The University of Hong Kong
Faculty of Engineering
Department of Computer Science

COMP7704

# Secure, Distributive Storing and Mobile-Based Image Capturing App

Submitted in partial fulfillment of the requirements for the admission to the degree of
Master of Science in Computer Science

By
WEI Lingnan
3035150749

Supervised by
Dr. T. W. Chim and Dr. S. M. Yiu

July 30, 2015

## Declaration of the Candidate

I, the undersigned, hereby declare that the work contained in this thesis is my own original work, and has not previously in its entirely or in part been submitted at any university for a degree.

Only the source cited in the document has been used in this draft. Parts that are direct quotes or paraphrases are identified as such.

I agree that my work is published, in particular that the word is presented to third parties for inspection or copies of the work are made to pass on to third parties.

The University of Hong Kong,

Signature

# Abstract

Currently, many cloud accounts from pop stars have been proved to be cracked by brute-force attack and lots of sensitive photos are leaked. This shows that some famous cloud storages are not as secure as expected. On the contrary, storing photos only locally is not reliable as the mobile may be lost or crashed.

Based on the extreme security problem, it is necessary to do a deep and detailed analysis of the current problem. Then a wide range of background research is conducted to realize the current situation as much as possible. And finally, based on all the research done before, our own solution to the given problem is proposed.

In this project, a smart phone application is developed. After selecting a photo, the user can choose to store it onto multiple personal cloud platforms. Apart from that, an encryption algorithm is added to the multiple parts to provide further guarantee of the security. It is demonstrated that, the app can provide a further protecting to the original photo, and makes it possible for the users to upload to and download from cloud platforms in a more secure way.

**Keywords: Secure, Distributive, Encryption**

# Contents

# 1. Introduction

This year, many cloud accounts from pop stars have been proved to be cracked by brute-force attack and lots of sensitive photos are leaked. This shows that some famous cloud storages are not as secure as expected. On the contrary, storing photos only locally is not reliable as the mobile may be lost or crashed.

In this project, a smart phone application will be developed. After taking a photo by the mobile camera, or selecting a photo in the mobile-based album, a user can choose to store it onto multiple personal cloud platforms such that a number of shares are required to reconstruct the original photo.

We will include some well-known cloud platforms (say site 1, site 2, site 3, site 4 etc.) in the app. The app user can choose to store on site 1 only. In this case, the storage is just as normal as the current cloud platforms, but there is a risk that someone can view the photo by attacking the typical cloud platform. The app user can also choose to store on site 1, site 2, site 3… and so on. In this case, the photo will be broken down into multiple parts and each site only got one part of the photo. If one of the cloud platforms is attacked, the information leaked is just a small part of the original photo, thus is still more secure than the current practice of the cloud platforms.

In order to overcome the information leaking problem mentioned above, an encryption algorithm is to be added to the multiple parts of the original photo, which are going to be uploaded into different cloud platforms. Under these circumstances, even when any of the cloud platforms is attacked, the hackers cannot view the part of original photo stored on the cloud platform, since they do not know the decryption key at all.

# 2. Research and Solution

As mentioned in the introduction section, the current popularly used cloud platforms are not as secure as expected. Before proposing our own solution for the situation, some analysis, as well as background research is needed to be conducted.

This section tends to do a deep analysis of the current problem, investigate a wide range of background research about the situation, which includes the cloud platform research, as well as the related app research, then will come up with our own solution to the problem.

## 2.1 Analysis of Problem

Nowadays, smart phones are more and more widely used in all around the world. Since the local storage for smart phones themselves cannot reach as large as possible, people tend to upload their files, including their private photos to cloud platforms, thus they can have more space for the smart phones local storage.

In this case, the cloud platforms become extremely vital, as they obtain all the private information for the users. Once the cloud platform is crashed or attacked, all the information will be lost or leaked without any doubt. For more terrible situation, the hackers, who have access to the personal information, illegally use or expand the personal information, which will lead to more horrible results via the Internet everywhere in the world. Thus the security of cloud platforms needs pay great attention to as much as possible.

However, compared with the side of cloud platform security, is there anything the side of users can do to avoid or reduce the possibility of information loss and leaking?

Since we cannot totally rely on the server side of cloud platforms, there are also some good action for the client side users to take. It is believed that, the storage to cloud platform can be more secure under both of the two sides of protection.

According to the discussion above, it is possible to develop a client side mobile-based app to reduce the possibility of loss and leak of private information. After some operation to the original photo, the original information of the photo is hidden for others, thus the information uploaded to the cloud is not seeable for all the others. Under this situation, we can say, the mobile-based app can provide another guarantee for the security of the original photo.

## 2.2 Background Research

Before stepping further, a solid background research is needed to get to know the situation we are now in. In this section, we will conduct a wide range of background research to find the current problems as much as possible.

It is believable that a background research is a challenging part for a dissertation project. It requires careful research, diligent compilation of data from multiple sources as well as intelligently forming our own opinions about the subject.

To make things simpler, we will first do a detailed research about the popular cloud platforms which are nowadays get used worldwide, anglicize their positives and negatives, and then find the aspects we can do to improve them. Then we will test some relative apps which aim to keep the users' private photos more safe. We tend to find the common practice they take to satisfy the requirements of safety through this stage. And finally, based on all the research we have done, we will be able to propose our own solution to the given problem.

## 2.2.1 Cloud Platform Research

For the cloud store platforms, we have tested 6 popular apps, namely: Dropbox, OneDrive, Google Drive, 百度云(designed by Baidu), 360 云盘, and 微云(designed by Tencent). For more specifically, we just test the photo storing function of the listed cloud platforms.

### 2.2.1.1 Dropbox

Dropbox has the function of uploading from local photo album, as well as downloading from the cloud. The app can access to the local photo album, and then upload to the cloud, or download from the cloud. We can take an overall look about Dropbox in Figure 1 below.

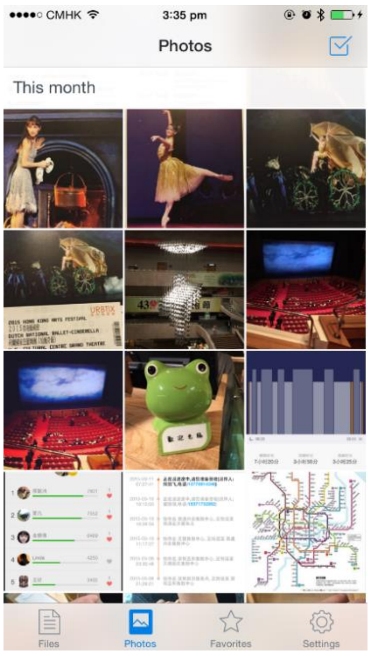


**Figure 1 Dropbox**

As for the security part, Dropbox provides additional layer of password to keep the app safer. For example, after the user enters the user name and password of Dropbox, it requires an extra layer of password to access the photos. For this layer of password, digital password and Touch ID is supported in iOS system, as Figure 2 shows.

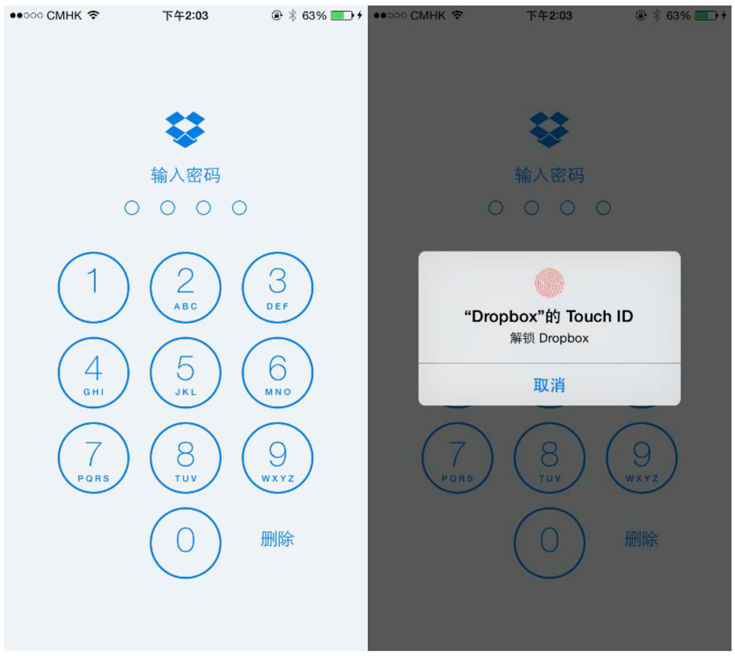


**Figure 2 Dropbox Password**

Recently, Dropbox has promoted Carousel to focus on the photo editing and sharing. Some new functions such as distinguishing photos by date or by location is quite new compared with other cloud platforms. We can see from the left part of Figure 3 that, Carousel can distinguish the photos by the right hand timeline, also it can automatically distinguish the photos by different locations. The photos shown in the left part of Figure 3 are all taken on 12 March, but they are distinguished into different parts, as they are taken in different locations.

Apart from that, Carousel makes sharing photos by album possible. We can see from right part of Figure 3 that, users can choose several photos together and share them through Carousel, message, email, or they can just copy the hyperlink of the photos for further sharing.

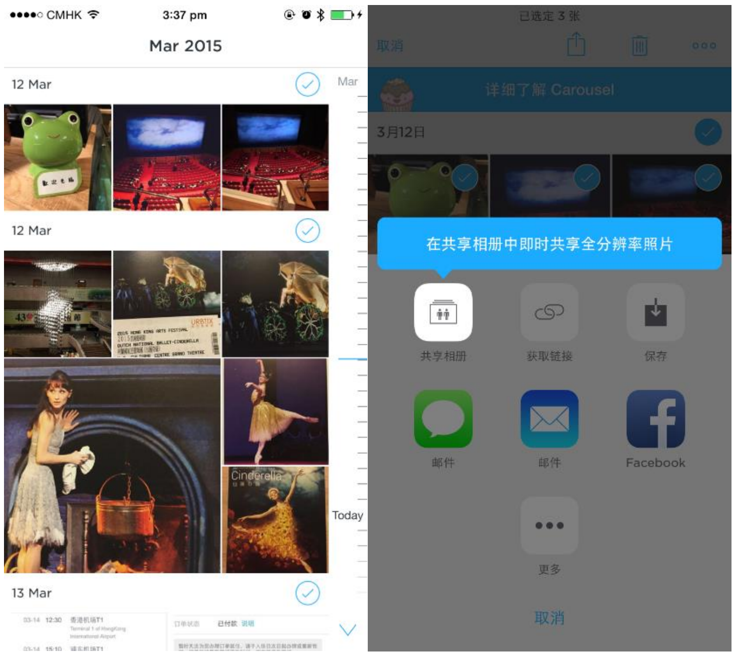


**Figure 3 Carousel Distinguishing and Sharing**

As for the security part, Carousel acts the same with Dropbox, as they are similar products in cloud storing in a same company. Figure 4 shows that the password of Carousel is also related to Dropbox.

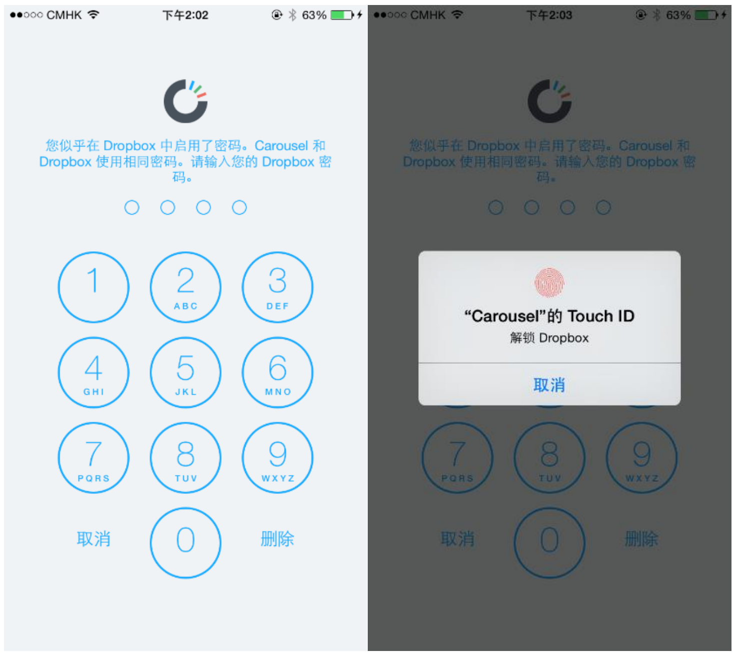

您似乎在 Dropbox 中启用了密码。Carousel 和 Dropbox 使用相同密码。请输入您的 Dropbox 密码。
"Carousel"的 Touch ID
解锁 Dropbox
取消
删除

**Figure 4 Carousel Password**

### 2.2.1.2 OneDrive

OneDrive is a more powerful platform generated by Windows system. One interesting function is automatically detection of photos, but it seems not as accurate as expected. Figure 5 below shows OneDrive considers the “green sea” as the grassland.

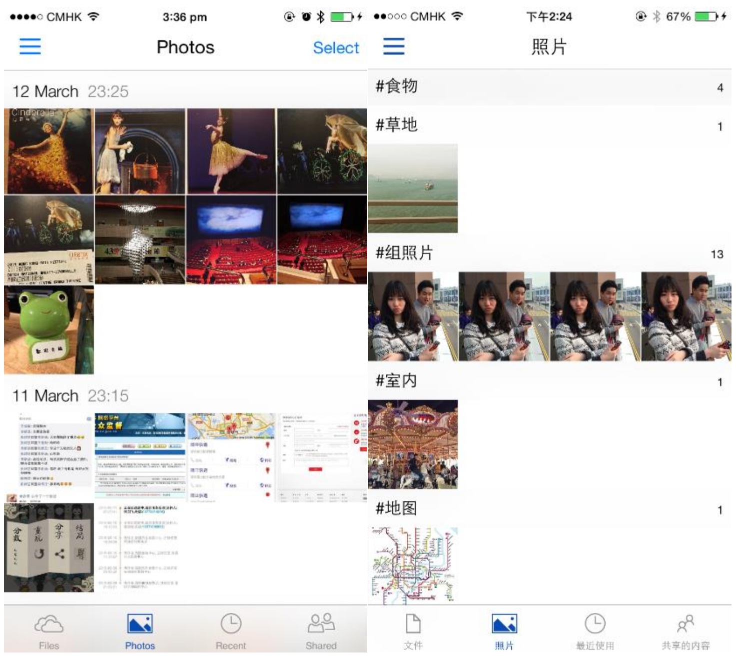


**Figure 5 OneDrive**

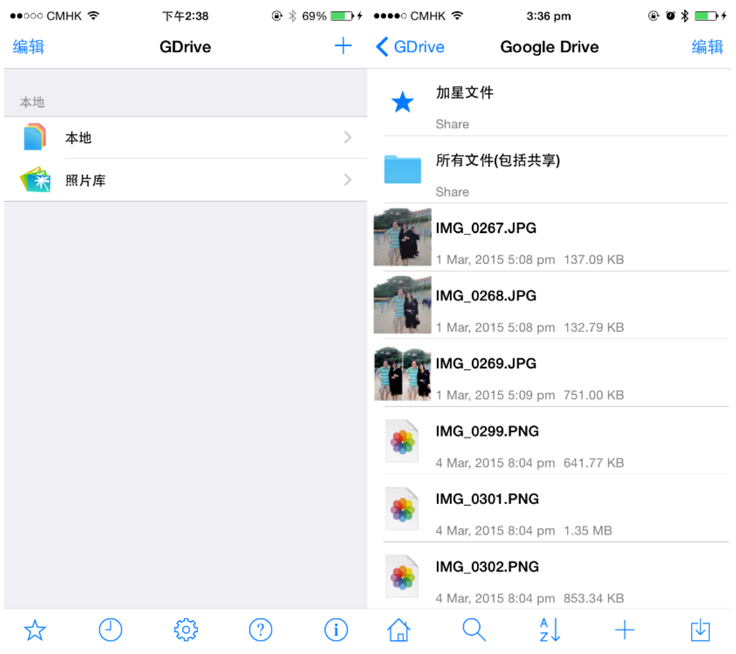
When it comes to the security part, it takes the same action with Dropbox. Digital password as well as Touch ID is supported in the app. Users can choose to take the additional layer of password or not via the setting section of the app by themselves.

To sum up, OneDrive is aimed to create a wide cloud platform which contains all categories of files, including documents, photos, and notes. However, it is also because of this aim that leads to the results that, OneDrive cannot pay much special attention on photo storing function. Though OneDrive is going to add some creative functions to the photo storing part, there is a long way to go before it becomes as accurate as the users will accept.

### 2.2.1.3 Google Drive

Google Drive seems rather different compared with Dropbox and OneDrive, because it classifies the files with local photos and cloud platform photos, and more focused in photo storing part, which can be obviously seen in the left part of Figure 6.

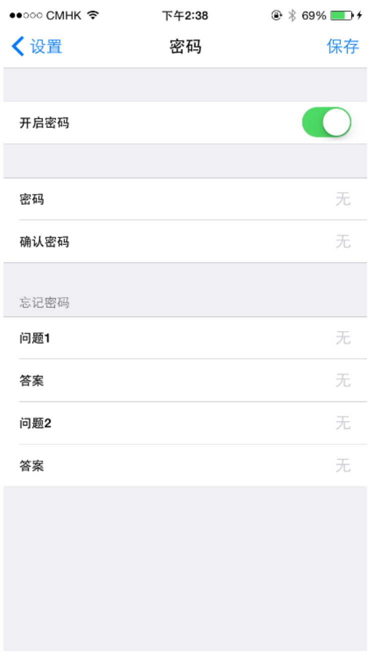
Another interesting function for Google Drive is that, it can automatically store the files attached in Gmail to Google Drive, as there is a strong backup with the Google series products. From the right part of Figure 6, we can easily see that, the photos or videos which are attached in Gmail before are automatically stored in Google Drive then.



加星文件
Share
所有文件(包括共享)
Share
IMG_0267.JPG
1 Mar, 2015 5:08 pm 137.09 KB
IMG_0268.JPG
1 Mar, 2015 5:08 pm 132.79 KB
IMG_0269.JPG
1 Mar, 2015 5:09 pm 751.00 KB
IMG_0299.PNG
4 Mar, 2015 8:04 pm 641.77 KB
IMG_0301.PNG
4 Mar, 2015 8:04 pm 1.35 MB
IMG_0302.PNG
4 Mar, 2015 8:04 pm 853.34 KB

**Figure 6 Google Drive**

As for the security part, Google Drive is also different with Dropbox or OneDrive. It cannot support the digital password or Touch ID, but just character password only, which is shown in Figure 7. Actually these three take same action for security, because they all provide an additional layer of password to protect the private information on the cloud. Of course this layer of password is optional by the setting part of the app.



开启密码

密码 无

确认密码 无

忘记密码

问题1 无

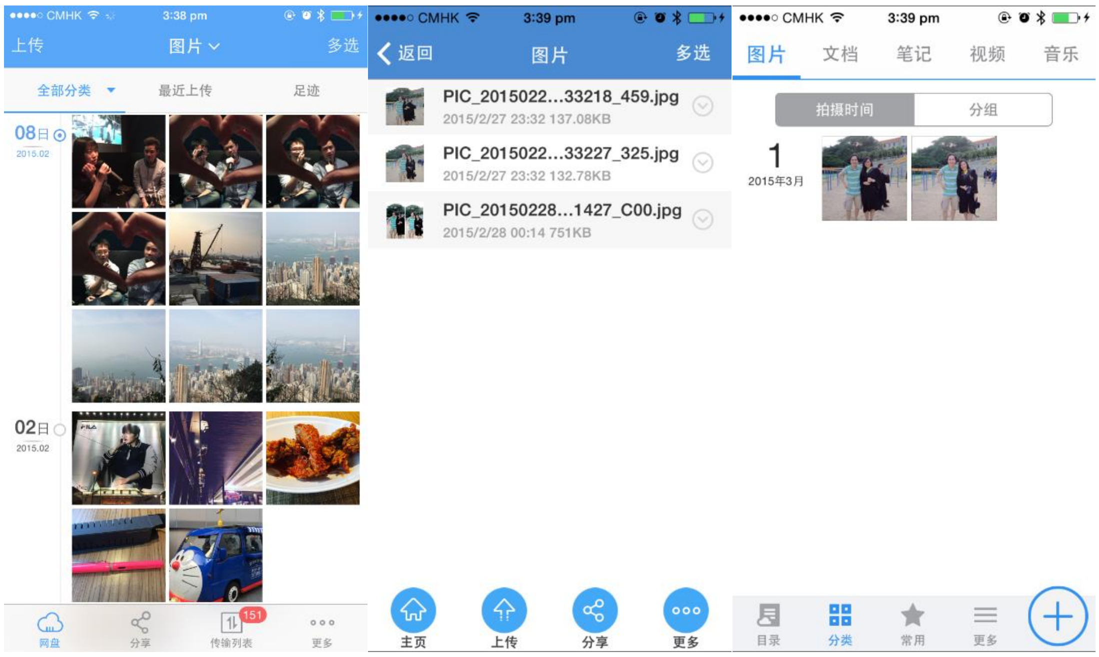
答案 无

问题2 无

答案 无

**Figure 7 Google Drive Password**

### 2.2.1.4 百度云 (designed by Baidu), 360 云盘, and 微云 (designed by Tencent)



PIC_2015022...33218_459.jpg
2015/2/27 23:32 137.08KB

PIC_2015022...33227_325.jpg
2015/2/27 23:32 132.78KB

PIC_20150228...1427_C00.jpg
2015/2/28 00:14 751KB

**Figure 8 Other Three Cloud Platform**

As Figure 8 shows, these three popular cloud platforms are similar with OneDrive by generating all the files (documents, photos, music and video) together in one platform, but nothing special for photo storing part. Thus we can just treat them as another OneDrive app.

To sum up, there are mainly two directions for nowadays cloud storing cloud platforms: one is as OneDrive, it generates all the categories of files into one platform; the other is as Carousel, it is more professional in photo storing and sharing functions. And for all those nowadays widely used cloud platform, one common practice for the security part is to add an extra layer of password to protect the private information to be viewed by others. Users can decide whether to add this layer of password by changing the settings themselves in the app.

However, for the cloud platform listed above, there is no specific function considering the leak of photo information. If the platform is hacked or attacked, all the photo information on the platform are so dangerous because they are seeable to the hackers, but there is not any further protection at all.

### 2.2.2 Related App Research

For further research, we have also tested some related apps with the function of protecting our photos in the smart phone. To be more detailed, we have tested 9 popular apps, namely: Photo Safe, Photorange, iVault, HiDisk, 1Passe, iPassword, FileMaster, Albums, and HiCalculator. After the detailed testing, we have found some common practice for the related apps.

One common practice is to set different layers of passwords as much as possible, such as password for logging in, password for entering into typical folders. Also there are different kinds of password they can use, such as digital password, character password, graph password and touch ID. Take the app of Photorange as example, we can easily see from Figure 9 that, Photorange provides the users with different kinds of password to choose, either Touch ID or graph password. And for some other apps, they can be supported by digital password or character password.

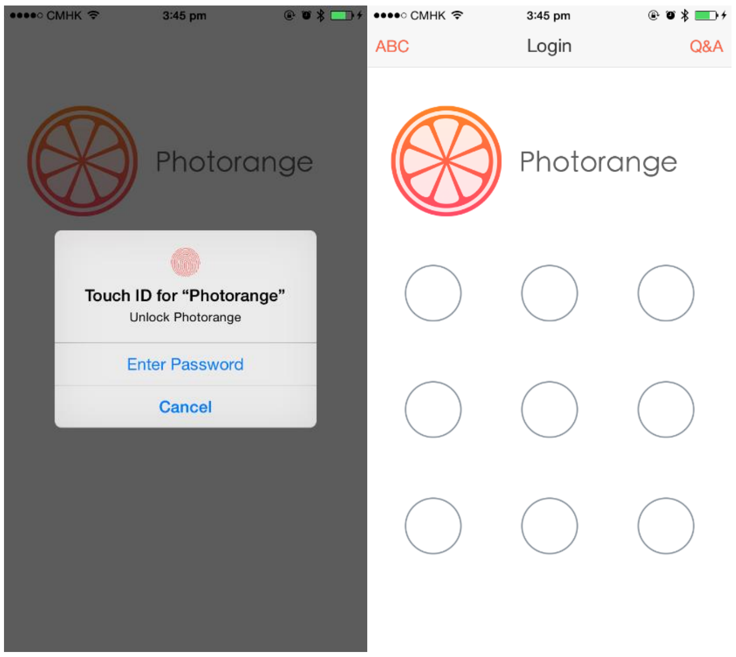


**Figure 9 Photorange Different types of passwords**

Another small trick is to use a different app icon but actually act as a photo album. One extreme example is the app of HiCalculator. Apart from the calculator icon, the operation function is also supported for HiCalculator.  We can go through the work flow of the app of HiCalculator as Figure 10 follows.

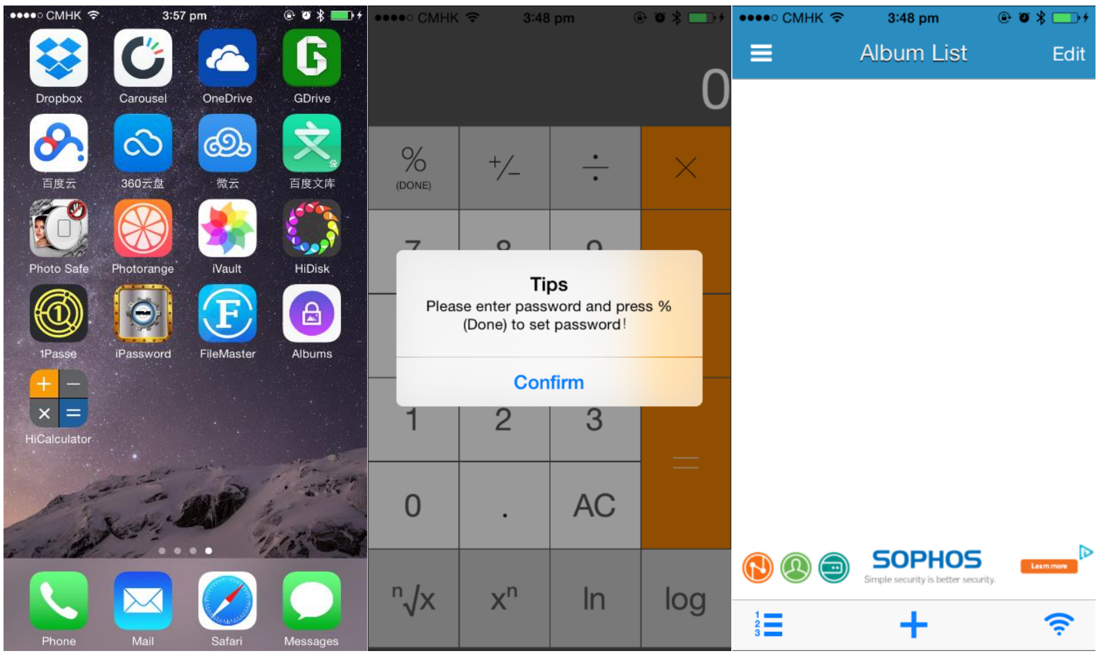


**Figure 10 HiCalculator**

In the desktop, there is an app called HiCalculator whose icon of calculator. If the user does not know the password to enter the deeper album, the app just act as a common calculator. After the user enters the right password, the calculation page will disappear and shows an album page. After the photos are selected, the original files in the local smart phone album will be deleted automatically. In that case, nobody will realize the private photos behind a "calculator" icon.

In conclusion, combining the current cloud platforms and photo protecting apps, the most widely used practice is still to add different layers of password to reach the aim of protection. This practice is quite easily understandable based on the fact that, with more layers of passwords, the possibility of each layer of passwords are hacked is lower.

However, it is also interesting to find that, neither the current cloud platforms nor photo protecting apps pay any attention to the original photos which is going to be uploaded to the cloud. It is believable that after some operation of the original photos, the original information of the photo is hidden for others, thus the information uploaded to the cloud is not seeable for all the others. Under this situation, we can say, the mobile-based app can provide another guarantee for the security of the original photo.

## 2.3 Solution

According to the previous work on reviewing and analysis of the current cloud platforms and photo protecting apps, we can say it is possible to develop a client side mobile-based app to reduce the possibility of loss and leak of private information. Thus here we propose our solution for the project as follows:

In the project, we will develop a smart phone app through which the users can choose different cloud platforms to store their photos, which will reduce the possibility of attacked or loss of their photos.

We will include some well-known cloud platforms (say site 1, site 2, site 3, site 4 etc.) in the app. The app user can choose to store on site 1 only. In this case, the storage is just as normal as the current cloud platforms, but there is a risk that someone can view the photo by attacking the typical cloud platform. The app user can also choose to store on site 1, site 2, site 3… and so on. In this case, the photo will be broken down into multiple parts and each site only got one part of the photo. If one of the cloud platforms is attacked, the information leaked is just a small part of the original photo, thus is still more secure than the current practice of the cloud platforms.

For more secure, an encryption algorithm is added before the uploading to different cloud platforms. Only the users or developers know the encryption and decryption key, so that even when the hackers get the particular part of photo, they cannot view the particular part at all. As for the reconstruction part, users can choose the uploading information, such as the time of uploading, and then the app will download the multiple parts, decrypt into original data, and then generate them together to the original photo.

In our project, we decide to use the Advanced Encryption Standard (AES) to encrypt and decrypt the photos. AES is a specification for the encryption of electronic data established by the US National Institute of Standard and Technology (NIST) in 2001. It performs well on a wide variety of hardware, from 8-bit smart cards to high-performance computers.

Based on the discussion above, it is foreseeable that our solution for the project will necessarily reduce the possibility of attacked or loss of users' photos, thus provide a great guarantee to their cloud platform storing photos.

# 3. Implementation of App

In this section, after the background research about both cloud platform and related apps, it is clear to have a rough idea to implement our own app. Before real implementation of the app, a relative high fidelity prototype is needed to test the basic functions of our ideal app, which needs the details of design as much as possible. Following that, the detailed functions of our app is going to be proposed, thus the usage of the app can be introduced naturally.

## 3.1 User Interface Design

According to the previous work, it is concluded that a good mobile interface for our app should possess the characters as follow: 1) function of uploading and downloading; 2) access to multiple cloud platforms; 3) ability to guarantee the security of the storage. Thus the user interface design proposed in this section will mainly focus on the above three sections.

In this part of user interface design of our app, a famous prototype tool Fluid UI is used. Fluid UI is professional in Android, iOS or Windows 8 mobile apps with custom libraries. What's more, it makes testing on real devices possible. After scanning the two-dimension code, it is practical for the navigation by our own device.

### *3.1.1 Welcome Page*

After entering the **WELCOME PAGE**, there are three main pages for the app all together: the **ACCOUNT PAGE**, the **UPLOAD PAGE**, and the **DOWNLOAD PAGE**. We can implement the navigation of these three pages via the tags of tab bar at the bottom of different pages. The layout of welcome page is quite simple, as it just provides the entrance of the app. It can be seen in Figure 11 below.

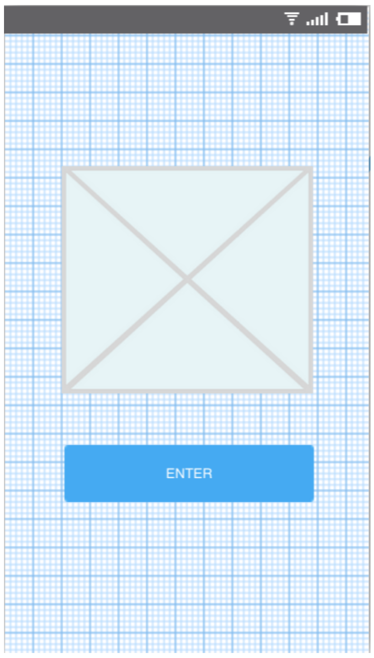


**Figure 11 User Interface Design of Welcome Page**

### *3.1.2Account Page*

In the **ACCOUNT PAGE**, the user can view all his cloud platforms, or is able to add new cloud platform accounts to the app by touching the "**ADD**" button at the top right of the page. This progress can be seen in Figure 12 below.

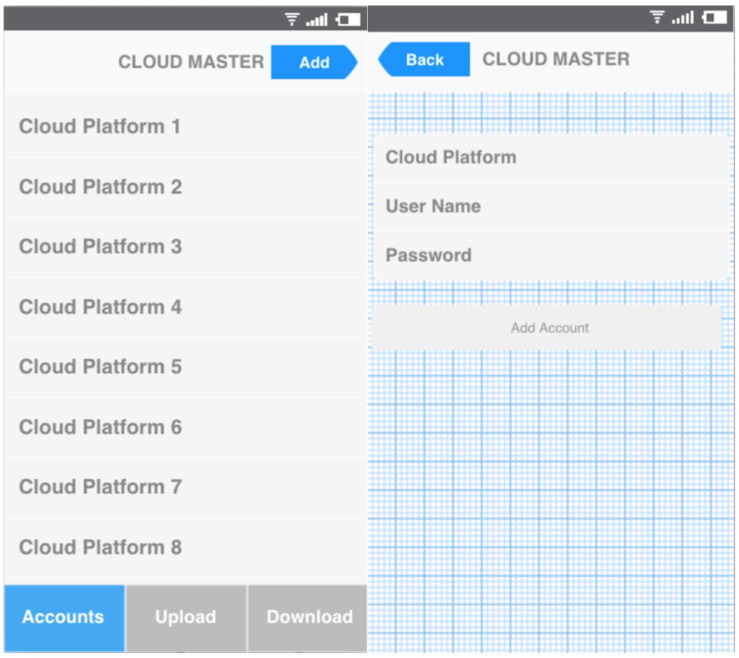


**Figure 12 User Interface Design of Account Page**

### *3.1.3 Upload Page*

In the **UPLOAD PAGE**, when the user needs to upload his photo, he can touch the "**SELECT**" button at the top left of the page, and choose to access local camera or photo album to get the photo. After selecting, the photo selected will be shown in the page, then the user needs to touch the "**NEXT**" button at the top right of page. Then the user can choose the cloud platforms he would like to upload the photo. When the user touch the "**UPLOAD**" button, the original photo will be divided automatically, encrypted with the AES special key, and then uploaded to the selected cloud platforms respectively. This progress can be seen in Figure 13 below.

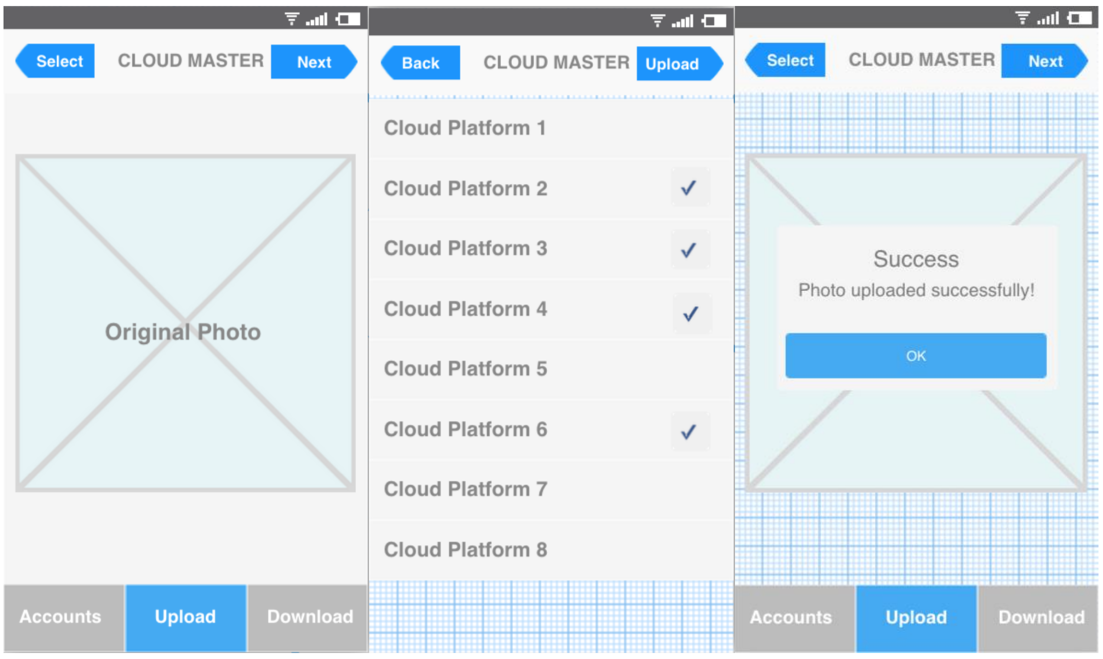


**Figure 13 User Interface Design of Upload Page**

### *3.1.4 Download Page*

In the **DOWNLOAD PAGE**, there are storing information listed in the page. The user can select any cell of the list, and then the app will download different fragments from the respective cloud platforms, decrypt the photo segments, and reconstruct the original photo. The reconstructed photo is seeable in another page, when the user can touch the "**DOWNLOAD**" button at the top right of the new page, it will be stored in the local photo album automatically for the user. This progress can be seen in Figure 14 below.

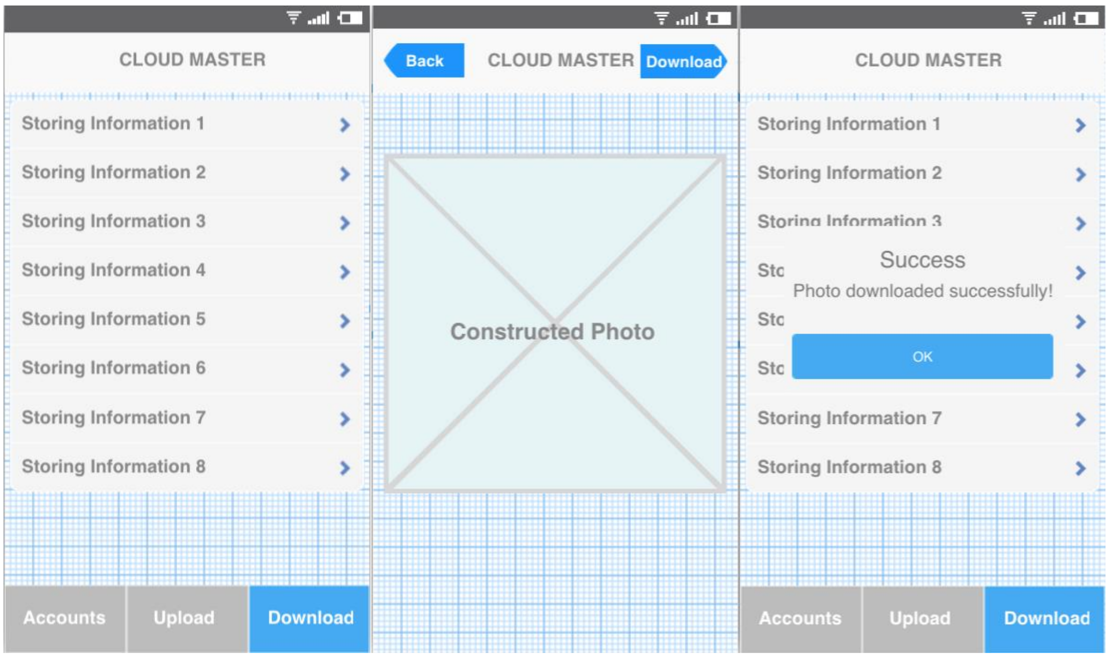


**Figure 14 User Interface Design of Download Page**

As a high fidelity prototype with the details of design as much as possible on it, Fluid can be tested on real devices via the two-dimension code below in Figure 15. Readers can just scan the two-dimension code and test the user interface on their own devices.

**Figure 15 Two-Dimension Code for User Interface Design**

## 3.2 Function of Accounts Access

Based on the user interface designed and tested in the former section, we did some modification of the user interface for app, and now it is much easier to focus on the code implementation part of our app. The app is implemented in Objective-C in Xcode, with three main functions: accounts access, upload and download. And all these three functions can be reached via the navigation bar at the bottom of the app.

As for the function of accounts access, when the user enters the app, he can view all his cloud platforms, or is able to add new cloud platform accounts to the app. After touching the “Add” button at the top right of the page, the user can input the account information and then choose “Add Account”. The page will navigate to the former page with the new account updated. We can take the look at the function of accounts access in Figure 16 as below.

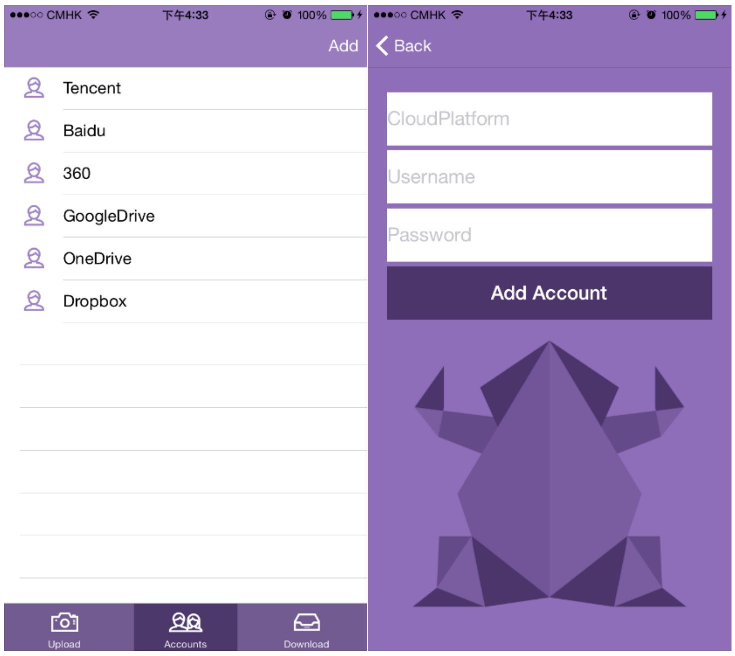
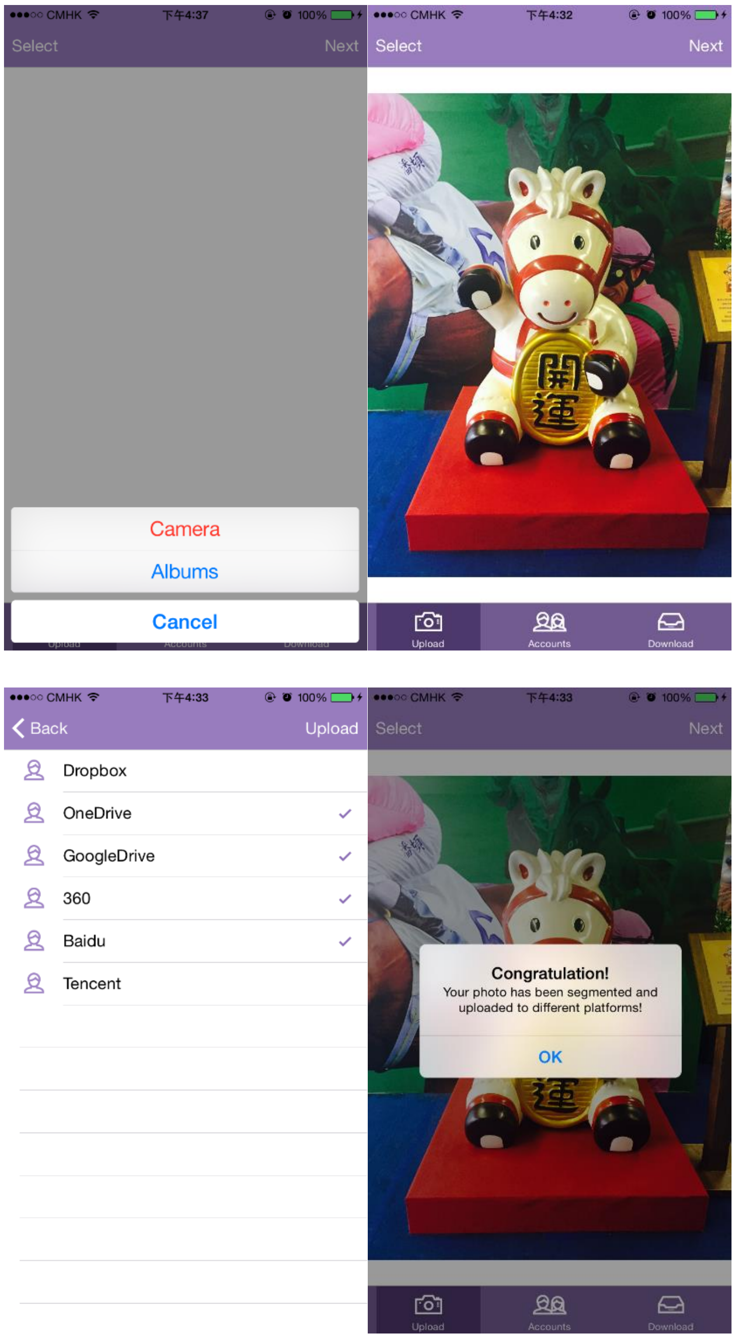


**Figure 16 Function of Accounts Access**

## 3.3 Function of Upload

As for the function of upload, there are 3 steps for the user to complete his uploading of his original photo. The operation flow of the function of upload can be easily seen by Figure 17 shown below.



Congratulation!
Your photo has been segmented and uploaded to different platforms!

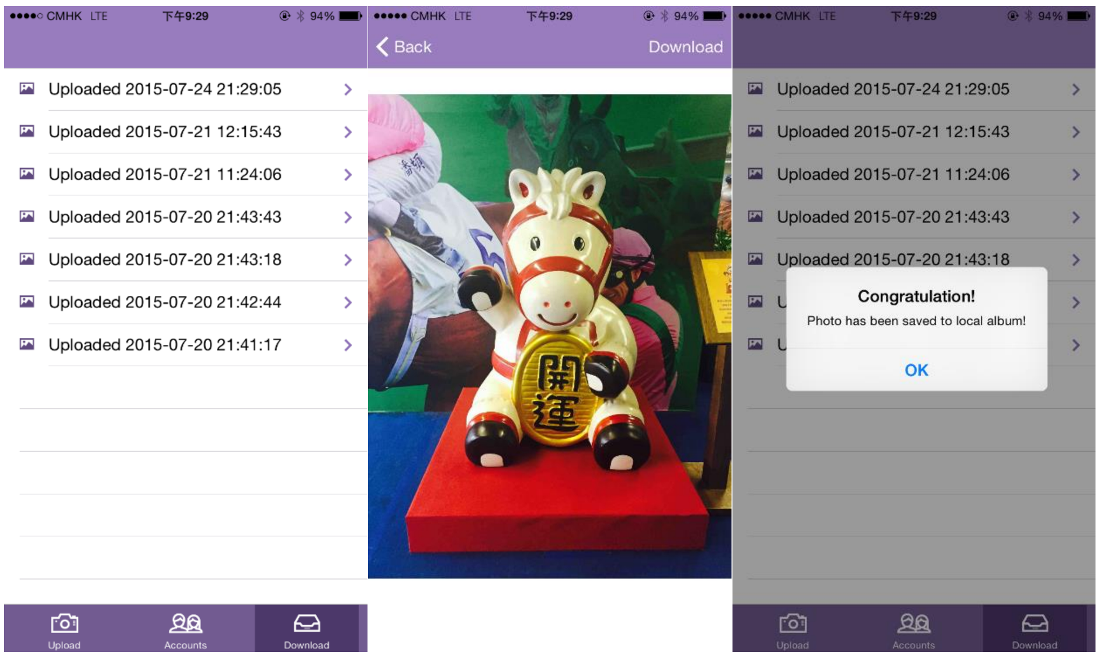
**Figure 17 Function of Upload**

Step 1: When the user needs to upload his photo, he can touch the “Select” button, and choose to access local camera or photo albums to get the photo.

Step 2: After selecting the photo, the picture will be shown on the screen, then the user need to touch the “Next” button and choose the cloud platforms he would like to upload the photo. Once tapping any cloud platform, there will be a kick off right to the cloud platform; when tapping again, the kick off will disappear, which means the cloud platform is not selected at all.

Step 3: When the user touch the “Upload” button, the app will divide the original photo into several parts, encrypt them and upload to the selected cloud platform respectively. After successfully upload the photo, there will be an AlertView showing the success of uploading.

## 3.4 Function of Download


Uploaded 2015-07-24 21:29:05
Uploaded 2015-07-21 12:15:43
Uploaded 2015-07-21 11:24:06
Uploaded 2015-07-20 21:43:43
Uploaded 2015-07-20 21:43:18
Uploaded 2015-07-20 21:42:44
Uploaded 2015-07-20 21:41:17

Congratulation!
Photo has been saved to local album!
OK

**Figure 18 Function of Download**

As for the function of download, there are 2 steps for the user to complete downloading of his original photo. The operation flow of the function of upload can be easily seen by Figure 18 shown below.

Step 1: In the download page, the user is able to access the reconstructed photo via the image storing information, which actually is the image storing time in our app. After tapping one cell of the storing information, the app will download different fragments, decrypt them and reconstruct to the original photo to the screen.

Step 2: When the user touches the “Download” button, the reconstructed photo will be automatically stored in the local photo album of the smart phone. Same with the function of upload, after successfully download the photo to the local photo album, there will be an AlertView showing the success of downloading.

At this stage, our app of CloudMaster has been implemented successfully. Based on the user interface designed before the code implementation, it is quite easy for us to develop the app with the knowledge of how the details of each page, as well as the navigation of different pages. As for our app of CloudMaster, the three main functions of accounts access, upload and download we desired perform well in real iOS device.

# 4. Analysis of App

According to the research and implementation clarified above, this section provides a detailed core analysis of our application. Thus it is able for us to explain how our app works for the aim of protecting personal photos to the cloud platforms.

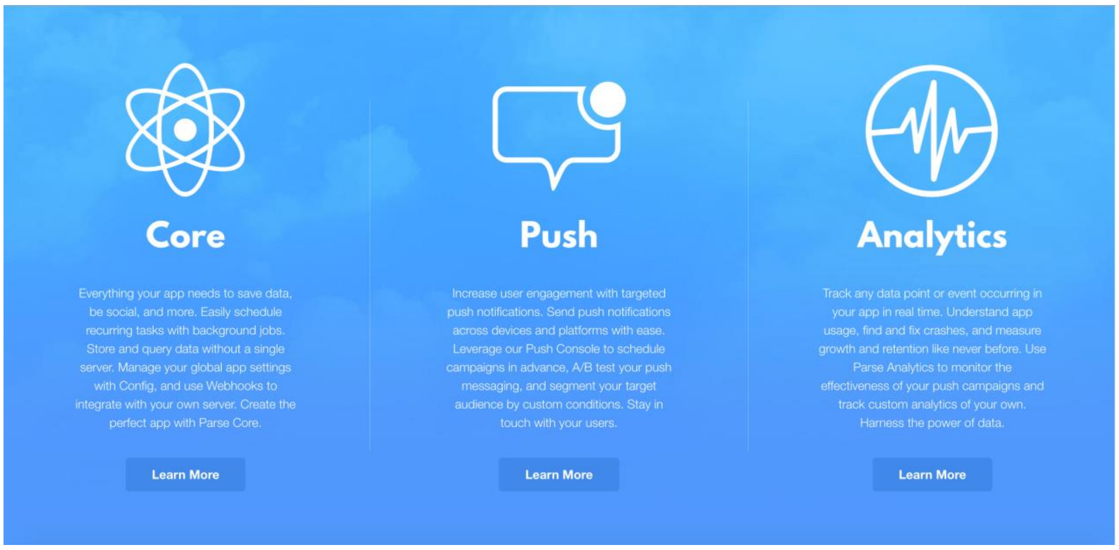
As for the detailed analysis part of our app CloudMaster, a 3rd backend Parse is used to detect the core data of our app. The cloud platform, the storing information, as well as the segments of the photos are accessible via the analysis part of Parse.

## 4.1 App Backend: Parse

In our app of CloudMaster, a 3rd backend Parse is used to detect the core data. Parse is useful for app developing, because it is able to save everything our app needs to save, and is easily schedule recurring tasks with background jobs. What's more, Parse makes storing and querying data without any single server at all.

As for the data analytics part, Parse tacks any data point or event occurring in the app in real time. It seems so smart that Parse understands app usage, is able to find and fix crashes, can measures growth and retention like never before. For mobile-based app developers, it is quite convenient to use Parse Analytics to monitor the effectiveness of the push campaigns and track custom analytics for their own apps.

To be more specific, the native SDKs of Parse make it easy to create apps for all the favorite devices we can imagine to use. As for our mobile-based iOS environment, it is possible for the developers to build powerful, full-featured mobile apps faster than ever before. The Parse Cloud handles the entire backend, and there is no need for use to worry about databases, performance, or scaling. All the developers need to do is just adding a few lines of codes in the primary projects.


Core

Everything your app needs to save data, be social, and more. Easily schedule recurring tasks with background jobs. Store and query data without a single server. Manage your global app settings with Config, and use Webhooks to integrate with your own server. Create the perfect app with Parse Core.

Learn More

Push

Increase user engagement with targeted push notifications. Send push notifications across devices and platforms with ease. Leverage our Push Console to schedule campaigns in advance, A/B test your push messaging, and segment your target audience by custom conditions. Stay in touch with your users.

Learn More

Analytics

Track any data point or event occurring in your app in real time. Understand app usage, find and fix crashes, and measure growth and retention like never before. Use Parse Analytics to monitor the effectiveness of your push campaigns and track custom analytics of your own. Harness the power of data.

Learn More

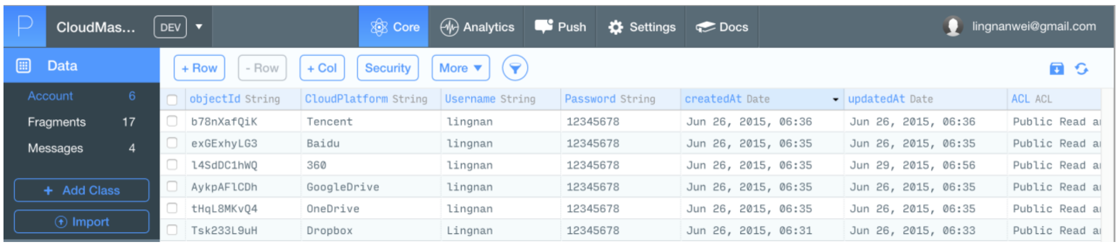
**Figure 19 Parse**

## 4.2 Analysis of App Backend

In this section, the backend of our app CloudMaster will be explained. For better understanding, the explanation will be shown by three main parts, namely: cloud platform information, storing information, and fragments of photos.

### *4.2.1 Cloud Platform Information: Accounts*

In Parse, we created a special class called "**Accounts**" to store the user's cloud platform information. In the class of Accounts, there are several attributes inside, namely: object Id of Accounts, Cloud Platform, Username, Password, created time and update time. The backend data of Accounts can be viewed in Figure 20 below.



| objectId String | CloudPlatform String | Username String | Password String | createdAt Date | updatedAt Date | ACL ACL |
|---|---|---|---|---|---|---|
| b78nXafQiK | Tencent | lingnan | 12345678 | Jun 26, 2015, 06:36 | Jun 26, 2015, 06:36 | Public Read a |
| exGExhyLG3 | Baidu | lingnan | 12345678 | Jun 26, 2015, 06:35 | Jun 26, 2015, 06:35 | Public Read a |
| l4SdDC1hWQ | 360 | lingnan | 12345678 | Jun 26, 2015, 06:35 | Jun 29, 2015, 06:56 | Public Read a |
| AykpAFlCDh | GoogleDrive | lingnan | 12345678 | Jun 26, 2015, 06:35 | Jun 26, 2015, 06:35 | Public Read a |
| tHqL8MKvQ4 | OneDrive | lingnan | 12345678 | Jun 26, 2015, 06:35 | Jun 26, 2015, 06:35 | Public Read a |
| Tsk233L9uH | Dropbox | Lingnan | 12345678 | Jun 26, 2015, 06:31 | Jun 26, 2015, 06:33 | Public Read a |

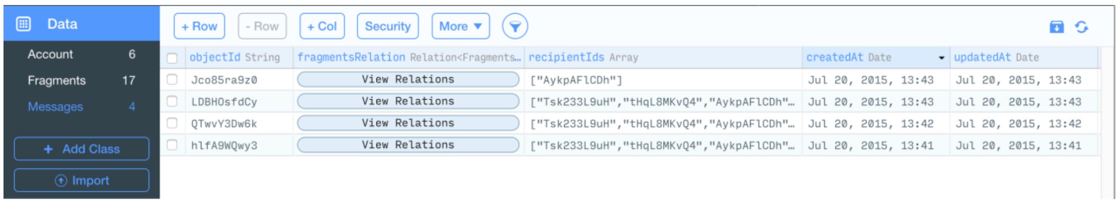
**Figure 20 Cloud Platform Information: Accounts**

The attribute of object Id is a unique Id for each cloud platform, when doing query or some other operations, Parse will search through the object Id of each class.

The attributes of Cloud Platform, Username and password matches with the adding account page in the former section we discussed before. After the user inputting the cloud platform name, the username, the password of his account, and touch the button of "**Add Account**", a new account will be created and added to the class of Accounts in the backend automatically.

### *4.2.2 Storing Information: Messages*

As for the storing information, a class called "**Messages**" is created for detection. The class of Messages contains these attributes: object Id of Messages, fragments relations, recipients Ids, created time and update time. The backend data of "Messages" can be viewed in Figure 21 below.



| objectId String | fragmentsRelation Relation<Fragments… | recipientIds Array | createdAt Date | updatedAt Date |
|---|---|---|---|---|
| Jco85ra9z0 | View Relations | ["AykpAFlCDh"] | Jul 20, 2015, 13:43 | Jul 20, 2015, 13:43 |
| LDBHOsfdCy | View Relations | ["Tsk233L9uH","tHqL8MKvQ4","AykpAFlCDh"… | Jul 20, 2015, 13:43 | Jul 20, 2015, 13:43 |
| QTwvY3Dw6k | View Relations | ["Tsk233L9uH","tHqL8MKvQ4","AykpAFlCDh"… | Jul 20, 2015, 13:42 | Jul 20, 2015, 13:42 |
| hlfA9WQwy3 | View Relations | ["Tsk233L9uH","tHqL8MKvQ4","AykpAFlCDh"… | Jul 20, 2015, 13:41 | Jul 20, 2015, 13:41 |

**Figure 21 Storing Information: Messages**

The attribute of object Id for Messages acts the same with the object Id for Accounts. It is more like an index for any typical class in Parse.

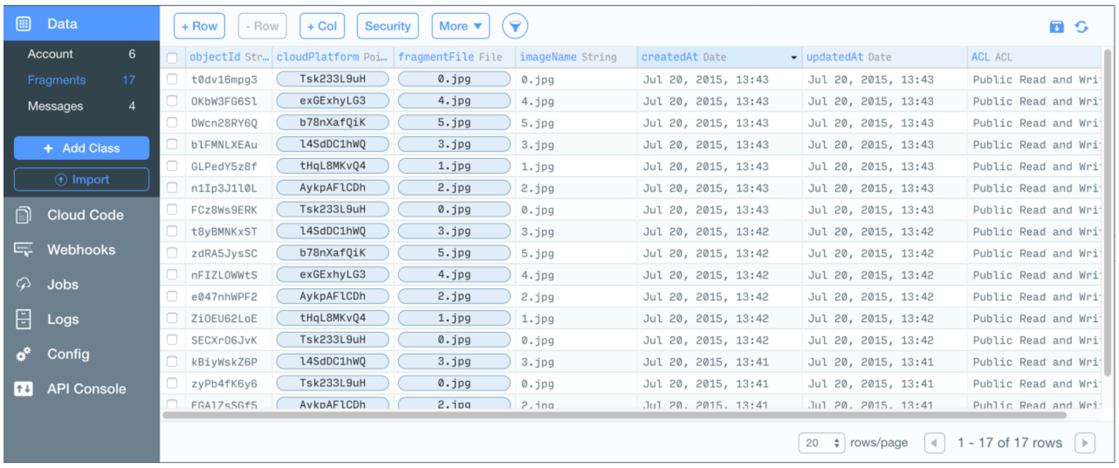
The attribute of fragments relations shows the relationship with the fragments. As the class of Messages records the storing information, it should have access to each fragments of the original photo. When we click “View Relations” in the attribute, there will appear all the segments related to the photo stored. With the help of this attribute, it is possible for us to store all the fragments information directly.

The attribute of recipients Ids is an array, which stores all the object Ids of Cloud Platforms the users choose to store. The sequence of the object Ids of Cloud Platforms are decided by the users themselves, as if they first select the cloud platform in the app, then the object Id of Cloud Platform will be added into the array first; while if they unselect the cloud platform in the app, the object Id of Cloud Platform will be removed automatically.

It is obvious to see that, in the class of Messages, it contains all the storing information, including the link to related fragments, as well as the object Ids of selected cloud platforms, while not any detailed information about the fragments. It is much safe compared with the common practice of collecting all the fragments information together in one place.

### *4.2.3 Fragments of Photos: Fragments*

As for the fragments information, a class called “**Fragments**” is created for better understanding. The class of Fragments contains these attributes: object Id of Fragments, cloud platform, fragment file, image name, created time and update time. The backend data of “Messages” can be viewed in Figure 22 below.



| objectId Str… | cloudPlatform Poi… | fragmentFile File | imageName String | createdAt Date | updatedAt Date | ACL ACL |
|---|---|---|---|---|---|---|
| t0dv16mpg3 | Tsk233L9uH | 0.jpg | 0.jpg | Jul 20, 2015, 13:43 | Jul 20, 2015, 13:43 | Public Read and Wri |
| OKbW3FG6Sl | exGExhyLG3 | 4.jpg | 4.jpg | Jul 20, 2015, 13:43 | Jul 20, 2015, 13:43 | Public Read and Wri |
| DWcn28RY6Q | b78nXafQiK | 5.jpg | 5.jpg | Jul 20, 2015, 13:43 | Jul 20, 2015, 13:43 | Public Read and Wri |
| blFMNLXEAu | l4SdDC1hWQ | 3.jpg | 3.jpg | Jul 20, 2015, 13:43 | Jul 20, 2015, 13:43 | Public Read and Wri |
| GLPedY5z8f | tHqL8MKvQ4 | 1.jpg | 1.jpg | Jul 20, 2015, 13:43 | Jul 20, 2015, 13:43 | Public Read and Wri |
| n1Ip3J1l0L | AykpAFlCDh | 2.jpg | 2.jpg | Jul 20, 2015, 13:43 | Jul 20, 2015, 13:43 | Public Read and Wri |
| FCz8Ws9ERK | Tsk233L9uH | 0.jpg | 0.jpg | Jul 20, 2015, 13:43 | Jul 20, 2015, 13:43 | Public Read and Wri |
| t8yBMNKxST | l4SdDC1hWQ | 3.jpg | 3.jpg | Jul 20, 2015, 13:42 | Jul 20, 2015, 13:42 | Public Read and Wri |
| zdRA5JysSC | b78nXafQiK | 5.jpg | 5.jpg | Jul 20, 2015, 13:42 | Jul 20, 2015, 13:42 | Public Read and Wri |
| nFIZLOWWtS | exGExhyLG3 | 4.jpg | 4.jpg | Jul 20, 2015, 13:42 | Jul 20, 2015, 13:42 | Public Read and Wri |
| e047nhWPF2 | AykpAFlCDh | 2.jpg | 2.jpg | Jul 20, 2015, 13:42 | Jul 20, 2015, 13:42 | Public Read and Wri |
| ZiOEU62LoE | tHqL8MKvQ4 | 1.jpg | 1.jpg | Jul 20, 2015, 13:42 | Jul 20, 2015, 13:42 | Public Read and Wri |
| SECXrO6JvK | Tsk233L9uH | 0.jpg | 0.jpg | Jul 20, 2015, 13:42 | Jul 20, 2015, 13:42 | Public Read and Wri |
| kBiyWskZ6P | l4SdDC1hWQ | 3.jpg | 3.jpg | Jul 20, 2015, 13:41 | Jul 20, 2015, 13:41 | Public Read and Wri |
| zyPb4fK6y6 | Tsk233L9uH | 0.jpg | 0.jpg | Jul 20, 2015, 13:41 | Jul 20, 2015, 13:41 | Public Read and Wri |
| FGA1ZsSGf5 | AykpAFlCDh | 2.jpg | 2.jpg | Jul 20, 2015, 13:41 | Jul 20, 2015, 13:41 | Public Read and Wri |

**Figure 22 Fragments Information: Fragments**

The attribute of cloud platform is actually the object Id of Cloud Platform, which matches with the object Id we mentioned in the class of Accounts and Messages.

The attributes of fragment file records the file of relative part of photo, while the attribute of image name shows the name of relative part of photo. With this two attribute, we can reconstruct the typical part of photo in the typical cloud platform easily. When we click a given fragment file, Parse will download the image file from its sever automatically.

After the explanation of the backend of Parse, we can easily find that, our app of CloudMaster can be monitored at the same time. And also owing to Parse, it is more convenient to know the deep principals how our app works in a right way.

# 5. Future work

According to the implementation, as well as the analysis we have done in the past two sections, it is demonstrated that our app CloudMaster can provide a guarantee to the original photos before uploading and downloading.

However, as a widely used mobile-based application which we aim to, there are still some aspects we can do to improve our product. In this section, we will explain four aspects which we need pay attention to in future work.

## 5.1 Dividing and Arranging Method

As for the dividing method, it is important in the breaking down of original photos. Based on the common life experience, we can easily imagine that, for a typical photo, if we divide it into more fragments, the information for each fragment will be less. That is to say, the more fragments, the more secure for each fragment, and then the more secure for the original photo.

When it comes to the arranging method for the fragments to the cloud platforms, it is also vital in information protecting for original photos. Take a common example that, there is an original photo which is divided into quite a number of segments. However, for a specific cloud platform, it just stores the segments which are quite close in the original photo. In this case, even though each fragment is quite small and contains just little information, there is still a lot of information of original photo in the whole specific cloud platform.

In our app CloudMaster, we took a simple dividing and arranging method to break down and store the original photo. To be more detail, if we choose four different platforms to upload the original photo, CloudMaster will simply divide the original photo into four fragments vertically, and then upload to the cloud platforms in the order when the cloud platforms are selected. This kind of simple operation for the original photo may not so secure in practice, as each fragment may contain quite a number of information, nor to say if two of the cloud platforms are attacked or hacked, half of the original photo will be reconstructed by the attackers or hackers, which is quite dangerous.

Thus, future work needs to pay attention to the dividing and arranging method of breaking down and uploading the original photo. Some good photo dividing algorithms, as well as the arranging strategies can be further researched and taken into consideration into the implementation part of our app.

## 5.2 Encryption before Dividing

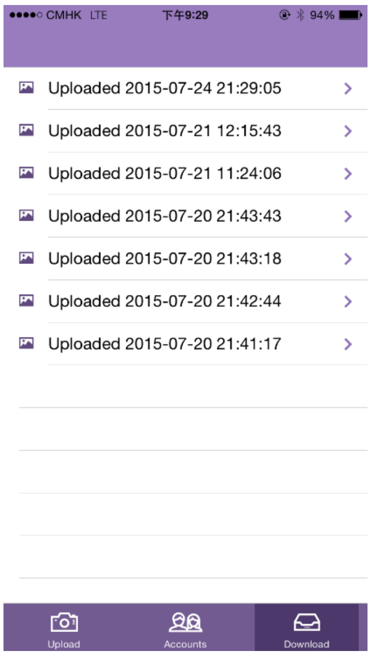
After a look back on the encryption method we took in the app, we find there is something interesting we can do to improve the security level of CloudMaster.

In the encryption of CloudMaster, we took the action of dividing photos first, then encrypting the segments, and finally uploading to the cloud platforms. Then what if there is attack in the stage of diving photos? If the encryption operation is done unsuccessfully, anyone can reconstruct the photo without much effort as the photo information is seeable to everyone.

Under these circumstances, we find we can change the order of three operations to improve the security level of CloudMaster. We can do the encryption first, then dividing the encrypted photo, and finally uploading to the cloud platforms in the future. Then the reconstructed flow will be downloading from cloud platforms first, then putting together, and finally decrypting to the original photo respectively.

## 5.3 Preview before Downloading

In our app CloudMaster, as for the function of download, it is only possible for users to find the photo by storing information, which is the image storing time in the page. This can be easily seen in Figure 23 below.


●●●●○ CMHK LTE 下午9:29 94%

Uploaded 2015-07-24 21:29:05

Uploaded 2015-07-21 12:15:43

Uploaded 2015-07-21 11:24:06

Uploaded 2015-07-20 21:43:43

Uploaded 2015-07-20 21:43:18

Uploaded 2015-07-20 21:42:44

Uploaded 2015-07-20 21:41:17

**Figure 23 Download Page**

This kind of interface may lead to some confusion to the user, when he has uploaded quite a number of photos to the cloud platforms, and cannot remember the image storing time for each photo so accurately. Although it is practicable that after reconstructing the selected photo, the user will recognize the photo until the right photo he wants to download, it is still time consuming and not convenient in real life.

Thus something needs to be considered to overcome the problem. One possible solution is that, we can apply a blur algorithm, which is able to protect the original information of the photo in some degree, and then showing the blurring photo in front of the image storing time. In this case, the user will not worry about the leaking of his original photo information; at the same time, it is able for the user to reach the photo he really wants to download from the cloud platforms.

## 5.4 Real Sever by API

One point which is quite obvious to see for the app CloudMaster is that, we do not connect any real server for current cloud platforms. In the project, CloudMaster is able to be detected and tested in the backend Parse, but it is not supported to the real severs of cloud platforms.

In this situation, future work needs to pay more attention to the real server API connection. It is foreseeable that, after the connecting to real severs, CloudMaster can be more powerful in photo storing, and will be a great extra guarantee to the current cloud platforms.

To sum up, there is still a long way to go for the improvement of our app CloudMaster. As for the function improvement, we need to apply new dividing and arranging method, conduct the encryption progress before dividing photos, to further increase the security level of CloudMaster. As for the usage improvement, we need to take a blurring algorithm for image previewing, and what's more, it is also necessary for us to connect the real servers of cloud platforms via their open API.

# 6. Conclusion

Reviewing the whole process of the project, it is not difficult to find, the developing of a specific application is not only based on computer science, but also related to other disciplines, such as a solid research on the former stage, a primary solution for the current problem, a detailed user interface design for testing before developing, a deep analysis and testing for the developed application after implementation, and so on.

In this project, we developed a smart phone application called CloudMaster, which has three main functions as below:

1. In the function of account access, it is possible for the user to view all his cloud platforms, as well as to add new cloud platform accounts to the app.
2. In the function of upload, after the user selecting the original photo, the app will vertically divide the photo into several fragments, then encrypt them with Advanced Encryption Standard (AES), and finally upload to different cloud platforms they selected.
3. In the function of download, after the user choosing specific storing information, the app will download the segments, decrypted them by related AES keys into original parts, and finally put them together to the original one. The user also has the option to save the downloaded photo into local albums.

As for the result of our app CloudMaster, it is demonstrated that, the app can provide a further protecting to the original photo, and makes it possible for the users to upload to and download from cloud platforms in a more secure way. As a client mobile-based application, CloudMaster conducts several steps of former operation to the original photos before uploading to cloud platforms, which increases the security performance for the photos in cloud platforms.

At end, there also proposes some practicable aspects which could be paid further attention to improve the performance of our app CloudMaster in the future. We can improve the app CloudMaster in both function performance and usage convenience for the users. It is foreseeable that our app CloudMaster will have an excellent performance in storing security, as well as user friendly in the future.

# 7. Reference

[1] Sweeney, Latanya. "k-anonymity: A model for protecting privacy." International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10.05 (2002): 557-570.

[2] Gonzalez, Rafael C. Digital image processing. Pearson Education India, 2009.

[3] Calder, Brad, et al. "Windows Azure Storage: a highly available cloud storage service with strong consistency." Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles. ACM, 2011.

[4] Pal, Nikhil R., and Sankar K. Pal. "A review on image segmentation techniques." Pattern recognition 26.9 (1993): 1277-1294.

[5] Cheng, Heng-Da, et al. "Color image segmentation: advances and prospects."Pattern recognition 34.12 (2001): 2259-2281.

[6] Hwang, Kai, Jack Dongarra, and Geoffrey C. Fox. Distributed and cloud computing: from parallel processing to the internet of things. Morgan Kaufmann, 2013.

[7] Subashini, Subashini, and V. Kavitha. "A survey on security issues in service delivery models of cloud computing." Journal of network and computer applications 34.1 (2011): 1-11.

[8] Yan, Liang, Chunming Rong, and Gansen Zhao. "Strengthen cloud computing security with federal identity management using hierarchical identity-based cryptography." Cloud Computing. Springer Berlin Heidelberg, 2009. 167-177.

[9] Tidwell J. Designing interfaces[M]. " O'Reilly Media, Inc.", 2010.

[10] Nielsen J. Designing web usability: The practice of simplicity[M]. New Riders Publishing, 1999.

[11] Mayhew D J. Principles and guidelines in software user interface design[M].

Prentice-Hall, Inc., 1991.

[12] Gong J, Tarasewich P. Guidelines for handheld mobile device interface design[C]//Proceedings of DSI 2004 Annual Meeting. 2004: 3751-3756.

[13] Miller, Frederic P., Agnes F. Vandome, and John McBrewster. "Advanced Encryption Standard." (2009).

[14] Daemen, Joan, and Vincent Rijmen. The design of Rijndael: AES-the advanced encryption standard. Springer Science & Business Media, 2013.

[15] Privat, Michael, and Robert Warner. "Transforming and Encrypting Data." Pro iOS Persistence. Apress, 2014. 221-257.

[16] Zeghid, Medien, et al. "A modified AES based algorithm for image encryption."International Journal of Computer Science and Engineering 1.1 (2007): 70-75.