

Project Specification

Project Backlog

- User related operations
 - Login
 - Register
 - Email verification
 - Captcha (to prevent attacks)
- Storage account operations
 - Link an account
 - Show the list of linked cloud disks (along with basic information)
 - Use nicknames and colors to distinguish different accounts
 - Support Dropbox, Google Drive, and OneDrive
- File operations
 - Show the list of files in every net disk in the same view (using color to mark which cloud that file belongs to)
 - Navigate among folders using AJAX (instead of reloading the whole page)
 - Create folders
 - Upload files to one cloud
 - Download files
 - Rename and delete
- Big file operations (files stored in chunks and span multiple clouds)
 - Display in the file list along with normal files
 - Upload (chunked upload to different clouds at client side and store metadata)
 - Download (fetch chunks and reassemble at client side)
 - Compression
 - Erasure coding
- File sharing
 - Contacts (we don't need friends, but this lets users find others quickly)
 - Generate shared links for public access
 - Share folders between chosen users (between users of different clouds!)
 - Upload to and download from the shared folder
 - Version tracking
 - Comments on files

First Sprint Backlog

Owner: Peng Wang (pwang1)

P: Peng J: Judy

- User related operations (P, modify from the social network project)
 - Login
 - Register

- Email verification
- Storage account operations (P: Dropbox & OneDrive, J: Google Drive)
 - Link an account
 - Show the list of linked cloud disks (P: backend, J: frontend)
 - Use nicknames and colors to distinguish different accounts (P: backend, J: frontend)
- File operations
 - Show the list of files in every net disk in the same view (using color to mark which cloud that file belongs to) (P: backend, J: frontend)
 - Download files (P: backend)

Second Sprint Backlog

Owner: Judy Zhu (dzhu1)

- User related operations
 - Captcha (to prevent attacks) (J)
- File operations (P: backend, J: frontend)
 - Navigate among folders using AJAX (instead of reloading the whole page) (J)
 - Create folders (P)
 - Upload files to one cloud (P & J)
 - Rename and delete (P)
- Big file operations (files stored in chunks and span multiple clouds)
 - Display in the file list along with normal files (J)
 - Upload (chunked upload to different clouds at client side and store metadata) (P & J)
 - Download (fetch chunks and reassemble at client side) (P & J)

Third Sprint Backlog

Owner: Peng Wang (pwang1)

- Big file operations (files stored in chunks and span multiple clouds)
 - Display in the file list along with normal files (J)
 - Download (fetch chunks and reassemble at client side) (P & J)
- File sharing
 - Contacts (we don't need friends, but this lets users find others quickly)
 - Generate shared links for public access
 - Share folders between chosen users (between users of different clouds!)

Data Models

Many data such as metadata of big files will be stored in users' clouds, not our db

```
class django.contrib.auth.models.User
```

```
class CloudInterface(models.Model):
```

```
    name = models.CharField(max_length=20, db_index=True)
```

```
    display_name = models.CharField(max_length=30)
```

```
    icon = models.CharField(max_length=30)
```

```
    class_name = models.CharField(max_length=30)
```

```

class StorageAccount(models.Model):
    user = models.ForeignKey(User, db_index=True)
    cloud = models.ForeignKey(CloudInterface)
    identifier = models.TextField()
    status = models.IntegerField()
    credentials = models.TextField(blank=True)
    user_full_name = models.TextField()
    user_short_name = models.TextField()
    email = models.TextField()
    display_name = models.TextField()
    color = models.CharField(max_length=8)

class Contact(models.Model):
    user = models.ForeignKey(User, db_index=True)
    contacts = models.ManyToManyField(User)

class PublicLink(models.Model):
    link = models.CharField(max_length=18, db_index=True)
    cloud = models.ForeignKey(CloudInterface)
    name = models.TextField()
    is_folder = models.BooleanField()
    item_id = models.TextField()
    created_at = models.DateTimeField()
    view_count = models.IntegerField()
    download_count = models.IntegerField()
    permissions = models.IntegerField()

class SharedItem(models.Model):
    cloud = models.ForeignKey(CloudInterface)
    name = models.TextField()
    is_folder = models.BooleanField()
    item_id = models.TextField()
    created_at = models.DateTimeField()
    view_count = models.IntegerField()
    readable_users = models.ManyToManyField(User)
    writeable_users = models.ManyToManyField(User)

```

UI Mockups

- Login

Sign In

Username

Password

[Sign me in!](#) [Get an account](#)

- Register

Sign Up

First name

Last name

Email

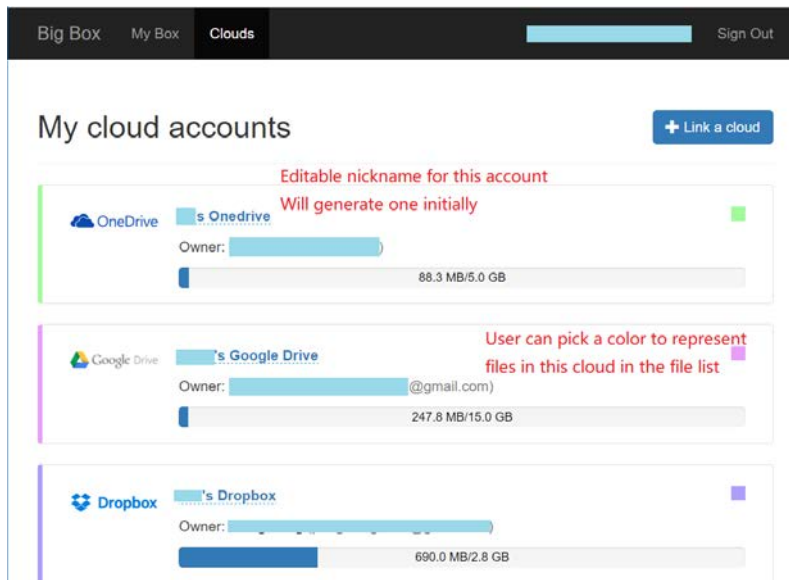
Username (letters and digits)

Password (letters and digits)

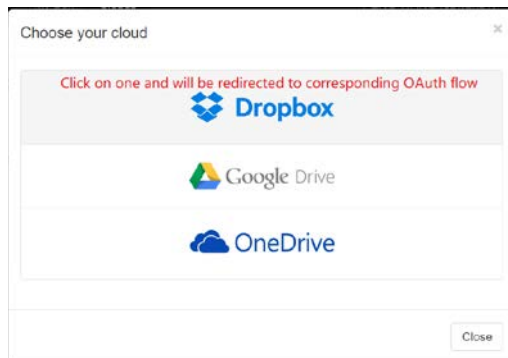
Password again

[Create my account!](#) [Already a user?](#)

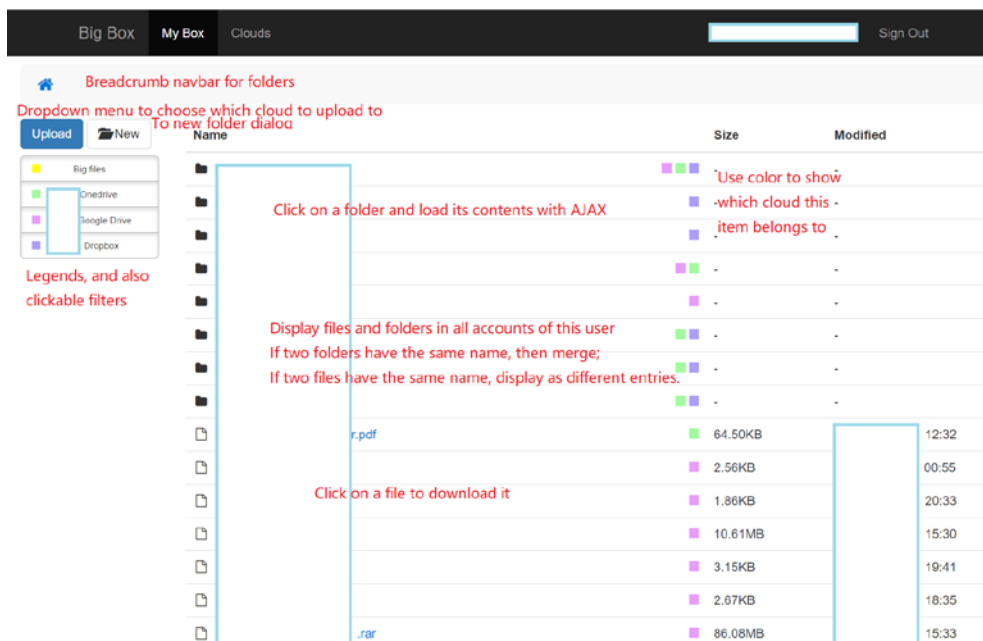
- List of accounts



- Link another account



- File list






- File upload (snapshot from <https://blueimp.github.io/jQuery-File-Upload/index.html>)

+ Add files...

Start upload

Cancel upload

Delete

	6053.jpg	608.42 KB	Start	Cancel
	6083.jpg	440.83 KB	Start	Cancel
	6322.jpg	448.59 KB	Start	Cancel

- Share files (snapshot from Dropbox)


cmu

To: Email or name

Can edit

No link created yet

Create a link



Add people you want to share with

Folder settings