Assignment 3: Building a simplified replica of Dropbox

Barry Denby

April 28, 2021

1 Assignment Information

Course:	MSCC/MSCBD
Stage / Year:	1
Module:	Cloud Computing
Semester:	2
Assignment:	3 of 3
Date of Issue:	2021-03-04
Assignment Deadline:	2021-05-23 (End of week 12)
Assignment Submission:	Upload to Moodle
Assignment Weighting:	24% of Module

2 Introduction

NOTE: read the whole assignment brief first before implementing it contains very important information

In this assignment you will be building a simplifed replica of the Dropbox cloud service. You will be required to have storage available for users where they can create arbitrary directory structures and can upload files and download files from the service. There should also be some access to some basic file sharing between accounts.

It is recommended that the structure of the assignment is closely followed with the brackets as you will need to get the directory structure working first before you will be able to upload and download files or attempt the later brackets after this.

NOTE: This is an individual assignment. It is not a group assignment. You can discuss ideas/algorithms but you cannot share code/documentation

3 Submission and Penalties

You are required to submit two separate components to the Moodle

- An archive containing your complete Google App Engine Python project. The accepted archive formats are: zip, rar, 7z, tar.gz, tar.bz2, tar.xz. The use of any other archive format will incur a 10% penalty before grading.
- A PDF containing documentation of your code. **If you do not provide documentation your code will not be marked.** Copying and pasting code into a PDF does not count as documentation.

There are also a few penalties you should be aware of

- Code that fails to compile will incur a 30% penalty before grading. At this stage you have zero excuse to produce non compiling code. I should be able to open your project and be able to compile and run without having to fix syntax errors.
- The use of libraries outside the SDK will incur a 20% penalty before grading. You have all you need in the standard SDK. I shouldn't have to figure out how to install and use an external library to get your app to work
- **An omission of a git repository attached to your email address that is registered for GCD will result in your application and documentation not being graded.**
- The standard late penalties will also apply

You are also required to submit as part of your archive a working Git repository.

- When I unpack your archive there should be a .git directory as part of it.
- This should be a fully working **local** git archive. It should not require access to a remote repository
- You are not permitted to upload your work to Github, Gitlab, or any other publicly visible git repository (assignment will be marked as a zero if it is)
- If you need a remote git repository the only permitted one is the college provided Gitlab which can be found at gitlab.griffith.ie
- There must be a minimum of seven commits in the git repository, one per completed bracket.

Very Important: Take note of the grade brackets listed below. These are meant to be completed in order. If you skip a bracket or do not complete a bracket following brackets will not be considered for marking. You should be well capable of producing strong and generally robust software by now. For example if there are six brackets and you fail the third one, then the fourth, fifth, and sixth brackets will

not be marked. Documentation brackets will be treated separately from Coding brackets.

You should also be aware that I will remove marks for the presence of bugs anywhere in the code and this will incur a deduction of between 1% and 15% depending on the severity. If you have enough of these bugs it is entirely possible that you may not score very many marks overall. I want robust bug free code that also validates all user input to make sure it is sensible in nature.

Also note that the percentage listed after the bracket is the maximum mark you can obtain if you complete that many brackets without error. Everything in all brackets is mandatory.

4 Plagiarism

Be aware that we take plagiarism very seriously here. Plagiarism is where you take someone else's work and submit it as if it was your own work. There are many different ways plagiarism can happen. I will list a few here (this is not exhaustive):

- Finding something similar online (full implementation or tutorial) that does the same job and submit that.
- Finding something similar online (full implementation or tutorial) and transcribing (i.e. copying it out by hand)
- Working together on an individual assignment and sharing code together such that all implementation look the same.
- Getting a copy of someone else's code and submitting/transcribing that
- Doing any of the above and attempting to conceal it by moving functionality around and renaming functions and variables.
- Paying someone to do your assignment
- Logging into someone elses Moodle account, downloading their assignment and uploading it to your own Moodle account.

I've had to deal with many cases of plagiarism over the last six years so I can spot it and diagnose it easily, so don't do it. To prevent plagiarism include but not limited to the following:

- Do all your code by yourself
- Don't share your code with anyone, particularly if anyone looks for a copy of your code for reference.
- Don't post your code publicly online. Remember the use of GitHub, Gitlab, BitBucket etc is prohibited.
- If you need to find information online only query about very specific problems you have don't look for a full assignment or howto.

- Change the default password on your Moodle account. The default password can be determined if someone is connected to you through social media or they get one or two details from you.

Be aware that if you submit your assignment you accept that you understand what plagiarism is and that your assignment is not plagiarised in any way.

5 Coding Brackets (80%)

1. Bracket 1 (10%)
   - Write the shell of an application that has a working login/logout service.
   - Create models of a user, directory, and file using appropriate datatypes.
     - If a user logs in for the first time a user model should be created for them.
     - a default root directory (path of /) should also be created for this user.
2. Bracket 2 (20%)
   - Add the ability for a user to create a directory (Bracket failure if the same directory name is allowed in the current directory)
   - Add the ability for a user to delete a directory (Bracket failure if the wrong directory is removed)
3. Bracket 3 (30%)
   - Add the ability to change into a directory
   - Add the ability to go up a directory with the special entry (path of ../)
   - If a user is in their root directory don't display (path of ../)
4. Bracket 4 (40%)
   - Allow the user to upload a file to the current directory and store it in the cloud storage bucket. This should not overwrite a file that is already there. If a file is already there ask the user if they want to overwrite it.
   - Allow the user to delete a file from the current directory (Bracket failure if the wrong file is deleted.
   - Allow the user to download a file from the current directory to the local machine.
5. Bracket 5 (50%)
   - Prevent the deletion of a directory that still has files remaining. Provide an appropriate warning to the user.
   - Prevent the deletion of a directory that still has directories remaining. Provide an appropriate warning to the user.
6. Bracket 6 (60%)
   - Add in the ability to detect duplicate files in the current directory. Display the files that match
   - Add in the ability to detect duplicate files in a user's entire dropbox. Display the files and paths that match

7. Bracket 7 (70%)
    - Add the ability to share files read only between multiple user accounts
8. Bracket 8 (80%)
    - UI design: Well thought out UI design that is intuitive and easy to use.

6 Documentation Brackets (20%)

NOTE: Documentation should be around 1,700 words in length total

9. Bracket 9 (15%): Document every method in your code from a high level perspective. i.e. give an overview of what the method does. Do not copy and paste code you will be penalised for this.
10. Bracket 10 (20%): Document every datastructure and database design you have used in your code and why you chose them. You do not need to provide an E-R diagram for database designs.