CSIS 0801 – Final Year Project

Detailed Intermediate Project Report

Personal Dropbox

Supervisor:

- Prof. Francis Lau

Members:

- Wilson Chandra (2011544570)
- Yonatan Nugraha (2011528590)

Contents

1. Introduction 3
1.1. Background 3
1.2. Objective 3
1.3. Project Deliverables 3
1.4. Limitations 4
1.5. Future Plan 4
2. Project Methodology 5
2.1. Process Models 5
2.2. Approach and Theory 5
2.3. Development Environment 6
3. Project Progress 8
3.1. Current Progress 8
3.2. Demo Videos 23

1. Introduction

1.1. Background

Dropbox is a cloud storage service that enables us to keep our photos, docs, videos, or any other files.

- With Dropbox, we can put our stuffs on one device, and easily access them from any other devices without having to bring all of the devices every time.
- With Dropbox, it is also easy to share our files with our friends. Moreover, we can control the privacy of each file to whatever we want, and make a restriction of file editing.
- With Dropbox, our files are stored in the Dropbox server and backed up regularly, so we do not need to worry when we lose our devices.
- With Dropbox, it is very convenient to work anytime and anywhere.

However, sometimes we worry about the privacy of our files and do not want put our confidential and even our very sensitive things there. Therefore we would like to implement our own cloud storage service using our own server which is more private.

1.2. Objective

Our goal for this project is

- ❖ To implement our own cloud storage service using the CS department's server, and perhaps we will use our own PC as the server in the future, which is more private
- ❖ To make a revolution of the current cloud storages, and develop some extensions to their current functionalities.

The details of our plan will be explained in the next section.

1.3. Project Deliverables

These are some features that we would implement in our own cloud storage service.

❶ Personal Dropbox: a self-hosted cloud storage service

Cloud storage service that are widely available, such as Dropbox, Google Drive, and SkyDrive are more than enough for average needs. However, sometimes we worry about the privacy of our files, thus we do not want to put confidential stuffs there. Therefore a self-hosted cloud storage service is a good solution to make our files more private.

❷ Drop anything more than files "One-Stop Organizer"

As the current cloud storages act more like a traditional folder, which allows us to put any kinds of files into them, we would make our own implementation which is capable of organizing anything more than just files, such as contact lists, notes, and emails, and to make it more in line with the current mobile devices trend, e.g: after buying a new phone, we can use this new

feature to transfer the contact list from our old phone into the new one. The idea is to provide one central source for the storage and management of our information.

❸ Synchronize files and folders from outside of the cloud storage folder

The original implementation of cloud storages only allows synchronizing to a single designated folder on a device. Sometimes when we want to synchronize some data which we don't want to move from the original location, the only choice we have is to copy the file into that designated folder, which takes a lot of space. Another bad thing is that often the duplicate of the file is out of date with the actual file.

To solve this inconvenience, we plan to synchronize the cloud storage into the other folders, such that any change made to the cloud storage will be applicable to that folders as well.

❹ Synchronize to Social Media

Uploading a bulk amount of photos or videos to social media, especially Facebook can be quite annoying. When we accidentally close your browser, the upload process stops, and often we have to see among the uploaded pictures and manually re-choose the pictures to be uploaded, and then restart the upload. One idea is to let a server do the upload for us.

❺ Portable applications on the cloud storage

Borrowing another devices can be a great solution when we need for a quick use to take a peek on the data we have online. However, what if we can’t view our data because the application is not installed on the device we borrow. What if the application is installed but not configured the way we like. Portable applications come to the rescue in this situation. Portable applications do not need installation and the settings are stored on the cloud storage. So, no matter what computer we use, the application state is the same across devices.

1.4. Limitations

- Only one person can use an account at a time. The system does not support multiple users using the same account at the same time.
- Security is not considered at this stage.
- Portable applications must be designed to be portable. This means not all applications can be run in portable mode.

1.5. Future Plan

The following idea will probably not included in our project, however, we may add it in the future.

- ✗ Expand the organizing capability, such as adding the online documents editor, which supports various documents format.
- ✗ Expand the portable applications capability, such as normal applications can run in portable mode.

2. Project Methodology

2.1. Process Models

We will divide our project into several phases according to the new features we plan. Each phase can be implemented separately.

❶ **Phase 1.** Implement the self-hosted cloud storage functionalities.

In this phase, we will implement the self-hosted cloud storage functionalities in two different platforms: web based application and PC Client.

❷ **Phase 2.** Drop anything other than files.

Our primary focus on this phase is to make our cloud storage capable of supporting these three formats: notes, contact lists, and emails. For this phase, we will implement it in two different platforms as well: web-based application and PC client.

❸ **Phase 3.** Synchronize files and folders from outside of the cloud storage folder.

Our primary focus here is to synchronize files and folders located outside of the designated cloud storage folder. For this phase, only PC client is implemented.

❹ **Phase 4.** Synchronize to Social Media - Facebook.

Our main focus here is to synchronize our cloud storage with Facebook, as there are more users using it compared to another Social Media. For future development, we will make it work with another Social Media such as Twitter.

❺ **Phase 5.** Portable applications on the cloud storage.

The main focus here is to host portable applications on the cloud storage. On this stage, only PC client is implemented. Future development can make the application work on the web.

❻ **Phase 6.** Testing and bug fixing.

This last phase will include testing for all of the previous phases.

2.2. Approach and Theory

The implementation philosophy of this project is to use common features that is available on average web hosting, so that this system can run in average web hosting scenario, and thus is accessible to those who are not willing to let his/her computer act as a server. This approach also enables a user to access the cloud storage using a "clean install" windows in a native way using windows explorer.

Unlike the openly available cloud storage server such as Seafile or Sparklesare, and even the premium software such as Microsoft SharePoint, which requires a very specific configuration, and often even a dedicated server in which average web hosting can't provide. This system runs well in a typical web hosting scenario.

❶ **Phase 1.** Implement the self-hosted cloud storage functionalities.

PC Client
First Implementation: Run SAMBA/FTP on server and map it to a drive letter.
Second Implementation: Use a cache folder, which a synchronizing program run at an interval of time to synchronize with the server. Further implementation can be hooking to the write process, and synchronize every time a write is detected to the file being synchronized.

Web based application
Receive the file and store it in the server. Make a user interface to read files from the server.

❷ **Phase 2.** Drop anything other than files.

We are going to make a web and PC application which can read the contents from server using file API (an API for representing file objects in web applications, as well as programmatically selecting them and accessing their data).

❸ **Phase 3.** Synchronize to other than cloud storage folders.

For this phase, we make use of an operating system feature, which is called “symbolic link”.

*A **symbolic link** is a special type of file that contains a reference to another file or directory in the form of an absolute or relative path and that affects pathname resolution.*

We make a symbolic link to another folder inside the Cloud Storage folder. This link is treated as a “real” folders and files by applications, however taking a very small space as the “link” is actually a pointer to the real file.

❹ **Phase 4.** Synchronize to Social Media.

We put photos on the cloud storage folder. And then after the photos are uploaded to the server, the server will do the upload to Facebook server.

❺ **Phase 5**. Put portable applications on the cloud storage.

We design an application in which the application will store data to the cloud storage folder instead of local PC. We can also write a companion web application to read the data from web, once we step to the next development stage.

2.3. Development Environment

Web based Application	
Language	**PHP**
Database	**SQLite**
Tools	**Symfony Framework, SQLite, PHP Storm**
Server	**Apache with PHP and SQLite**

PC Client	
Language	**Java/C++**
Database	**SQLite**
Tools	**SQLite, Visual Studio, intelijIDEA**
Server	**Java Runtime, .net Framework**

3. Project Progress

3.1. Current Progress

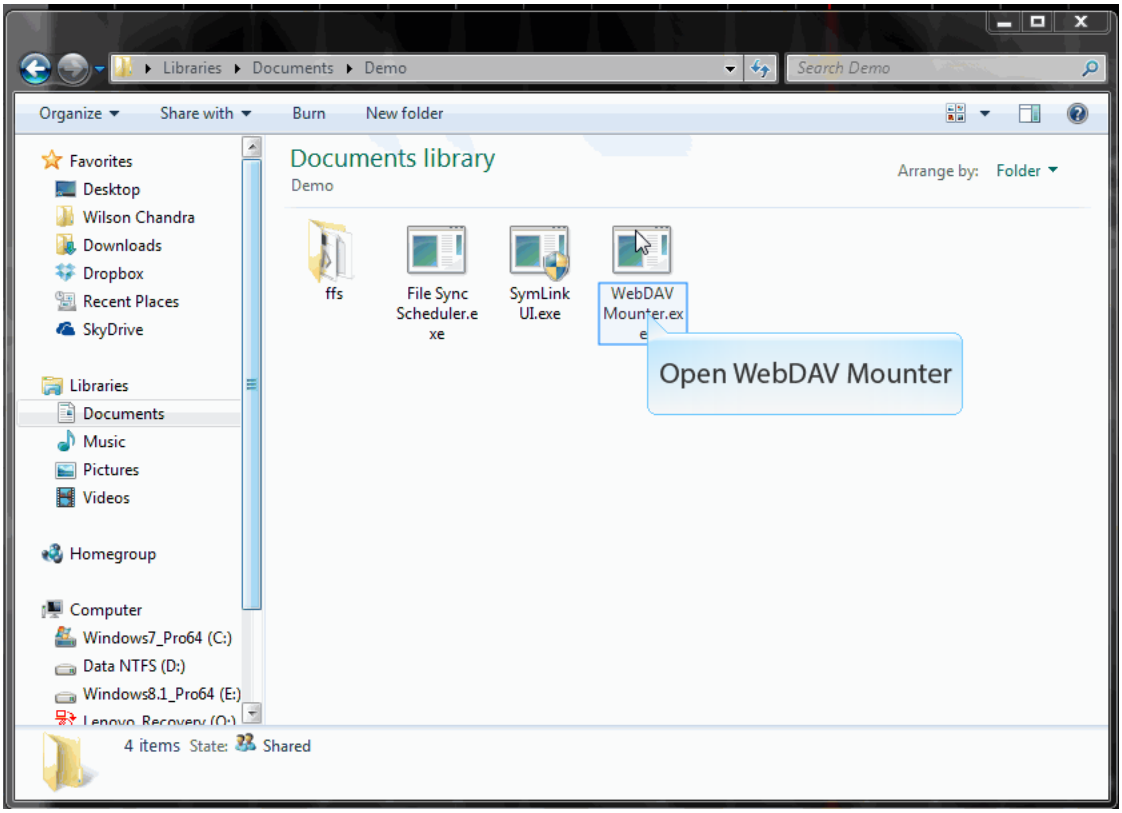
We have completed some phases and made demo videos. Please note that the purpose of this demo is to show the technical preview of this project, demonstrating the backend features used. The final version will include improvements for user-friendliness and can be different from this implementation.

❶ **Phase 1.** Implement the self-hosted cloud storage functionalities.

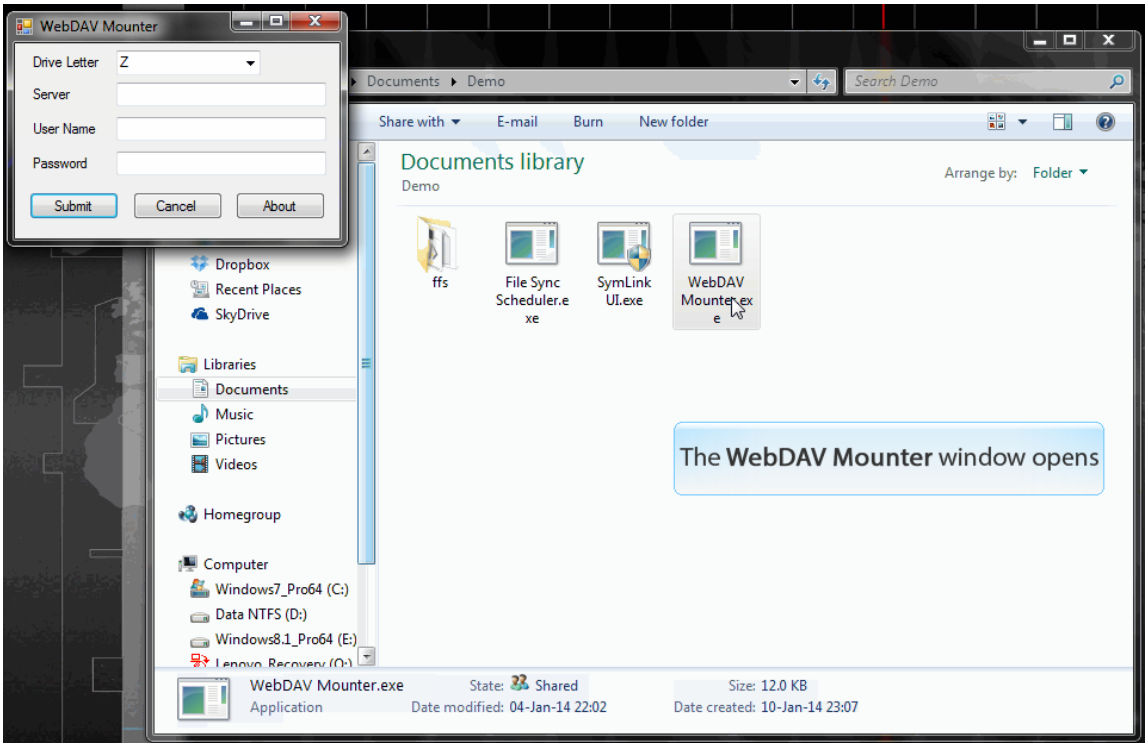
This section uses WebDAV to access the cloud storage. WebDAV is supported natively by Windows Explorer, and thus this program mounts the WebDAV into drive letter.

Here are some screen shoots of the demo video we have made.

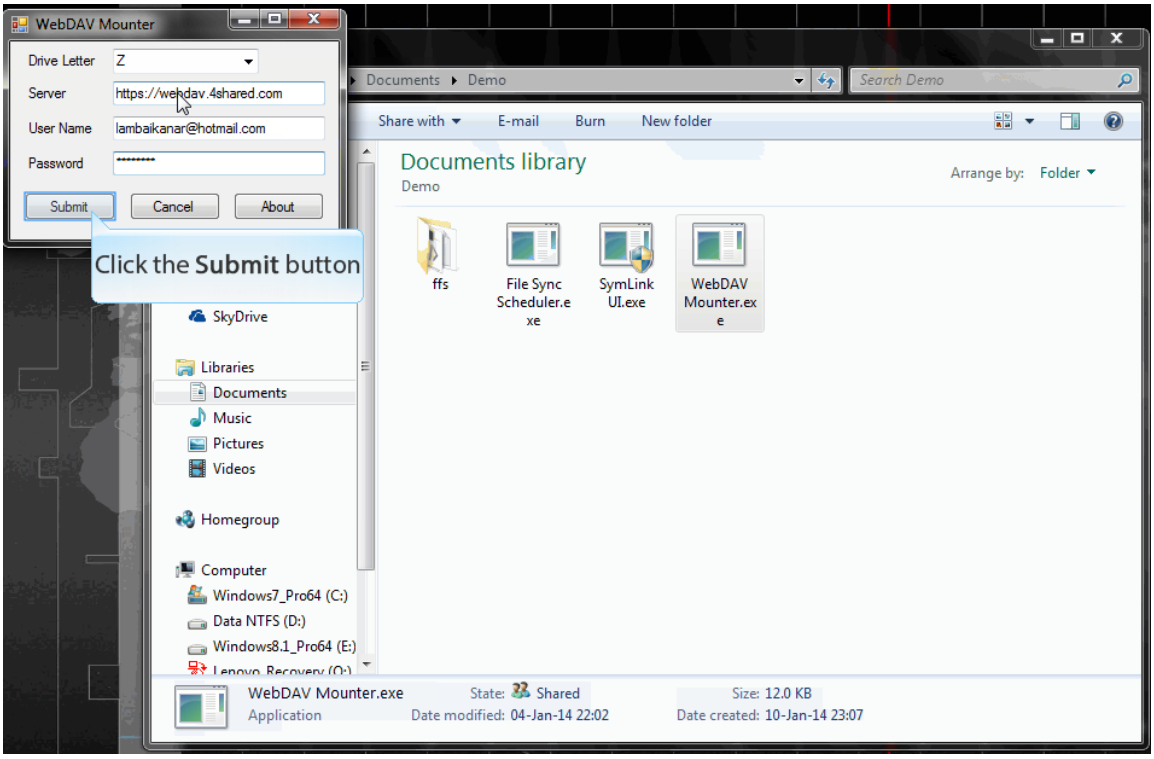
1. Open WebDAV Mounter



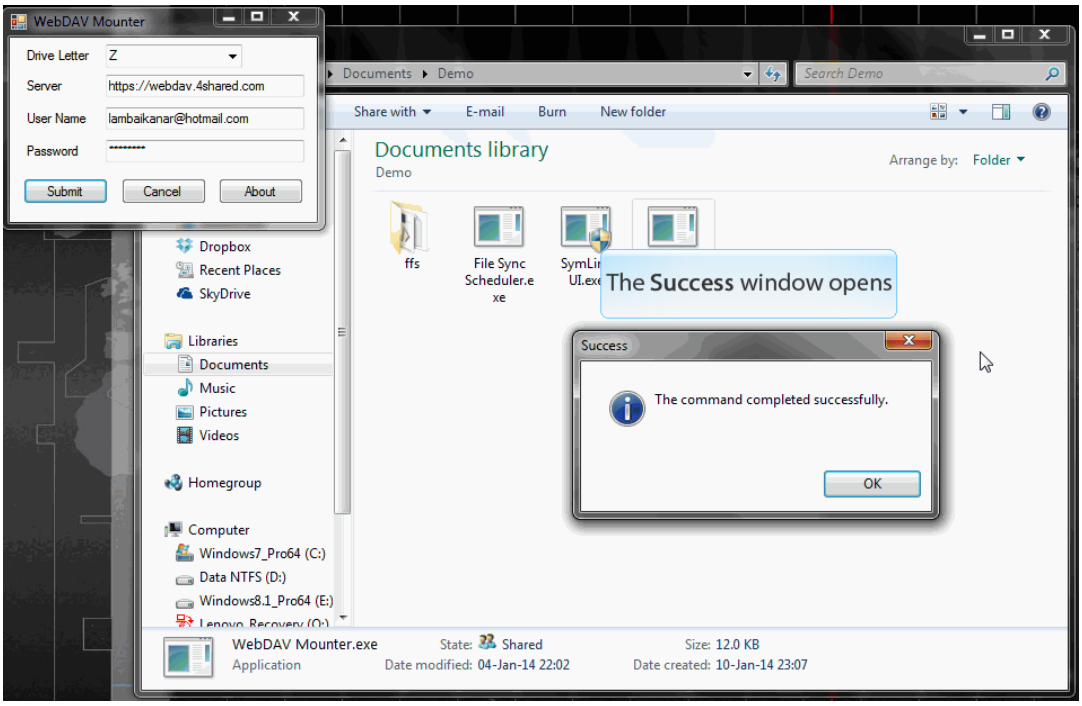
2. WebDAV mounter windows opens.



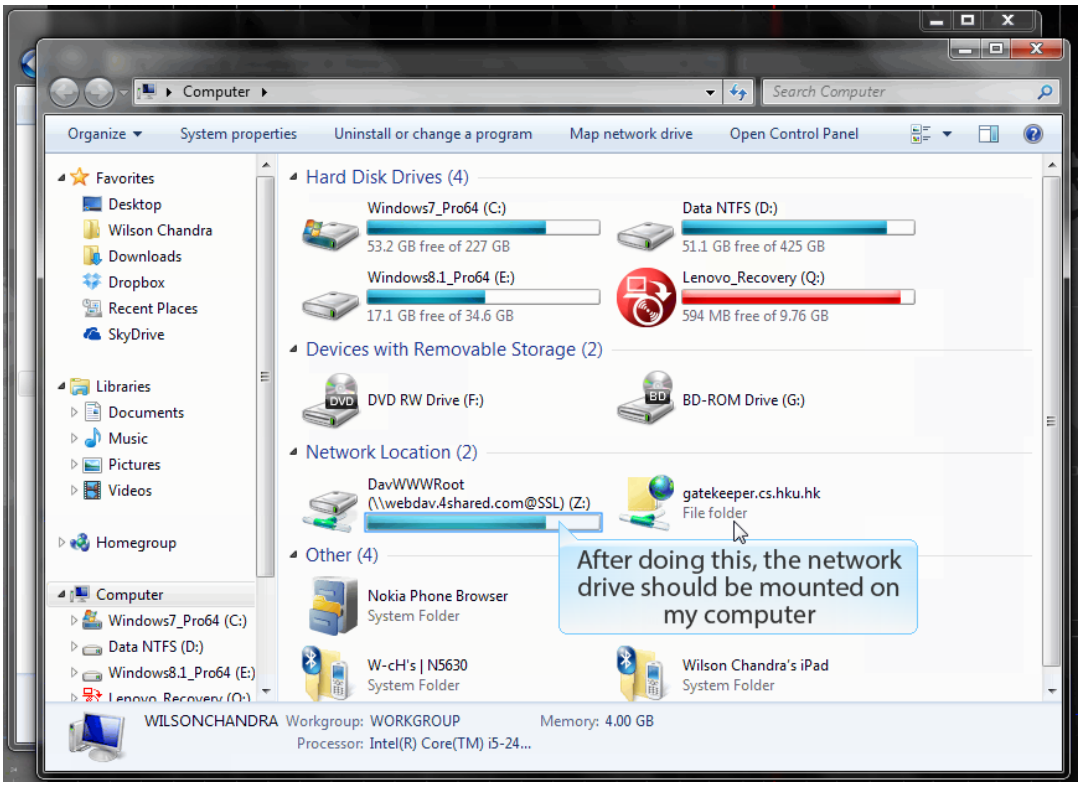
3. Type in the server name, user name, and password of the server we would synchronize with. In this demo, we are going to use 4shared as our server



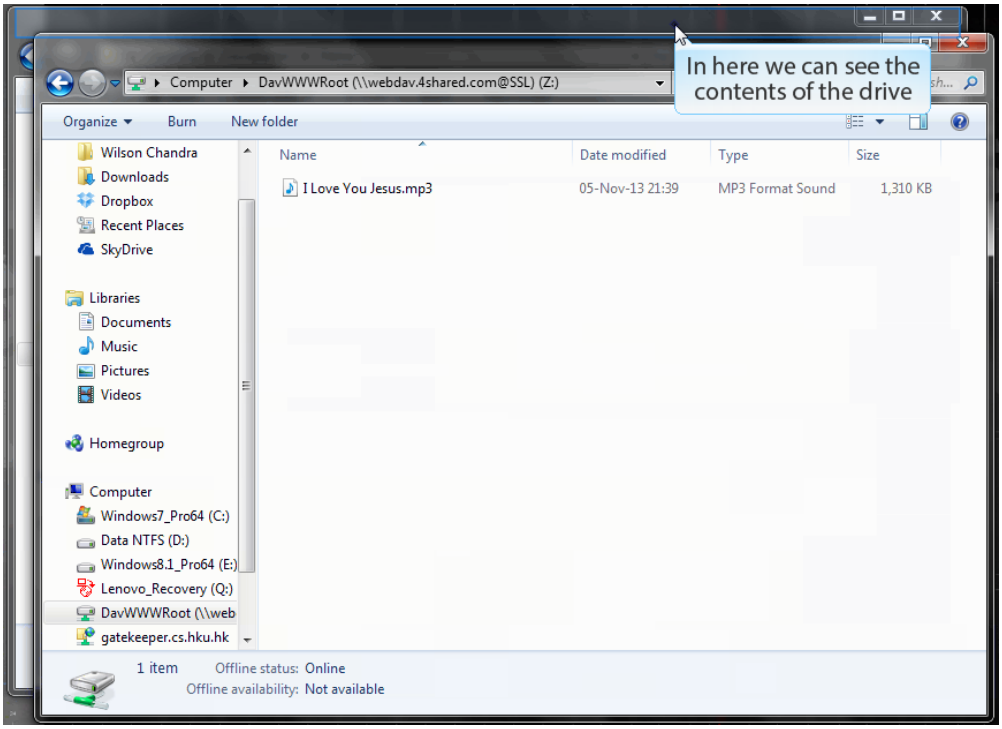
4. After clicking submit button, command prompt windows will be opened shortly and the command is completed successfully.



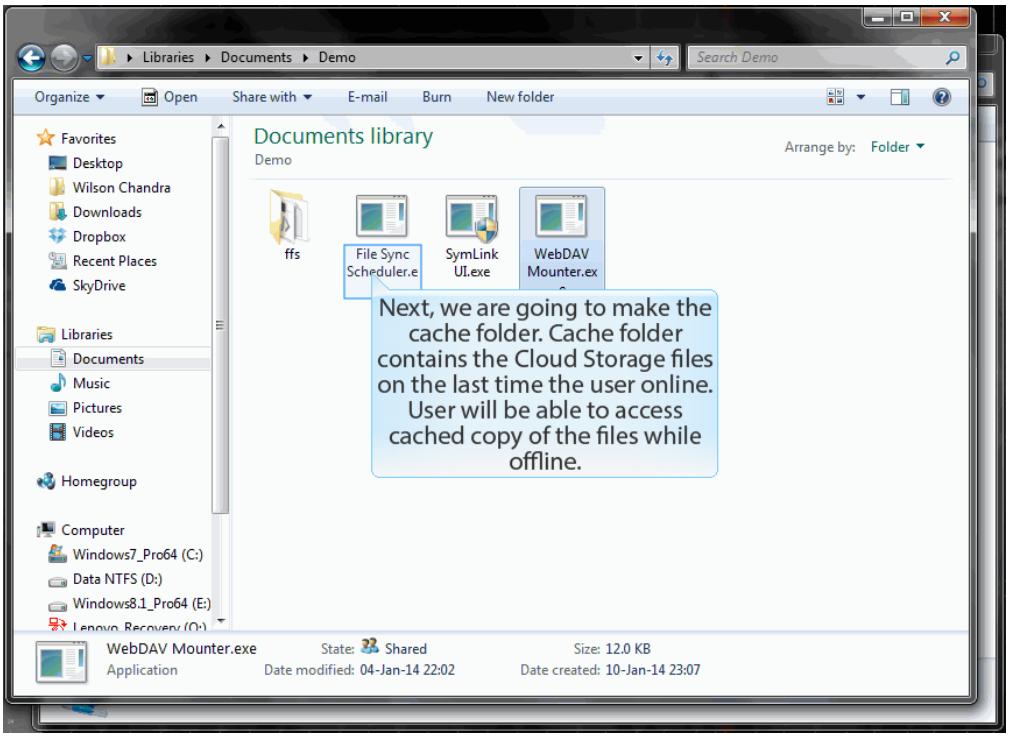
5. After executing the command on the WebDAV mounter, the network drive should be mounted on the computer



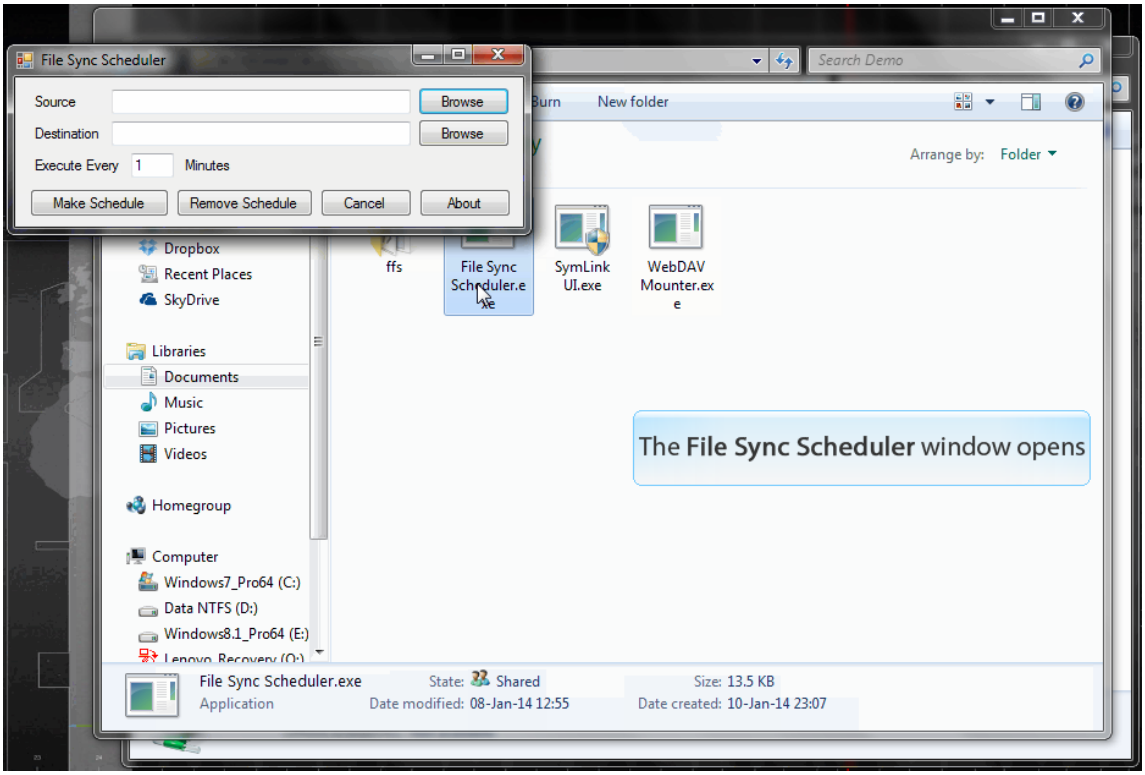
6. And we can see the contents of the drive.



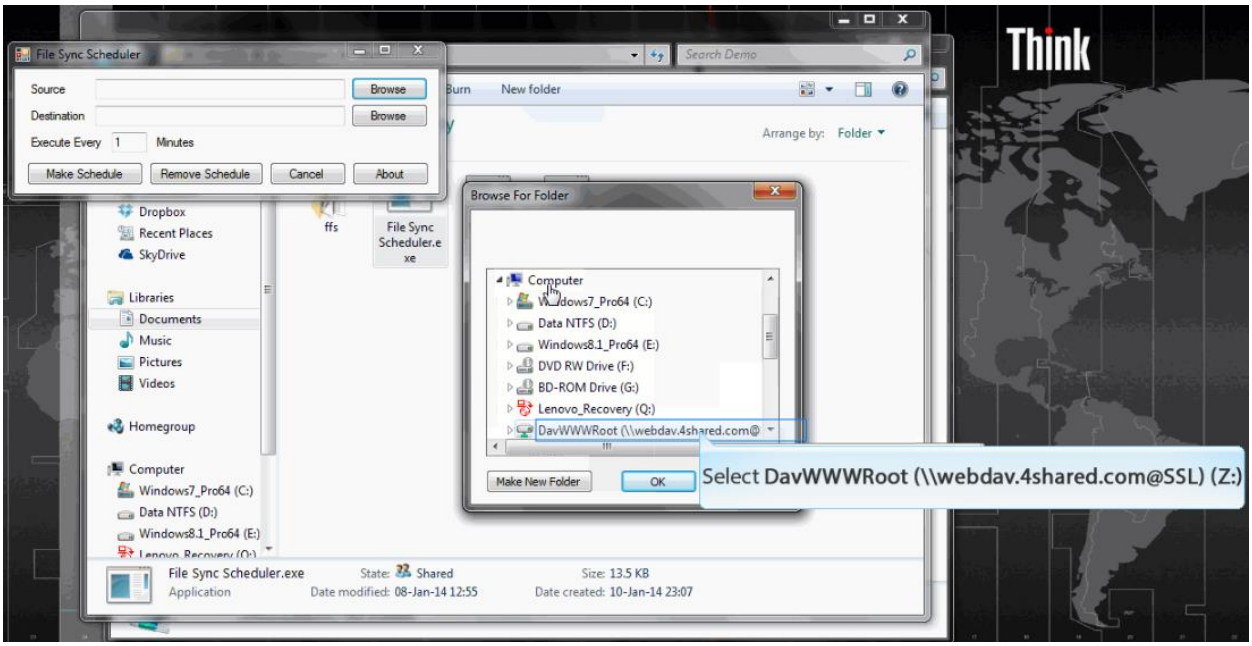
7. Next, we are going to make the cache folder. In this scenario, we use FreeFileSync as the backend for the cache folder synchronization. This program uses FreeFileSync to synchronize between the Cloud Drive and Cache Folder, and runs the synchronization task at the specified interval.



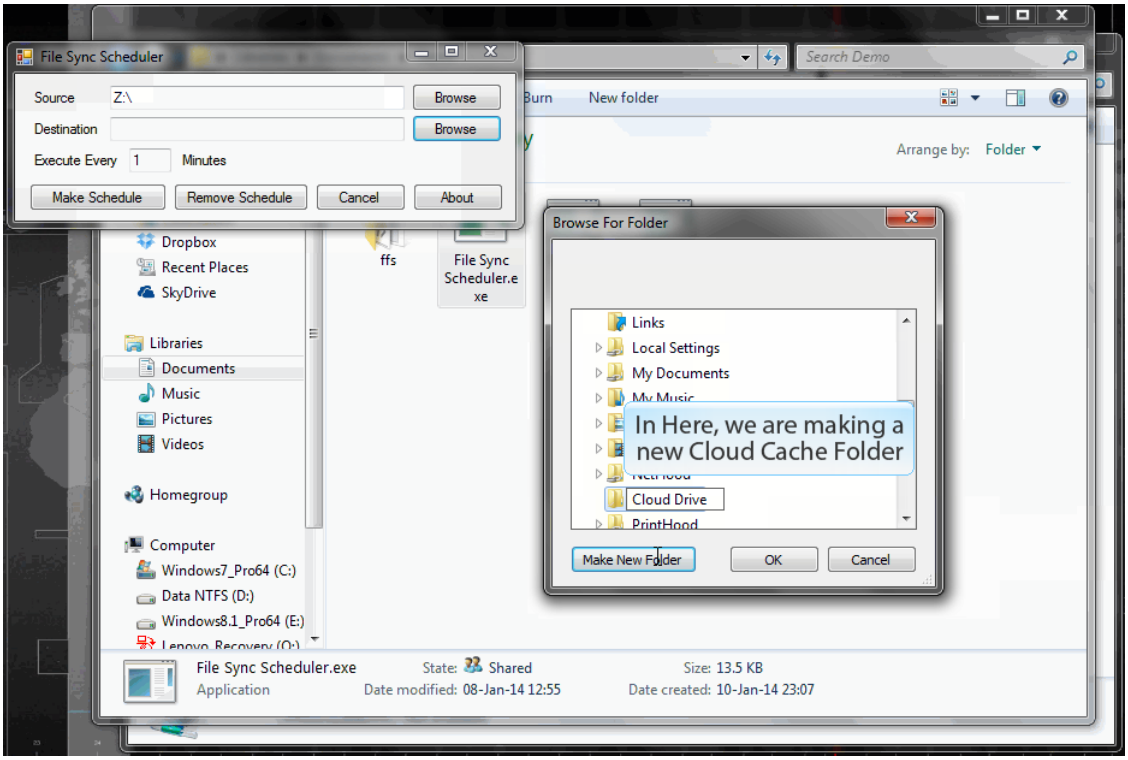
8. Open the File Sync Scheduler



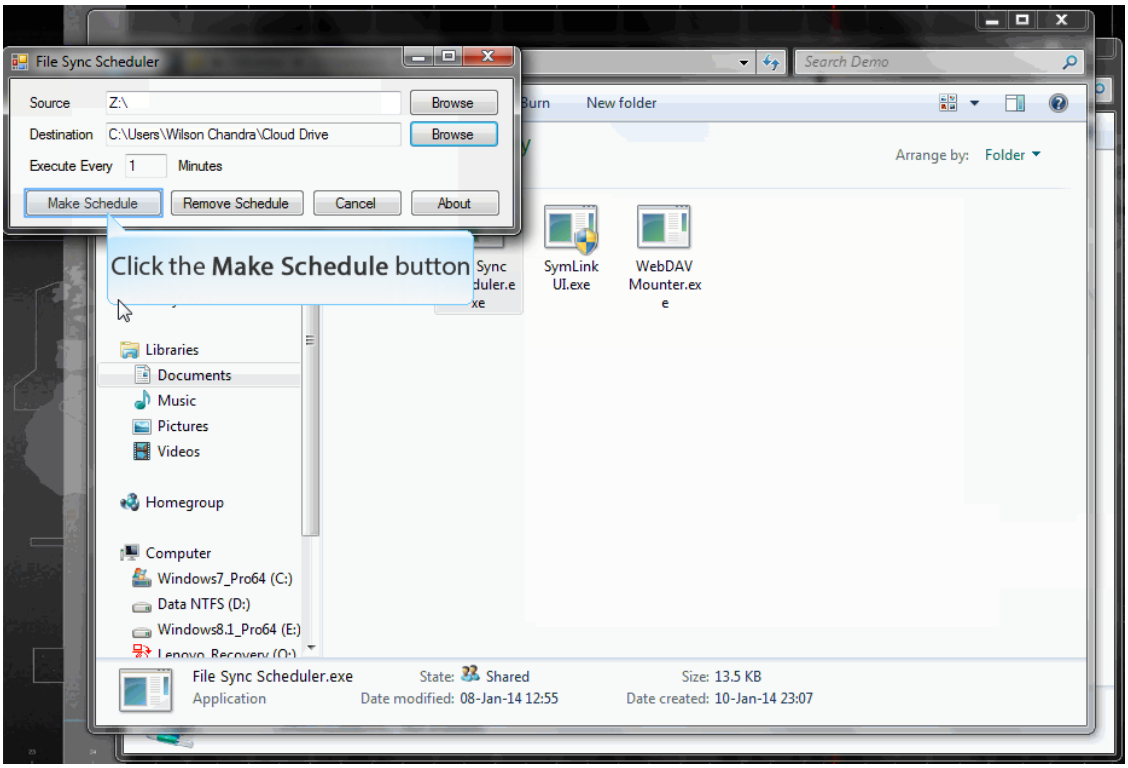
9. For the source, browse for DavWWWRoot (\\webdav.4shared.com@SSL) (Z:)



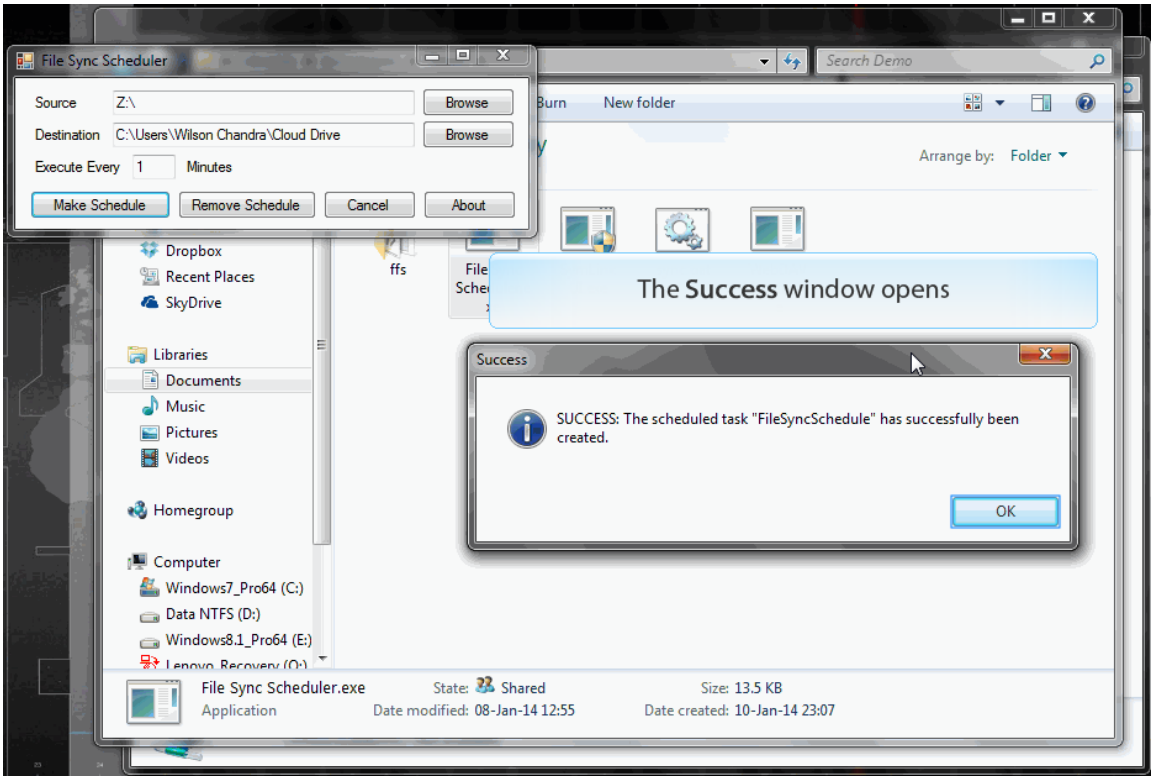
10. For the destination, we are going to make a new Cloud Cache Folder and browse to that folder



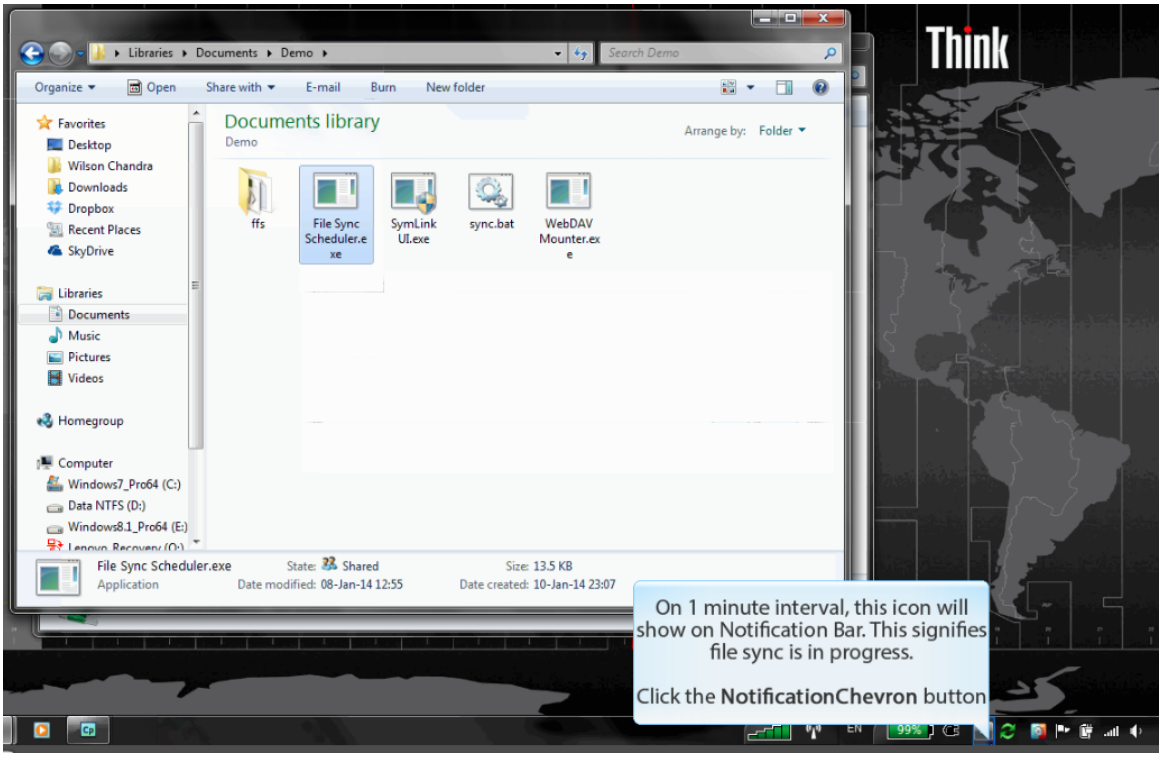
11. After selecting the source and destination path, type in the time for the execution of the program and click the make schedule button.



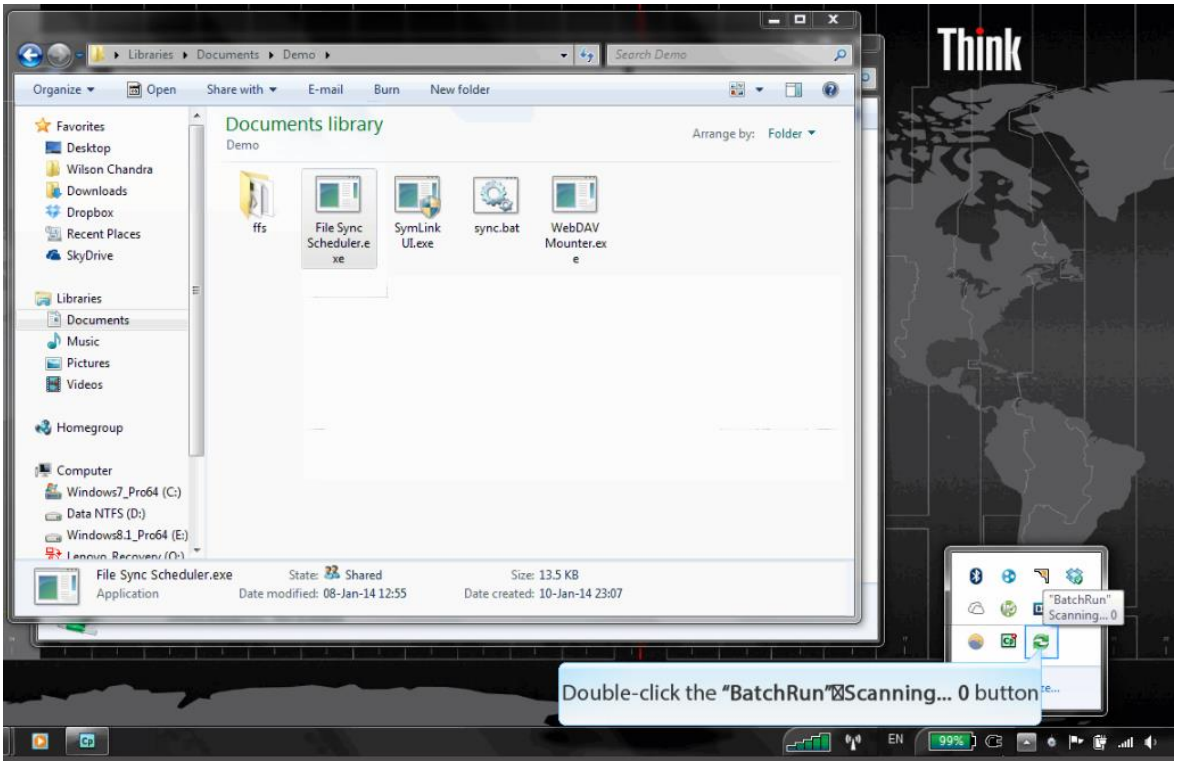
12. The scheduled FileSyncSchedule has successfully been created.



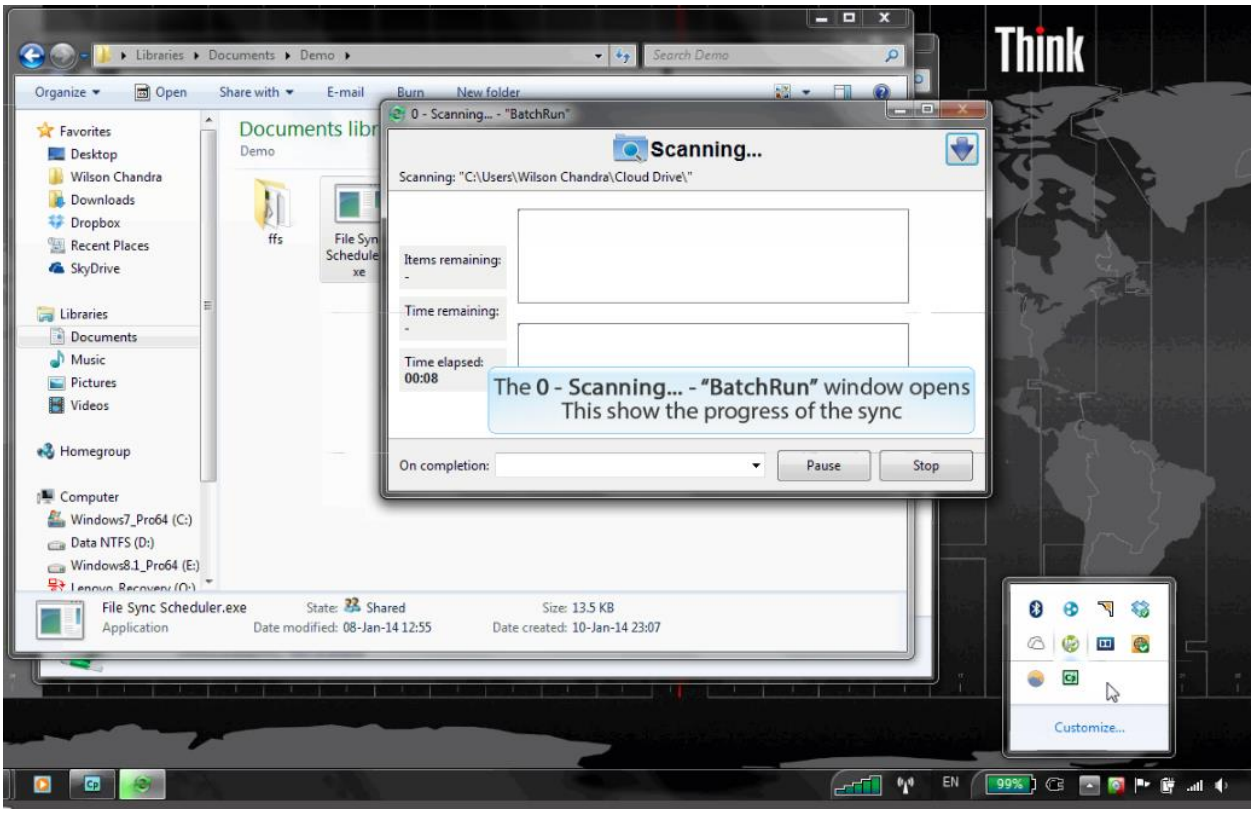
13. On 1 minute interval, an icon will show on the Notification Bar, signifying that file sync is in progress. Click the NotificationChevron button.



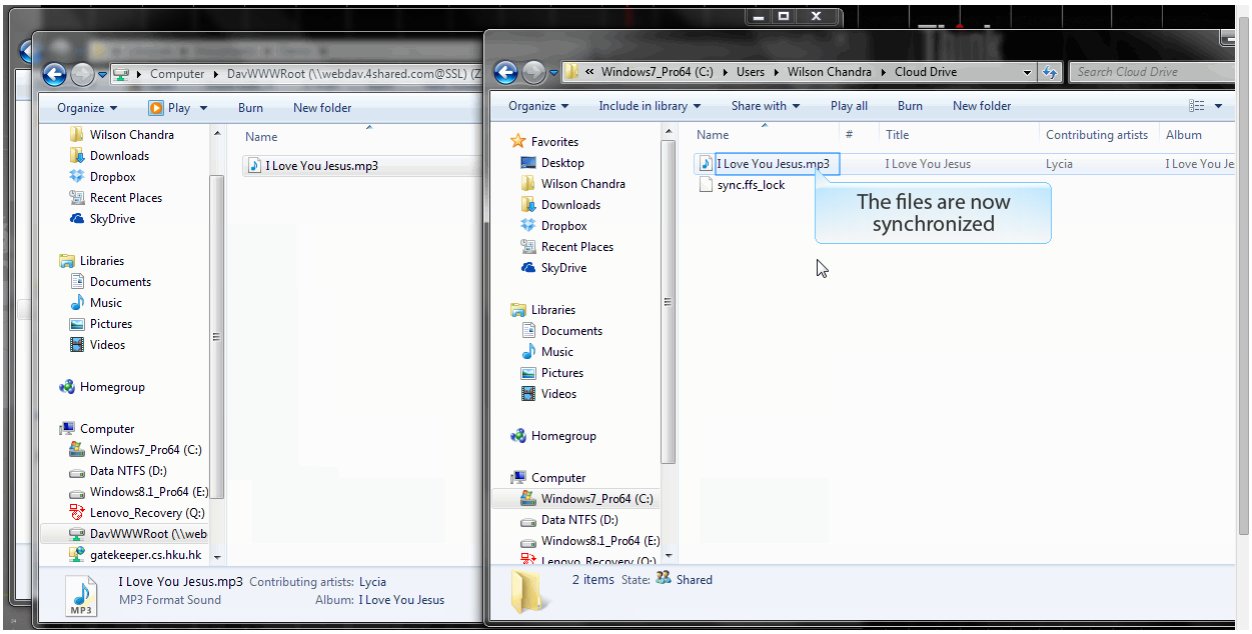
14. Double-click the BatchRun Scanning button



15. The windows opens. This shows the progress of the sync.



16. The files are now synchronized. The contents of the Cloud Storage and the Cache Folder are the same.



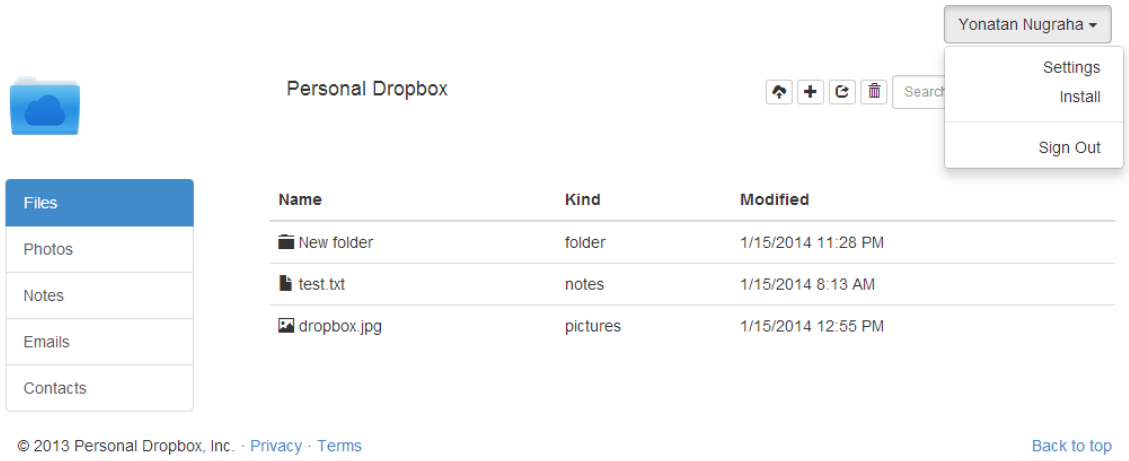
❷ **Phase 2.** Drop anything other than files.

In this phase, we are going to make a web and PC application which can read the contents from the server using file API. File API is an API for representing file objects in web applications, as well as programmatically selecting them and accessing their data.

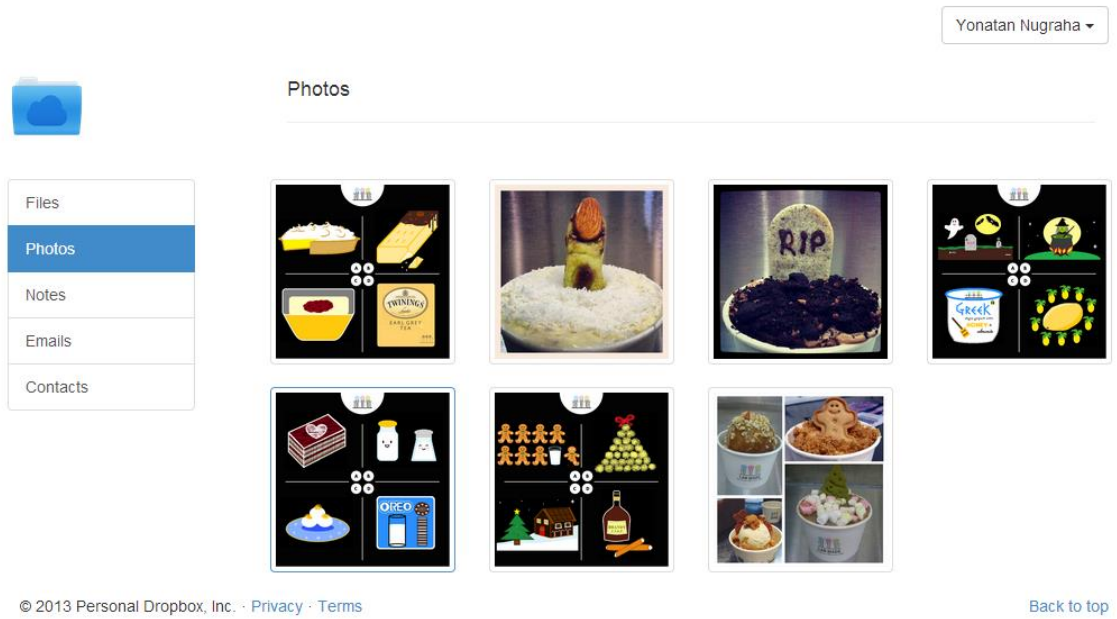
We have done the Web UI prototype for all of the contents we are planning to make, including the new features such as notes, emails, and contacts. Now, we are still working on how to synchronize the files and folders on the server into the web and PC application

Here are some screen shoots of the Web UI prototype.

Web UI Prototype – Files

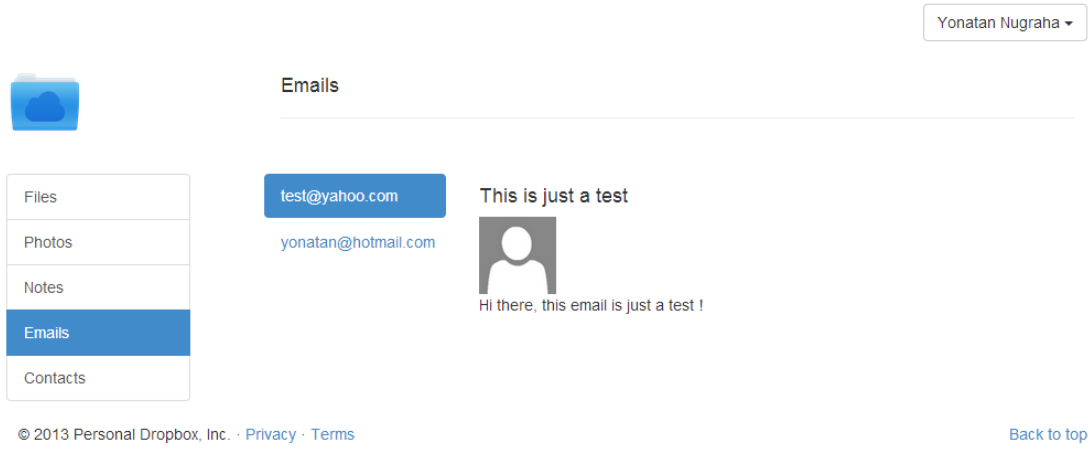


Web UI Prototype – Photos

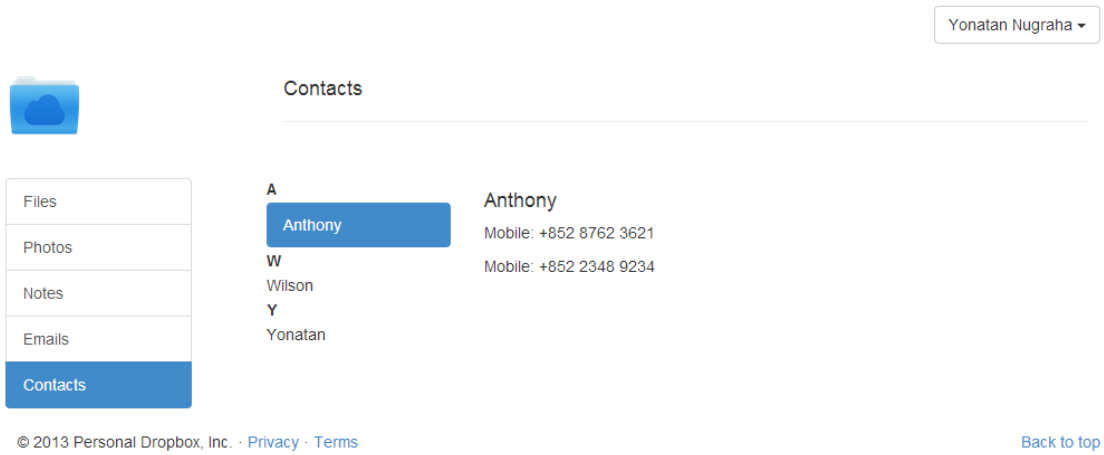


Web UI Prototype – Notes

Web UI Prototype – Emails



Web UI Prototype – Contacts



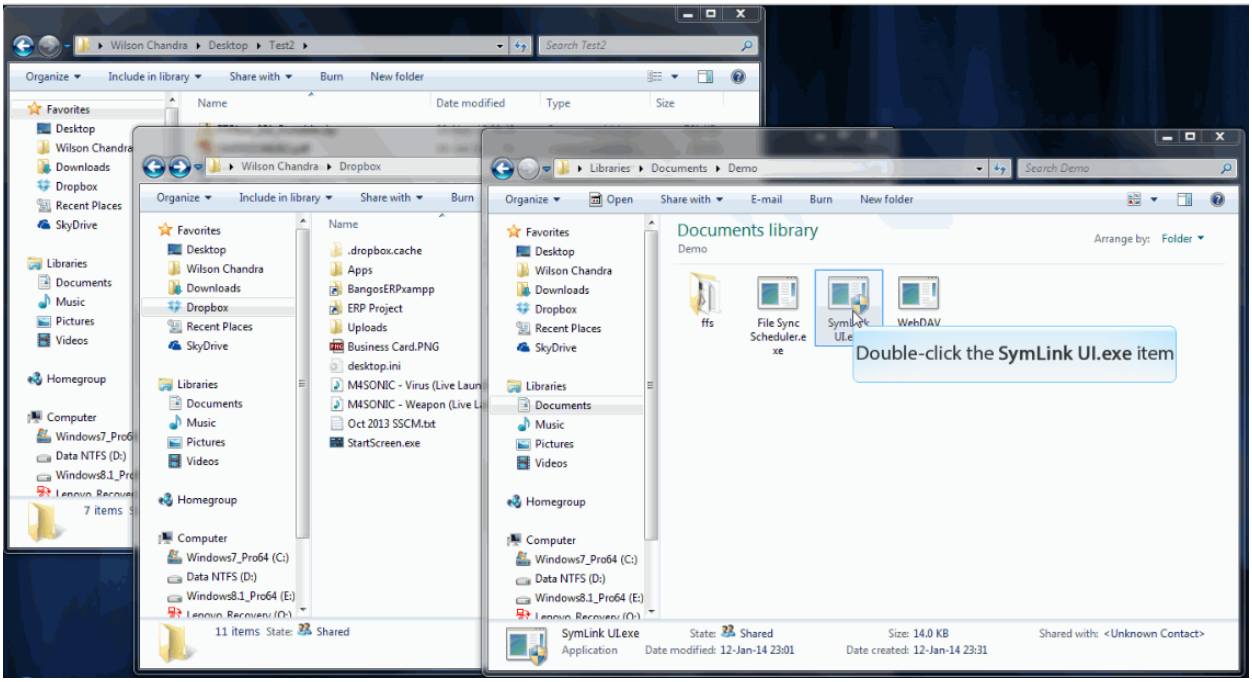
❸ **Phase 3.** Synchronize to other than cloud storage folders.

In this phase, we have made the program to synchronize folders located outside of the cloud storage folders itself. This program uses Symbolic Link feature which is available within Windows. In this way, not only the cloud storage client program recognize this, but all programs will also recognize this.

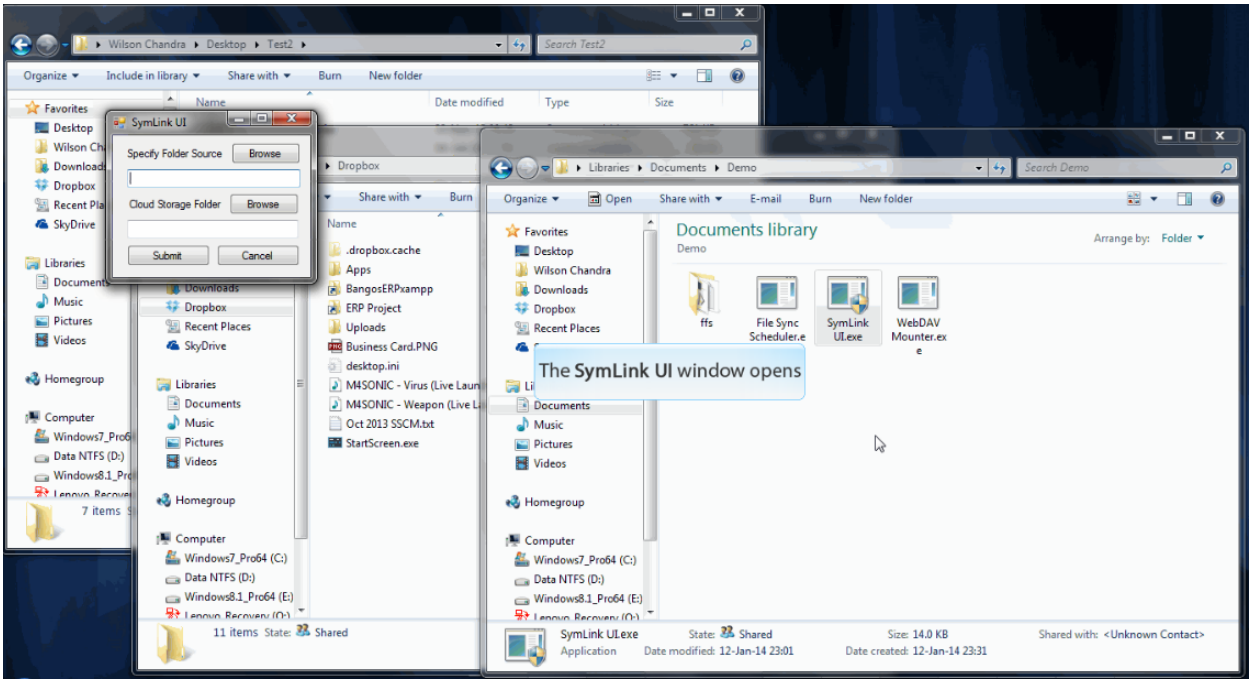
Note that this program can be used with any cloud storage services, it can be also used with our implementation of Personal Cloud Storage.

Here are some screen shoots of the demo video we have made.

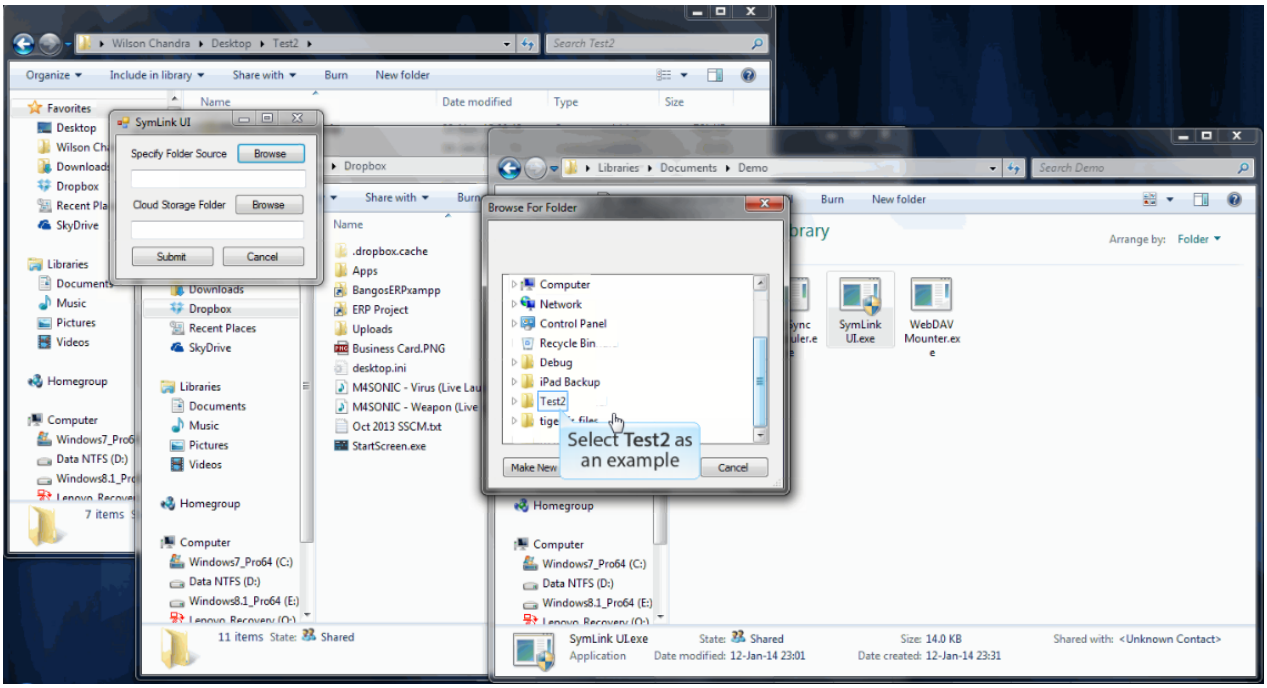
1. Double click the SymLink UI.exe



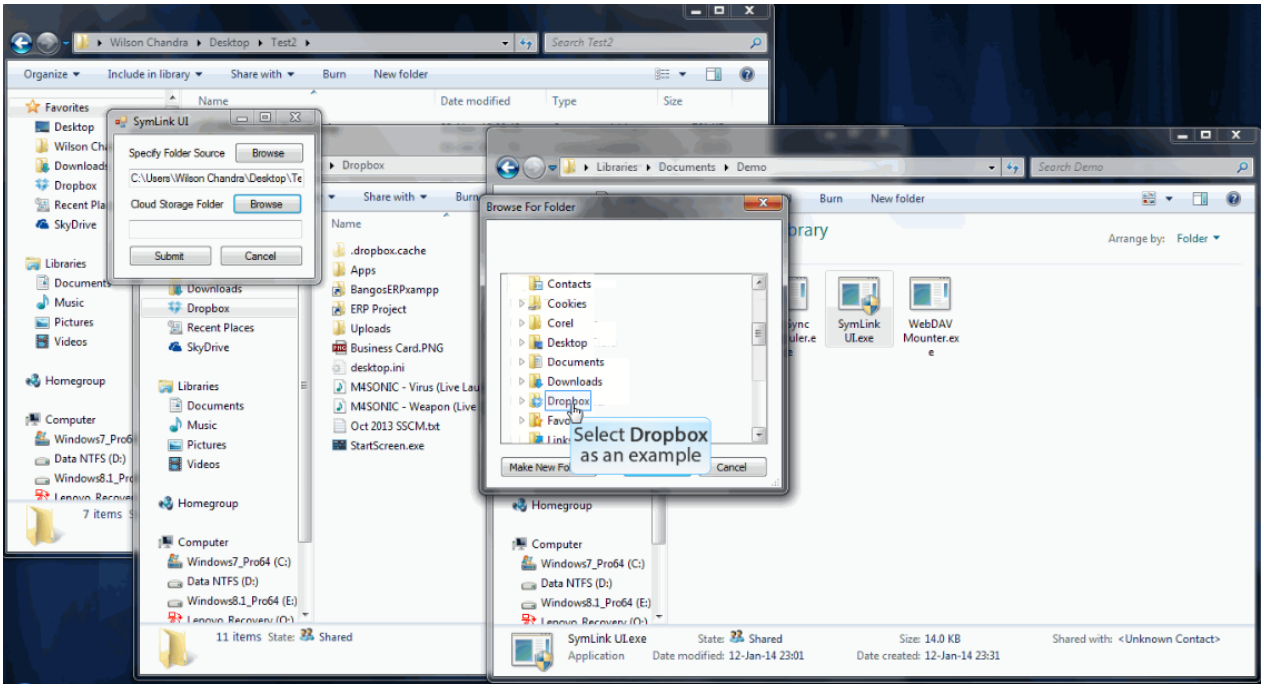
2. The SymLink UI windows opens



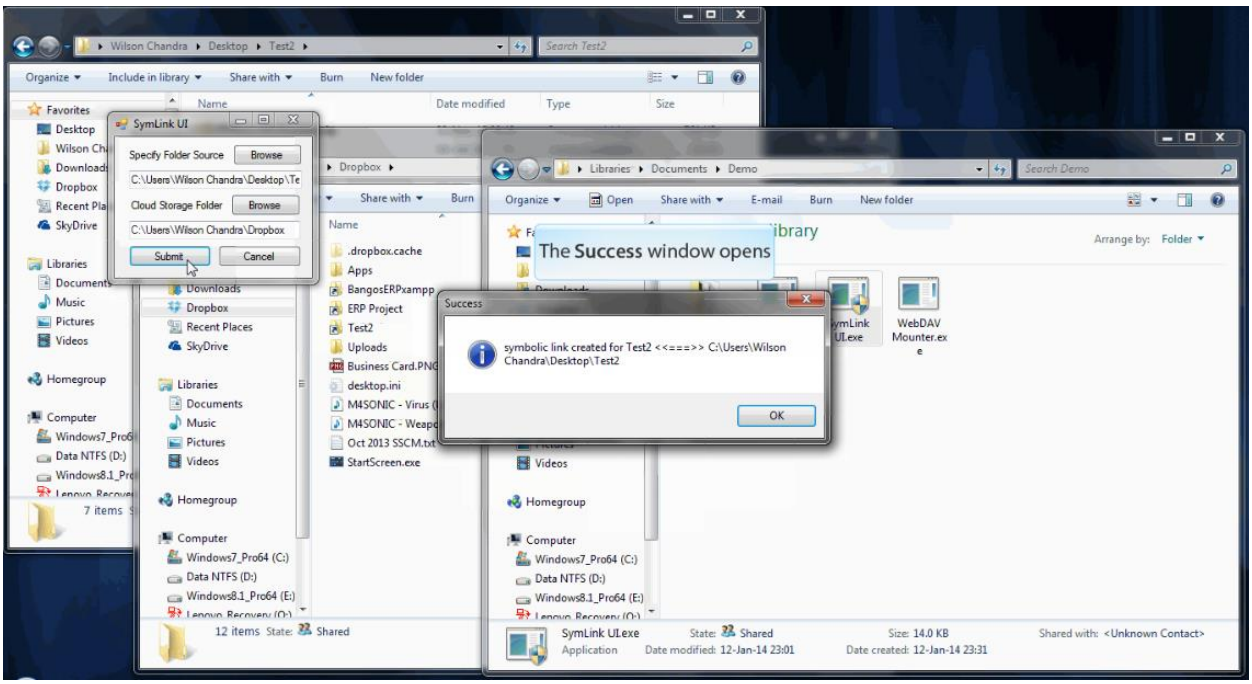
3. Browse to locate the folder to be synchronized and select folder Test2 as an example.



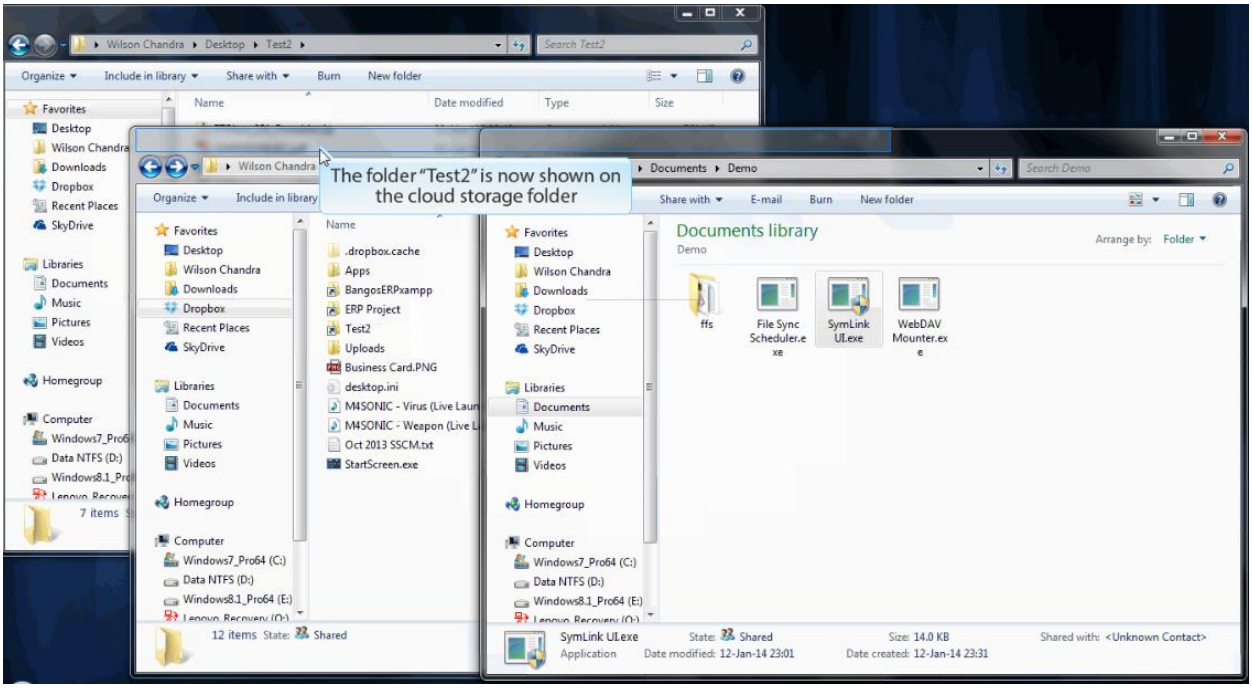
4. Browse the location of the cloud storage folder and select Dropbox as an example.



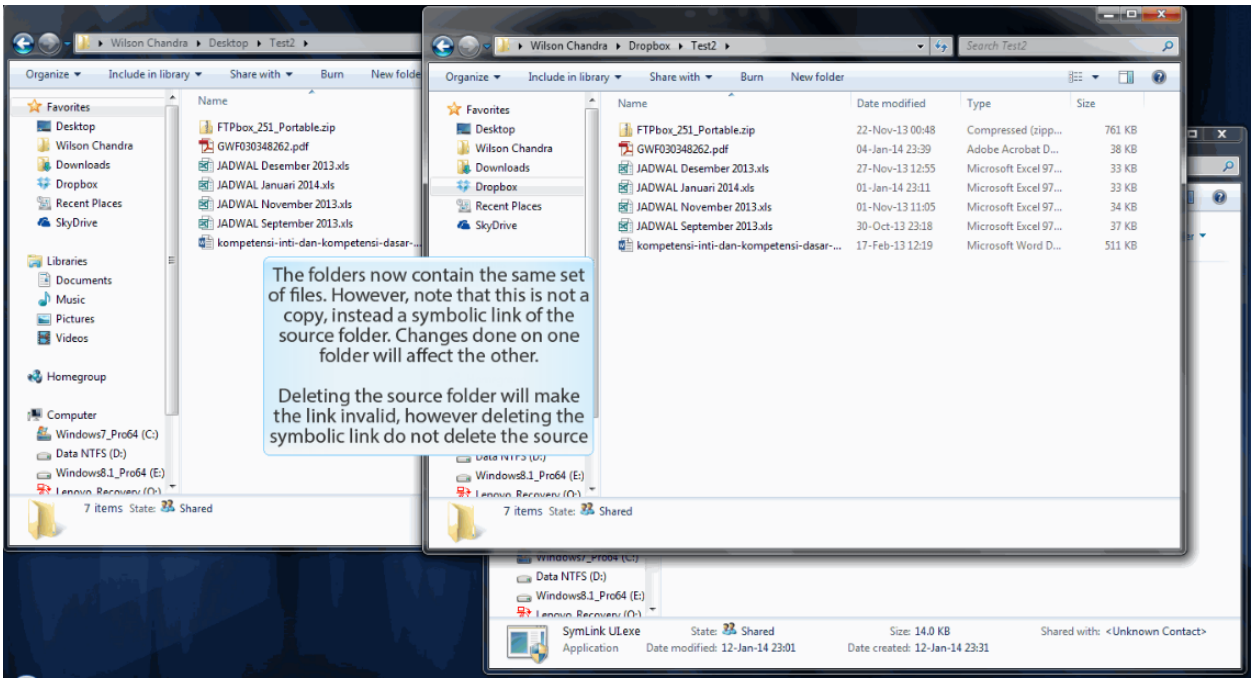
5. Click on submit button and the symbolic link is created for Test2



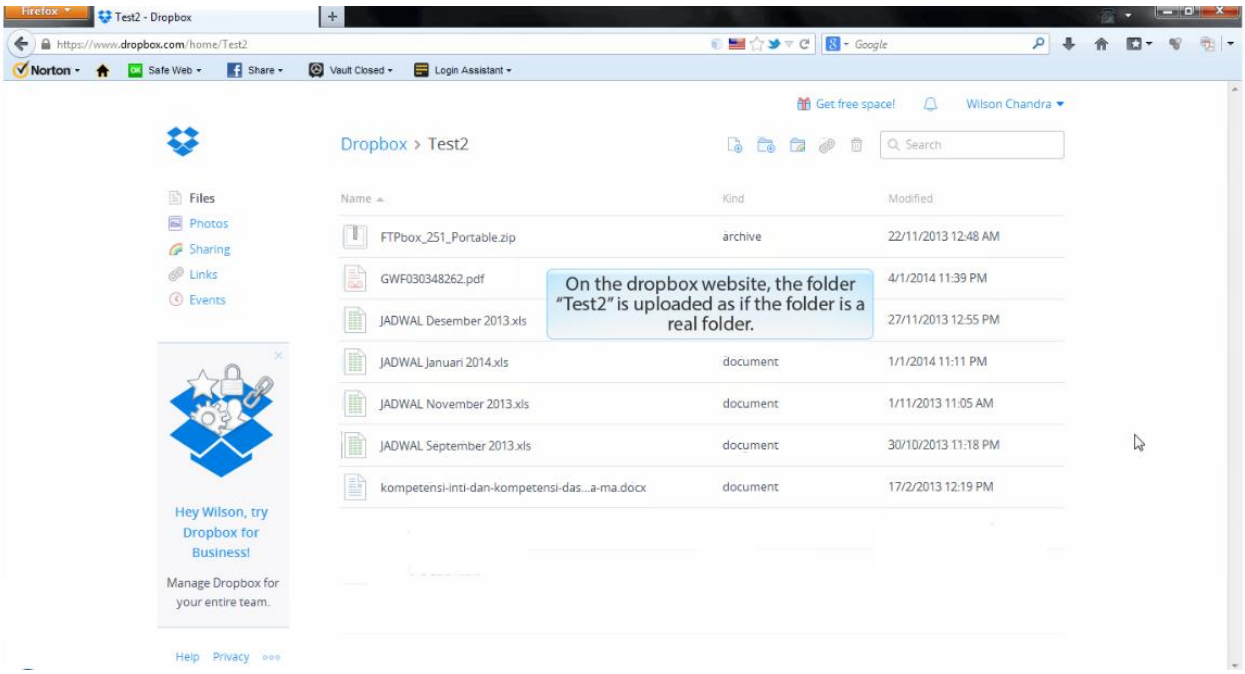
6. The folder Test2 is now shown on the cloud storage folder



7. Both the source folder and the cloud storage folder now contain the same set of files. However, note that this is not a copy, instead a symbolic link of the source folder. Changes done on one folder will affect the other. Deleting the source folder will make the link invalid, however deleting the symbolic link don not delete the source.



8. On the Dropbox website, the folder ‘Test2’ is uploaded as if the folder is a real folder.



❹ **Phase 4.** Synchronize to Social Media.

- We are going to implement this phase on second semester

❺ **Phase 5.** Portable applications on the cloud storage.

- We are going to implement this phase on second semester

3.2. Demo Videos

Here are some links to the demo videos.

❶ **Phase 1.** Implement the self-hosted cloud storage functionalities.

http://i.cs.hku.hk/~wchandra/FYPDemo/Part%201%20-%20WebDAV%20and%20Sync.htm

❷ **Phase 2.** Drop anything other than files.

http://i.cs.hku.hk/~wchandra/FYPDemo/Part%202%20-%20Web%20Dropbox.swf

❸ **Phase 3.** Synchronize to other than cloud storage folders.

http://i.cs.hku.hk/~wchandra/FYPDemo/Part%203%20-%20External%20Folder%20Sync.swf