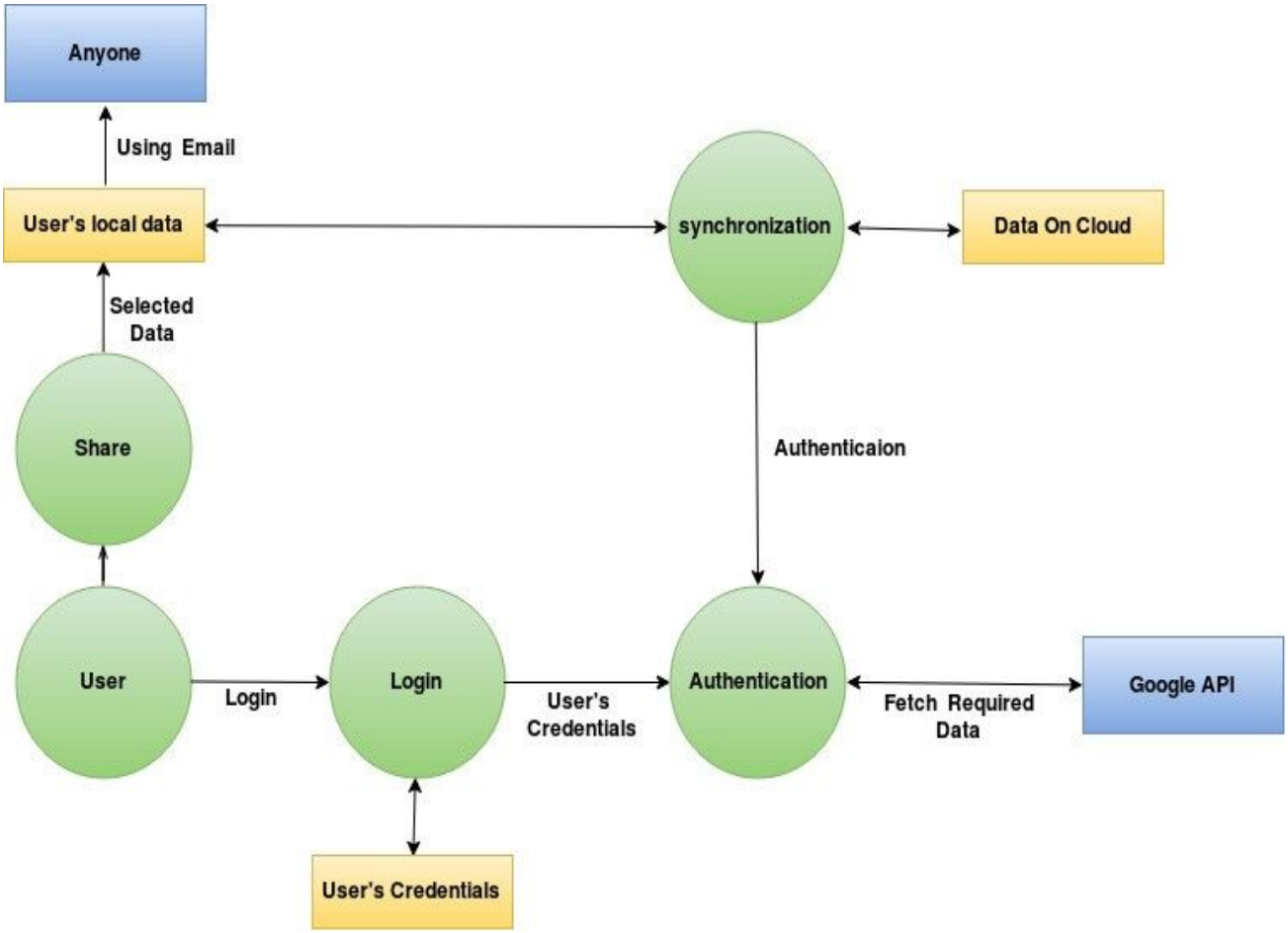

Fig. 3.1

- UML Diagrams - Use Case Diagram

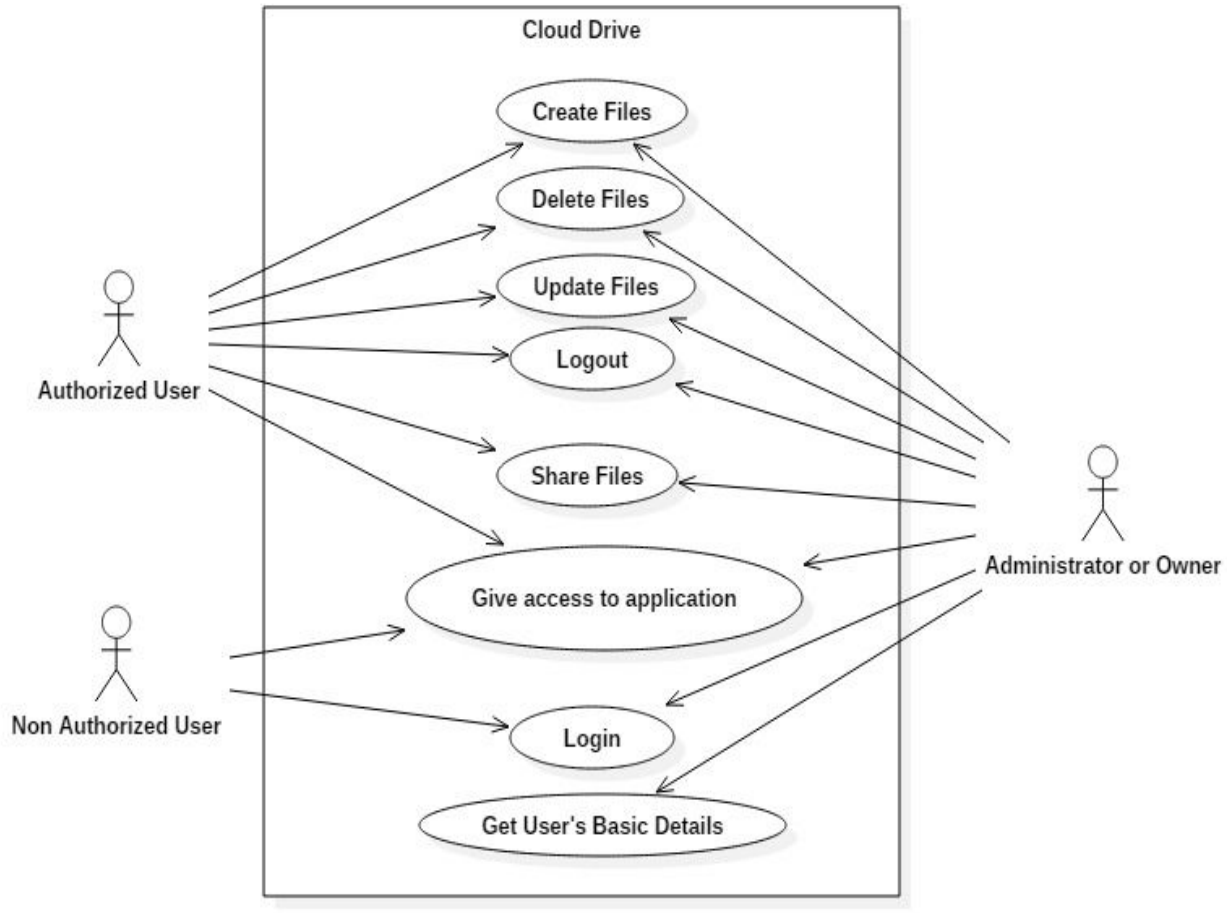

Fig. 3.2

4. Testing

Software Testing is evaluation of the software against requirements gathered from users and system specifications. Testing is conducted at the phase level in software development life cycle or at module level in program code.

Black-box testing

It is carried out to test functionality of the program. It is also called 'Behavioral' testing. The tester in this case, has a set of input values and respective desired results. On providing input, if the output matches with the desired results, the program is tested 'ok', and problematic otherwise.

White-box testing

It is conducted to test program and its implementation, in order to improve code efficiency or structure. It is also known as 'Structural' testing. In this testing method, the design and structure of the code are known to the tester. Programmers of the code conduct this test on the code.

4.1 Test Case I → Login Checked by Google Login Feature

Input	**Output**	**Status**
Email= shalabh.com	Invalid email	Fail
Email=shalabh@gmail.com		Successfully Log in
Password= 123	Invalid password	Fail
Password= 12345		Successfully Log in

4.2 Test-Case II → For Upload File

The list of requirements for file upload feature must have in order to be a good module in app.

1. Set the file size required for app. Make sure file upload module recognizes file size limit
2. File upload module should have the file type recognizer functionality built into it.
3. Accept characters and alphanumeric content in names.
4. Don't accept empty files with extension that leads to SQL injection.
5. Authorize the file upload and transfer to only registered accounts.
6. Create new copy of the same uploaded file to avoid overwriting.

For Security Purposes - security options to check while testing the application

1. Upload virus file and measure the response.
2. Upload files consistently without any wait period (denial of service check).
3. Try to upload file as a guest user or anonymous user.

5. Implementation and User Interface

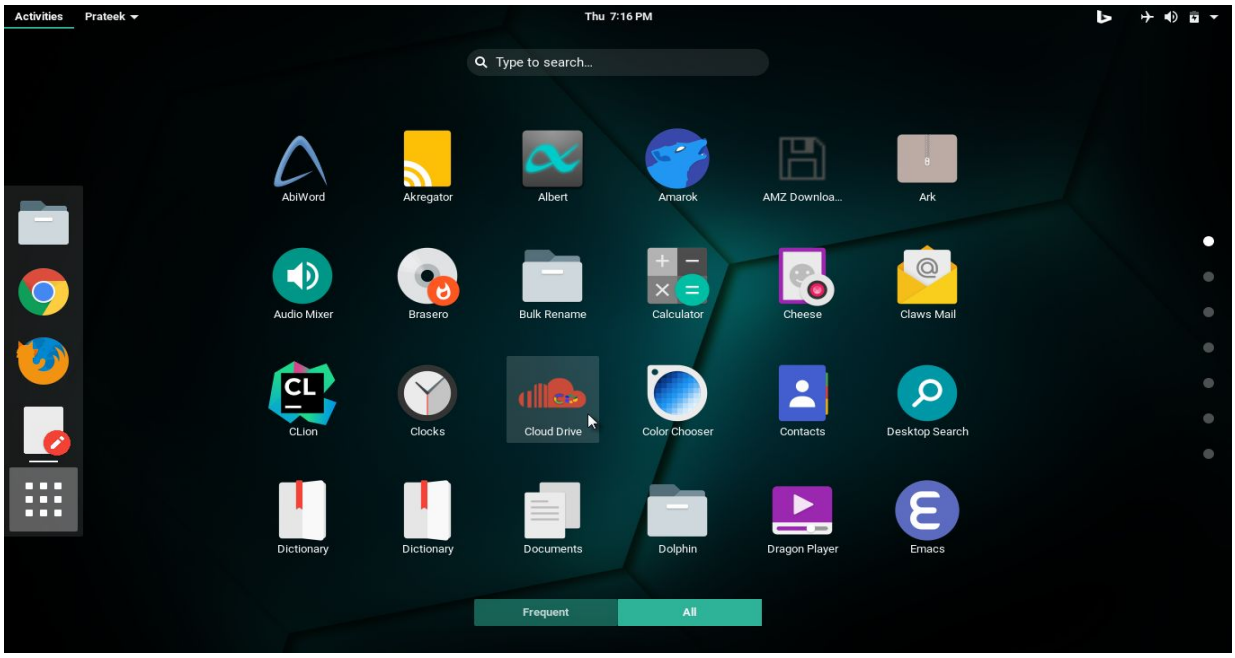

Fig. 5.1 - Start_menu

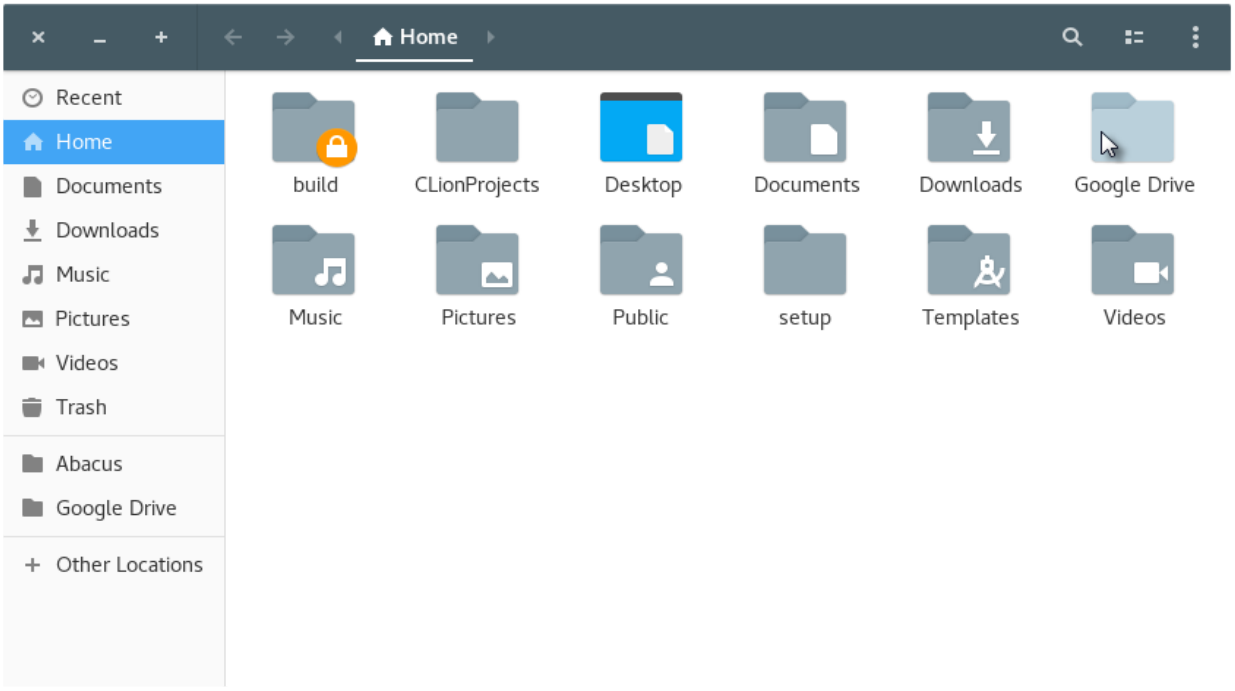

Fig. 5.2 - Home_dir

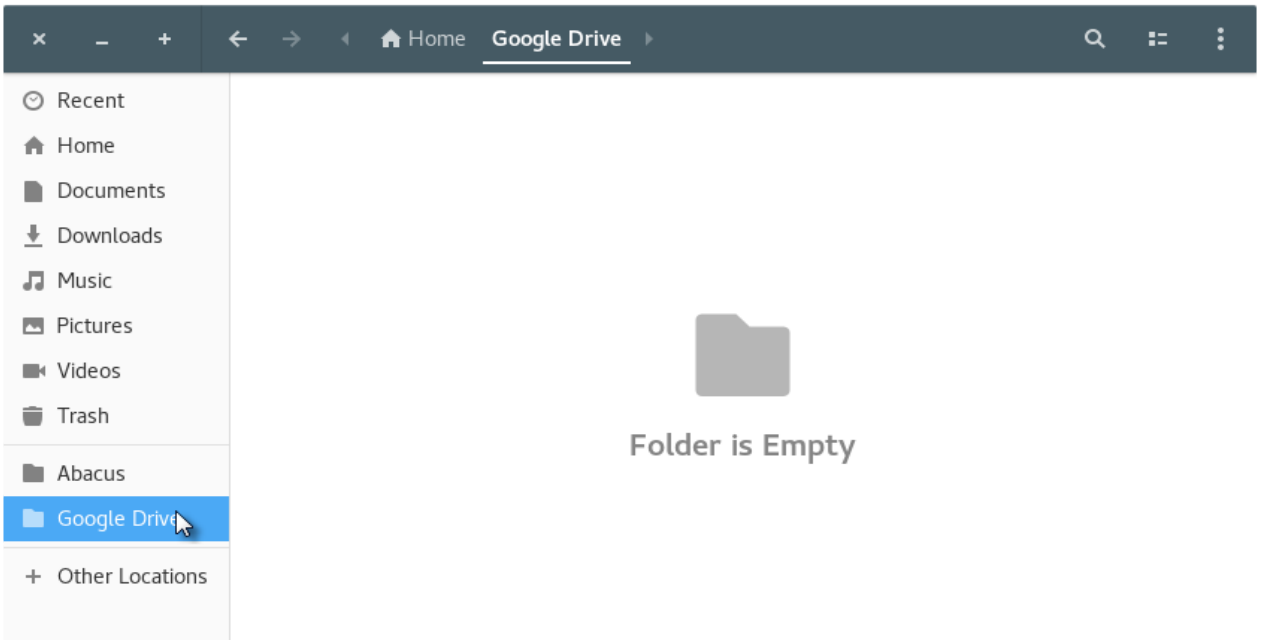

Fig. 5.3 - Bookmark

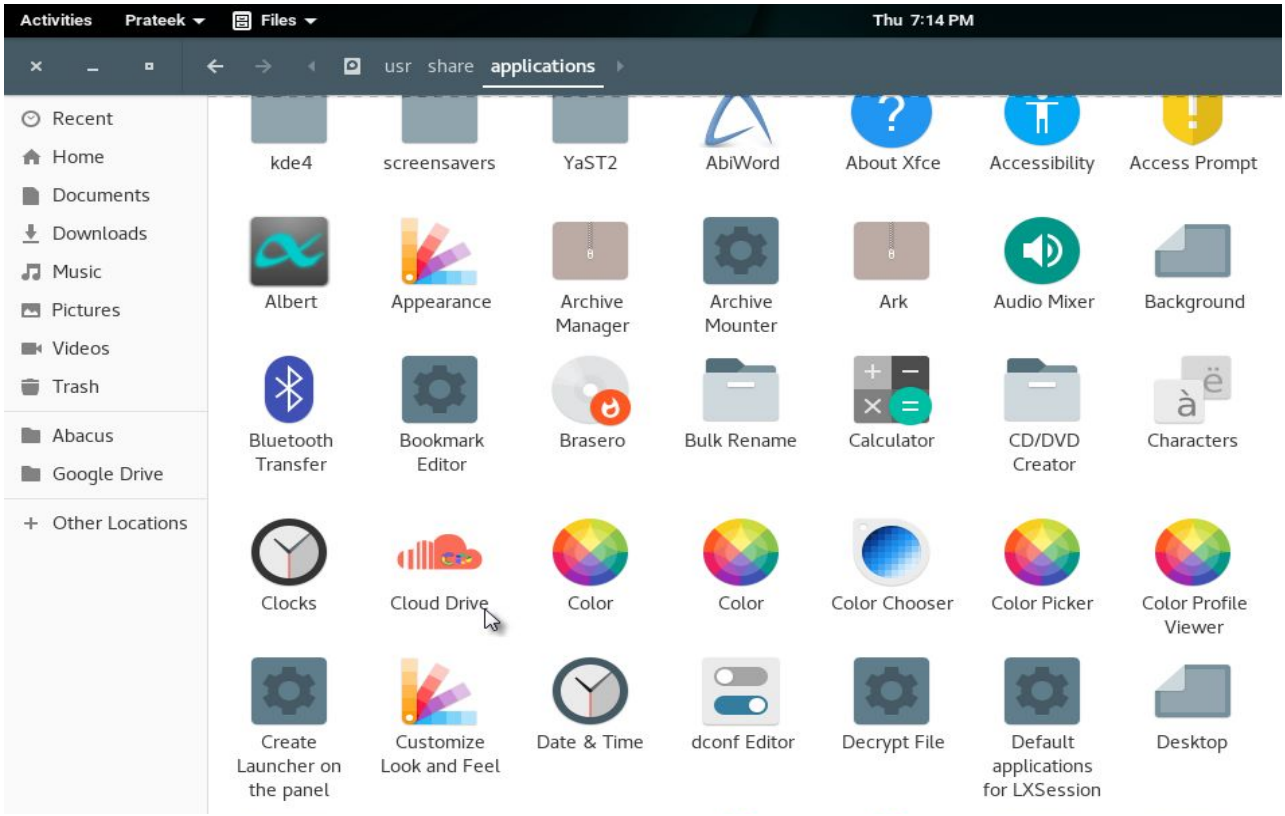

Fig. 5.4 - start_menu_entry

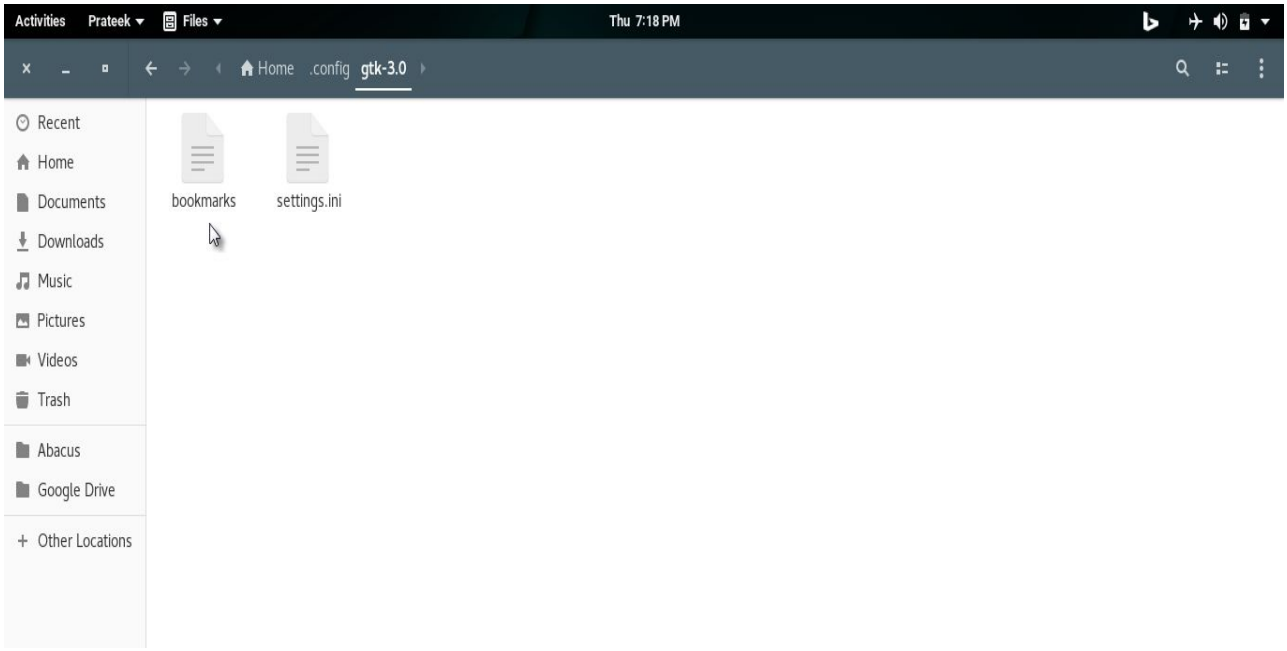

Fig. 5.5 - nautilus_bookmark_entry

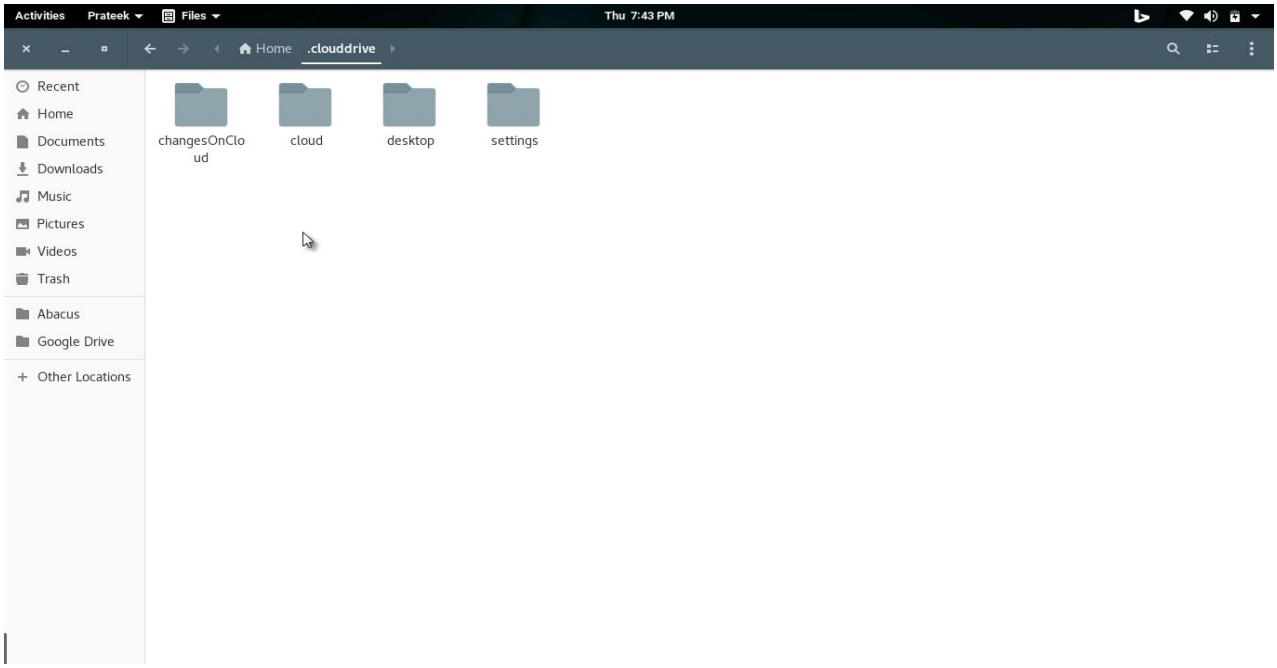

Fig. 5.6 - Settings

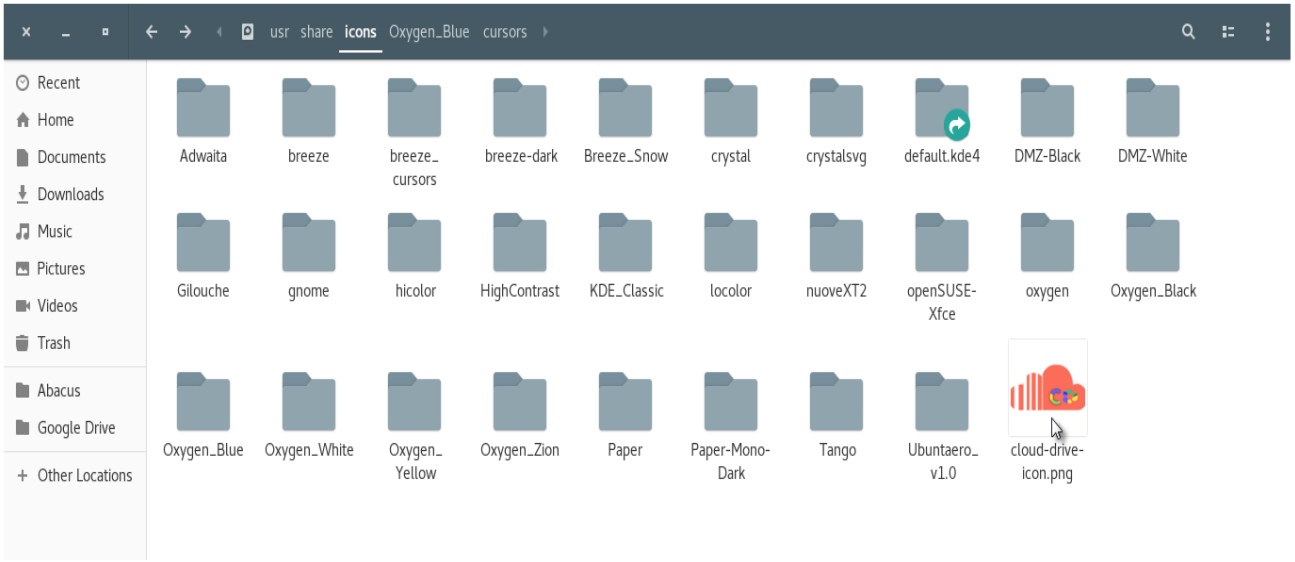

Fig. 5.7 - icon_location

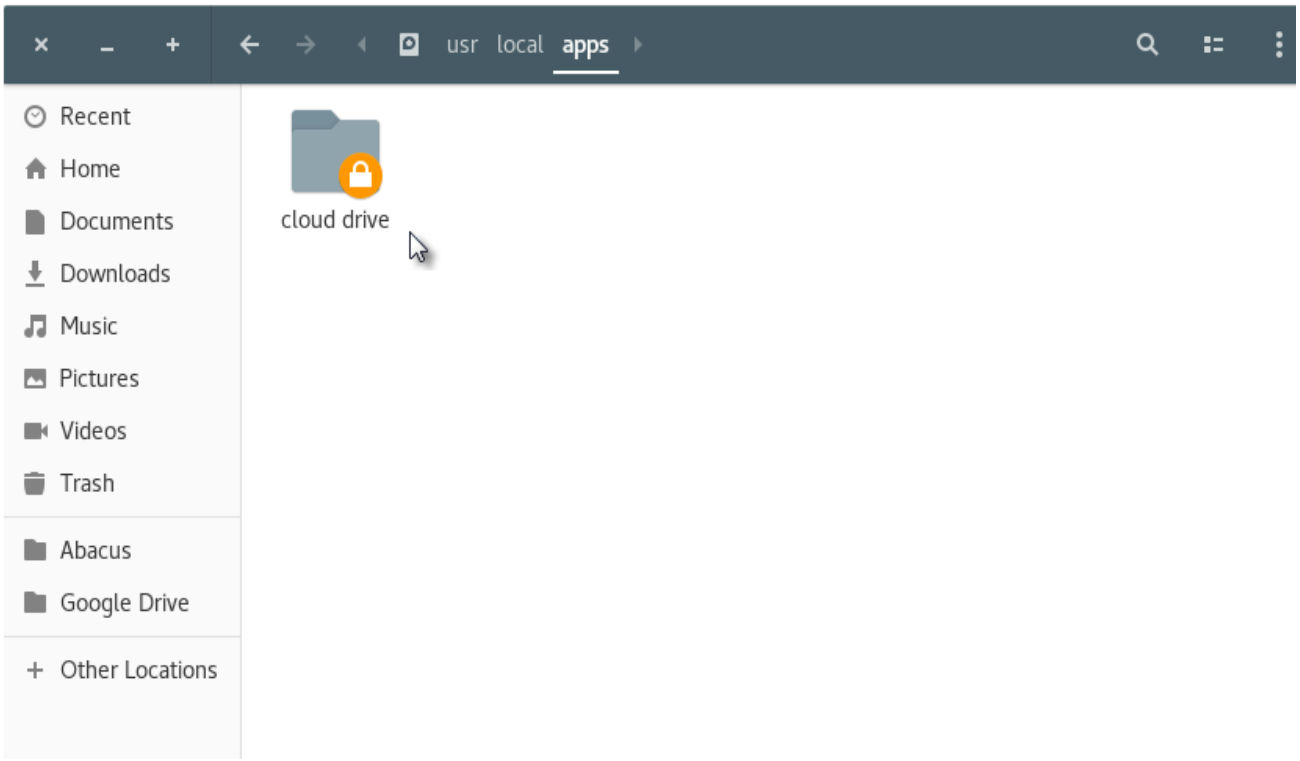

Fig. 5.8 - installation_directory

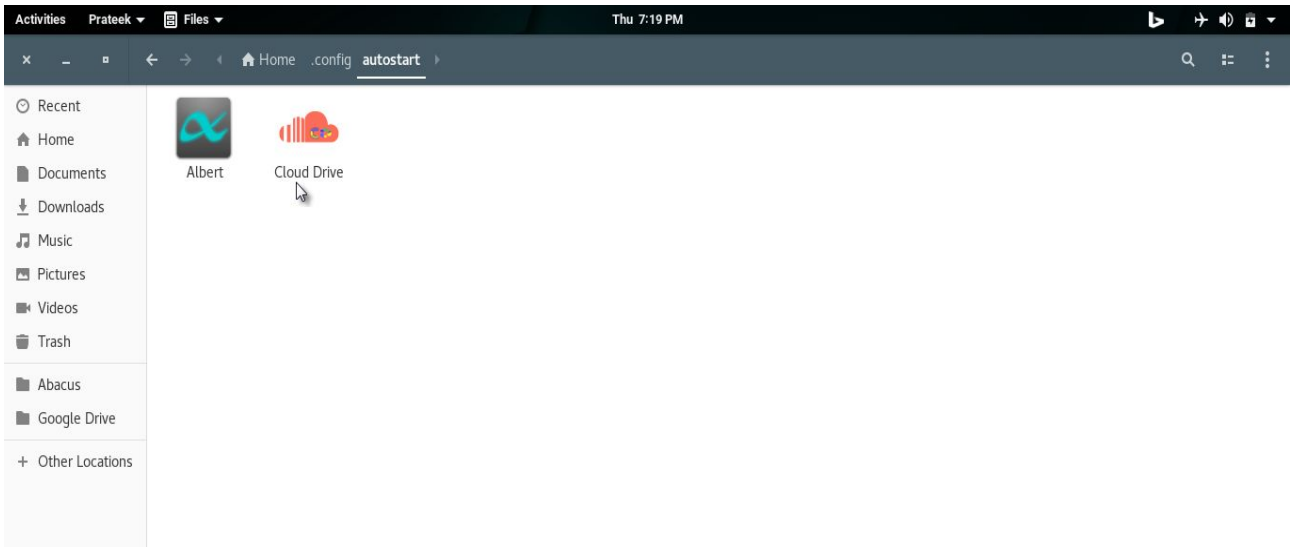

Fig. 5.9 - Startup

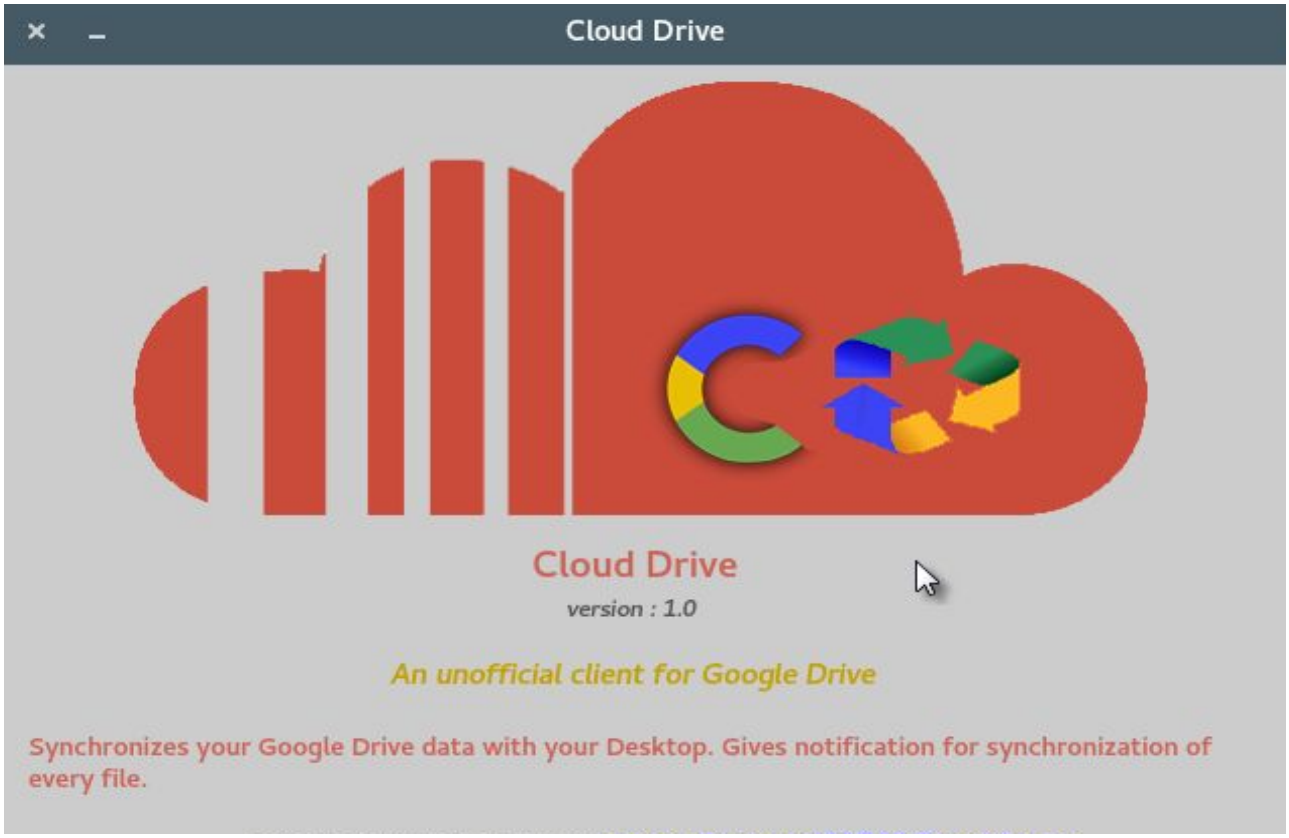

Fig. 5.10

Fig. 5.11 - start_menu_kde

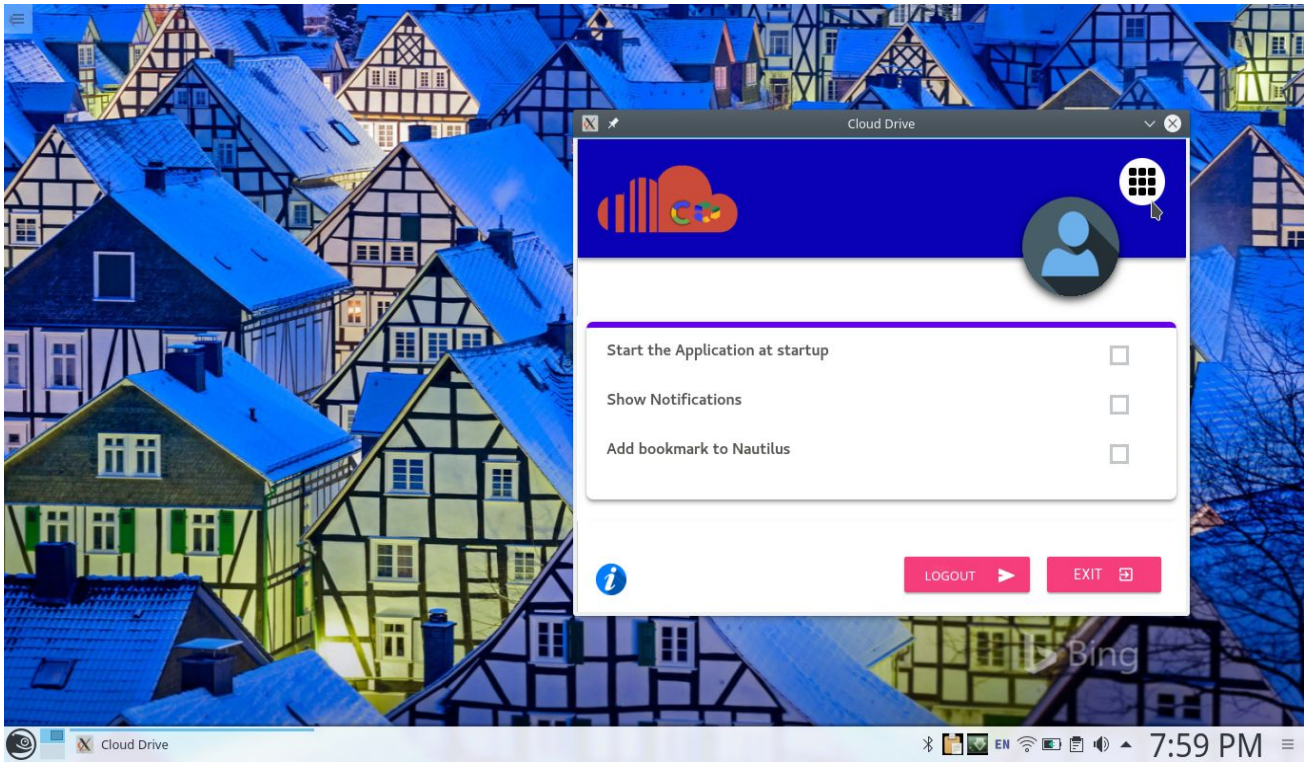

Fig. 5.12 - main_window_on_kde

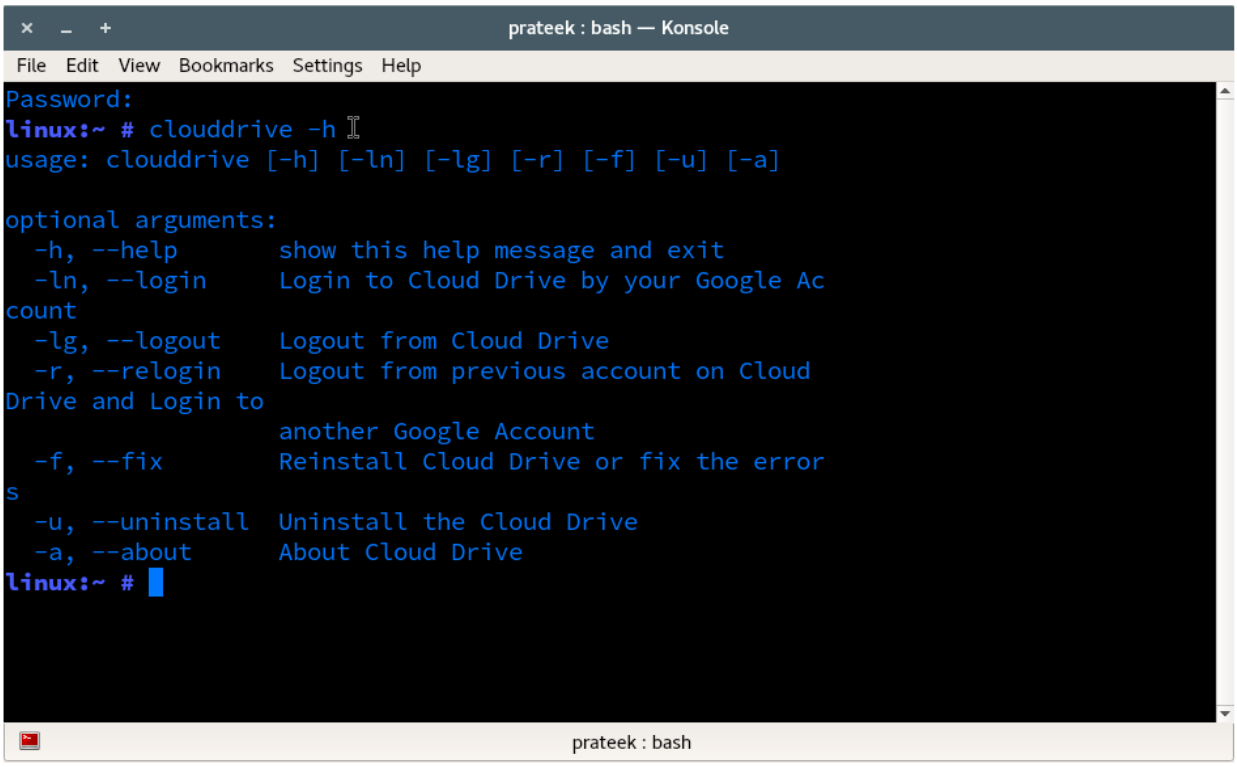

Fig. 5.13 - Terminal

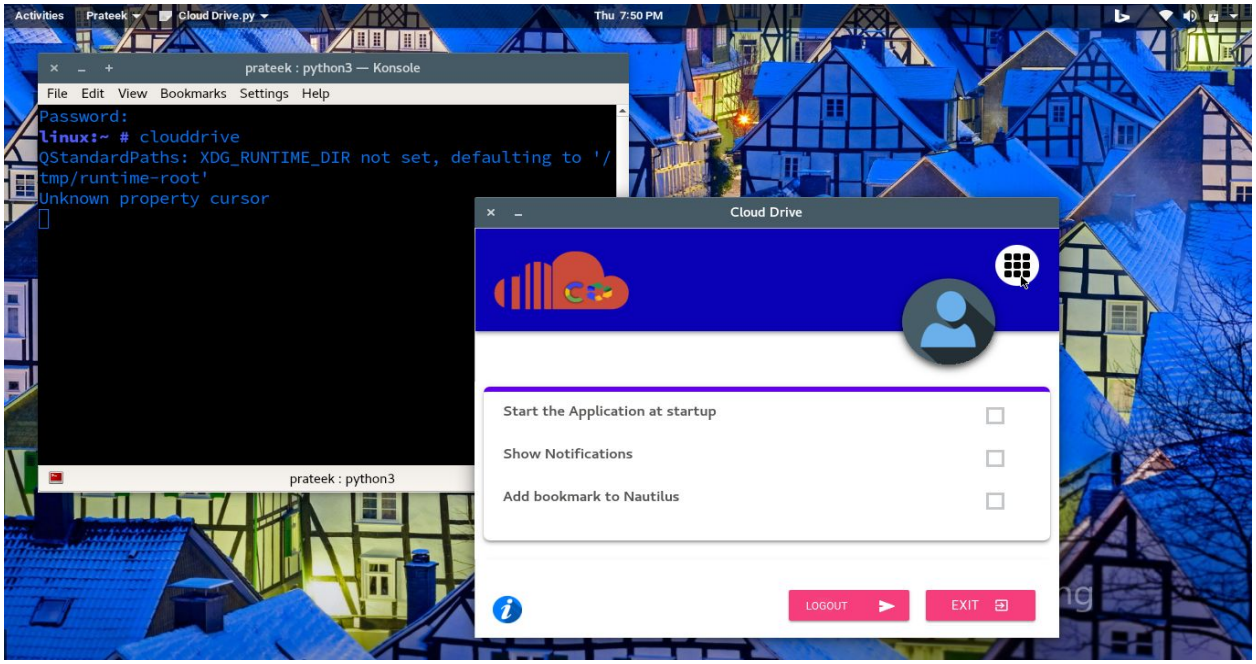

Fig. 5.14 - launch_from terminal

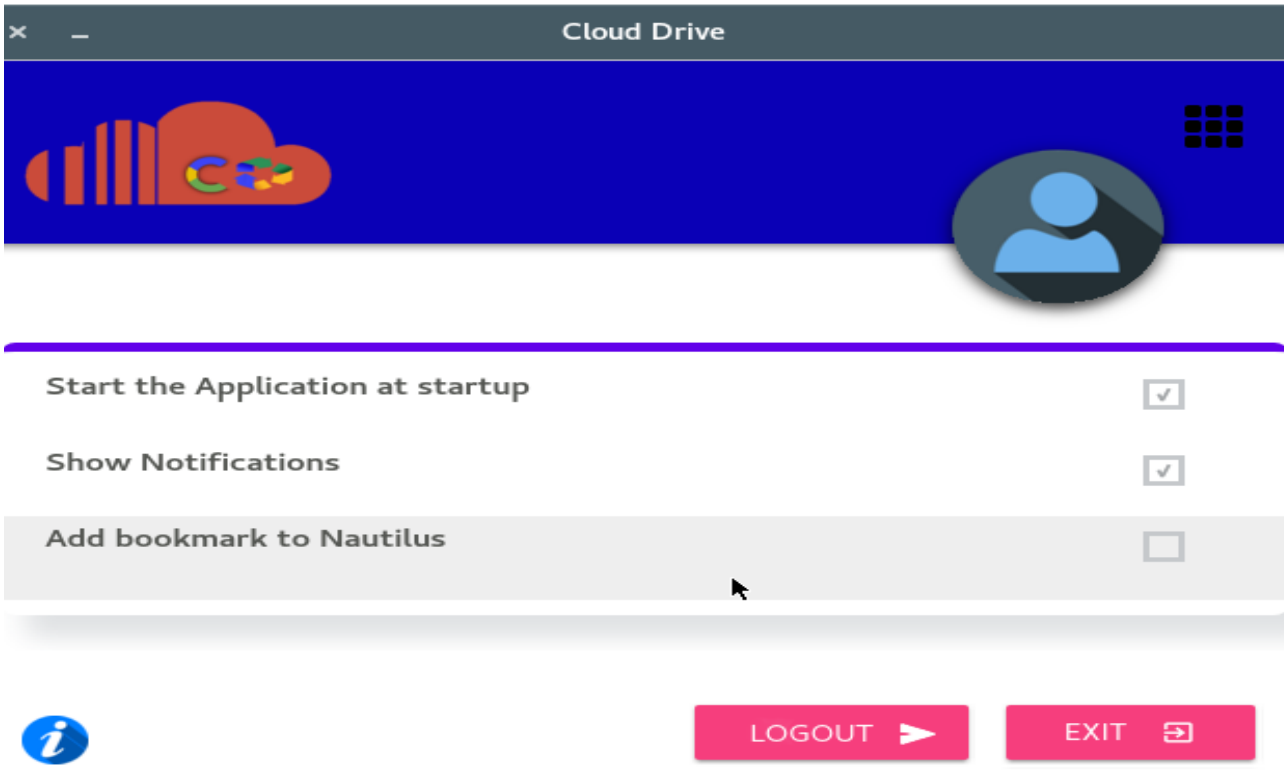

Fig. 5.15 - main_launcher

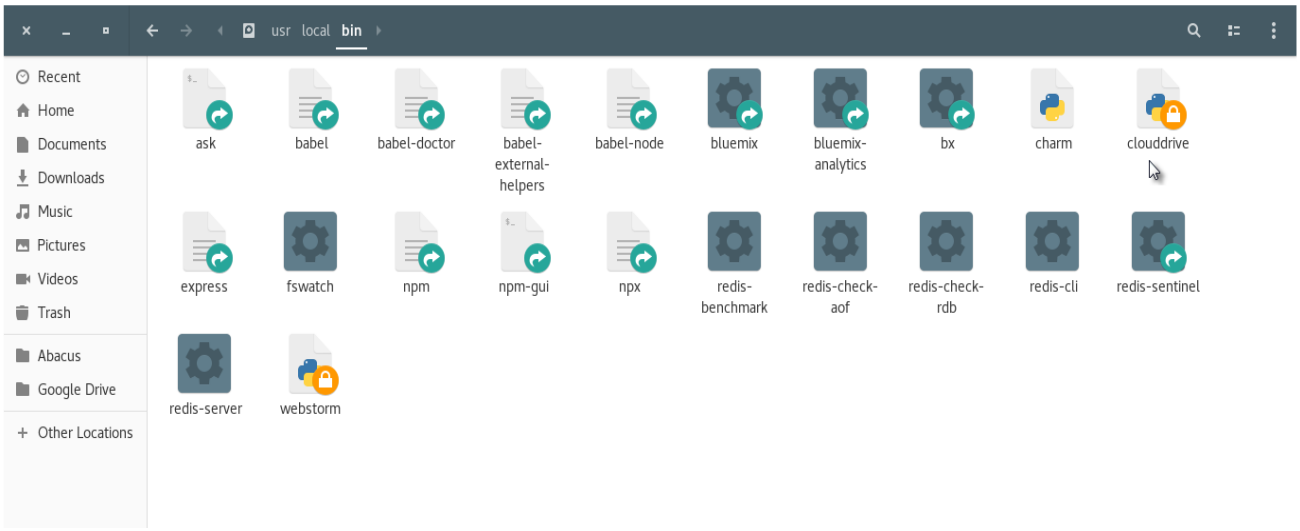

Fig. 5.16 - clouddrive_command

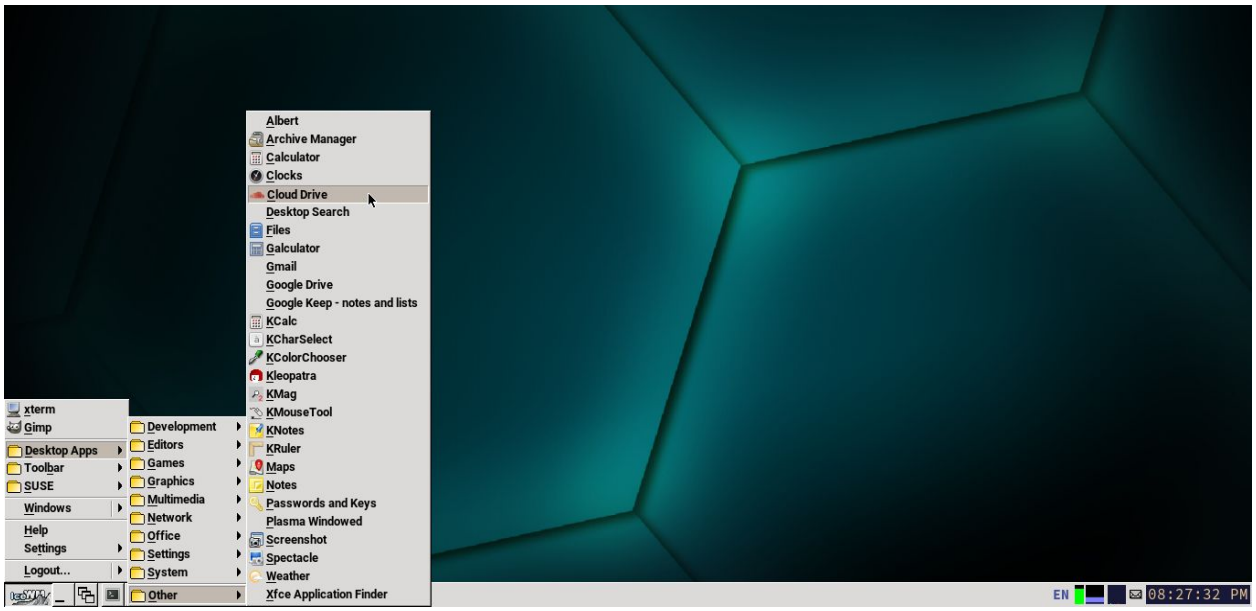

Fig. 5.18 icewm_startmenu

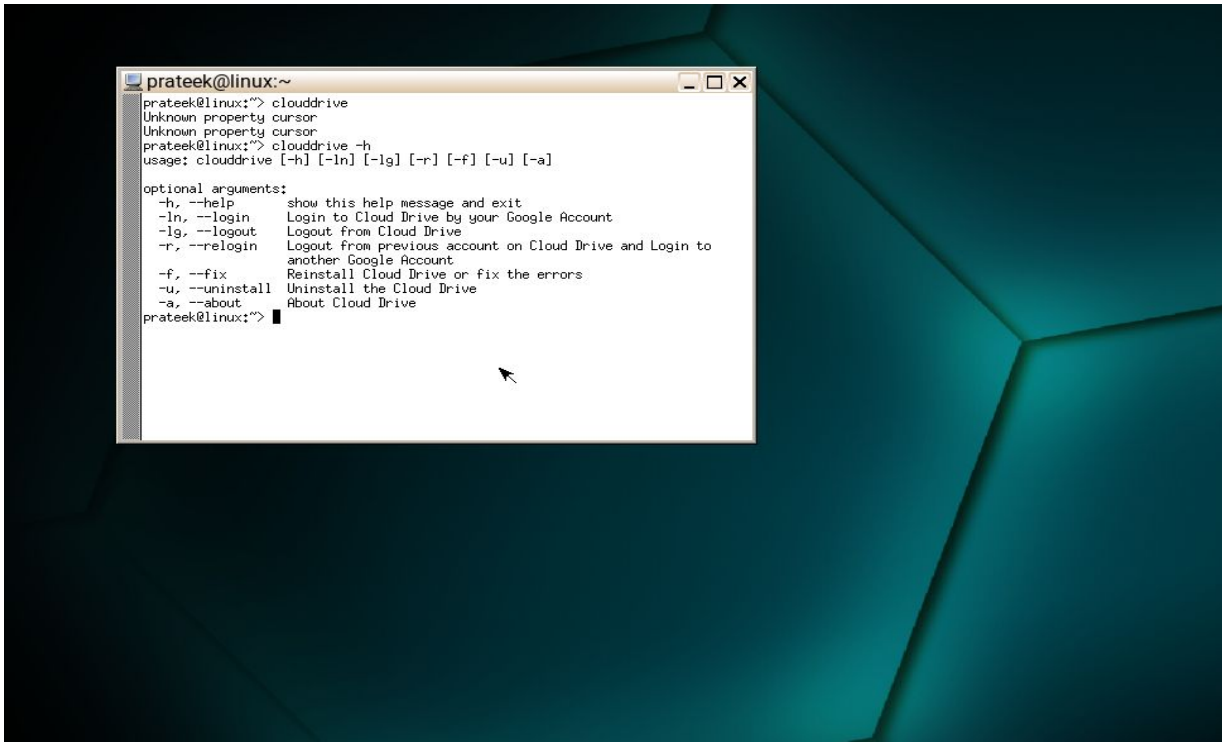

Fig. 5.17 - icewm_terminal

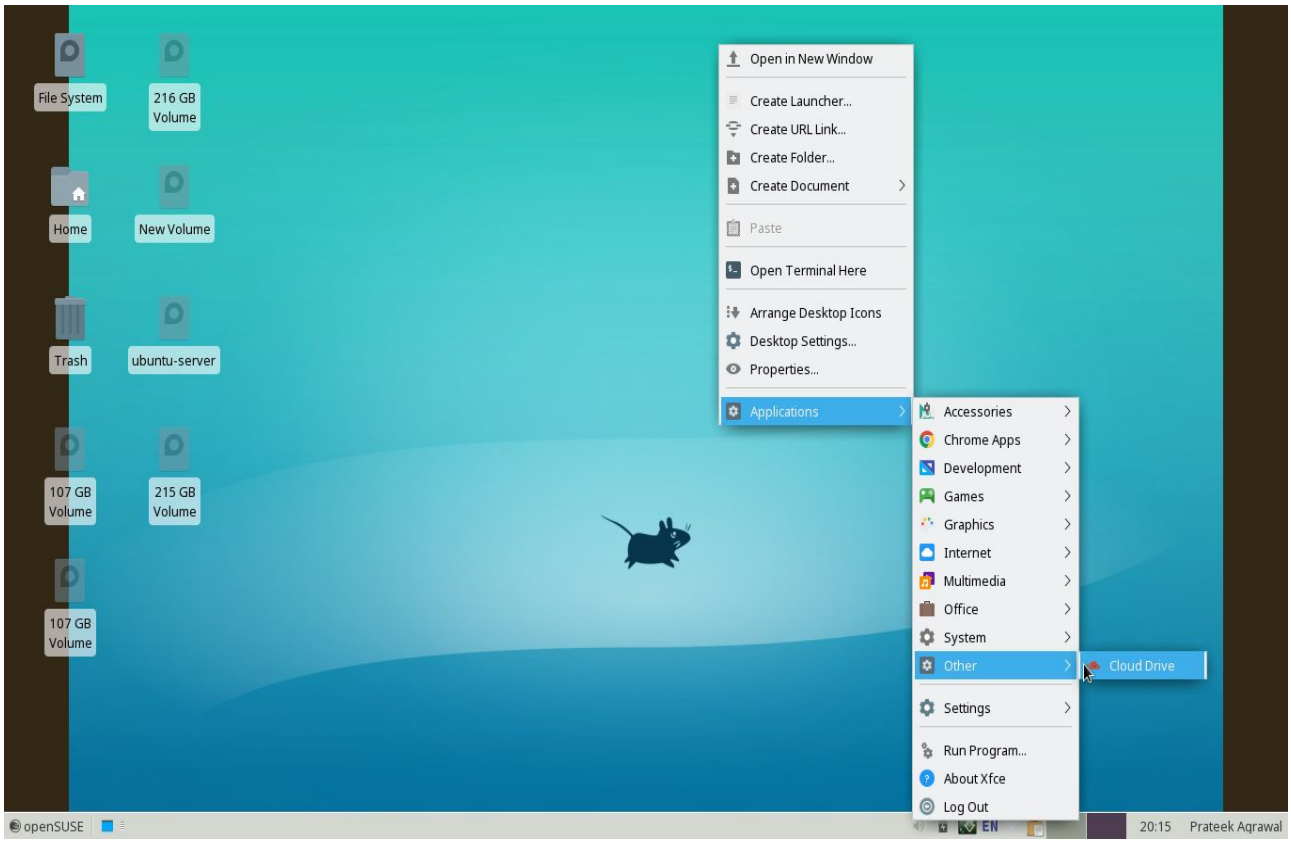

Fig. 5.18 xfce_support

6. Conclusion

This project has demonstrated an desktop application that was successfully developed using Google Drive API, PyQt5. The case study presented in this project provides the advanced functionality to manage the data, files, folders through linux system. It also provides the sophisticated functionality for searching, filtering and interface that offer the user the capability to manipulate the data.

Cloud Drive is user-friendly storage option where all test data can be loaded and is accessible from all remote locations. These provide multiple storage capacity options for your files and folders, where users can store backup local files and sync in files and folders across multiple computers.

7. References/Bibliography

https://developers.google.com/api-client-library/python/start/get_started

https://developers.google.com/drive/v3/reference/