**SYNC TOOL**

**Team members:** Nick Baker, Marcus Waller, Anudeep Metuku, Ana Cehan, Thomas Luo, Carlos Montemayor, Emmanuel Ankutse, and Sasha Holt

**RFP:** This proposal is for Sync Tool.

**Milestones**

2/6 - Proposal Acceptance

2/6 - Schedule

2/15 - Storyboard

2/22 - Design/Decomposition

2/24 - Design Review

3/2 - Test Plan Review

3/9 - Code Complete

3/11 - Unit Test

3/18 - Integration Complete

3/20 - Demo

3/23 - Integration Testing Complete

4/1 - Delivery and Final Acceptance Test

**Problems needed to be solved**

* How to dynamically get all the folders and locations of files so that they are backed up in the same folder hierarchy as the original files.
  + We can solve this by standardizing the files as bytes and storing their types as enums or strings, where we can reconstruct the file after we have all the actual byte data by using set templates depending on the file type
* How to read data from any type of file
  + There are a set number of ways to read information from files, but we could use the generic ‘File’ class to store the structure and create a series of templates for specific file types, or use resources from File-related classes to copy the information
* Efficiently and effectively discerning the changes made to the current directory
  + By keeping track of every set of changes, we only have to differentiate between the current and the previous, making the information stored minimal. In the case of trimming backups we will simply have to merge the data from the oldest with the second oldest, and so on until the nth oldest backup
* Detecting partial backups
  + We can set some type of close behavior, find methods that automatically execute to close the program and insert code to signal that the backup isn't complete
  + We could have a signature mark the end of a backup, and detecting it not being there signifies a partial backup
* Stopping the backup if the program closes unexpectedly
  + We can have automatic behavior that happens when the program is requested to close stop the data transfer process
  + Encapsulate the data transfer in an try catch
* Dealing with multiple partial backups
  + Force the user to deal with all partial backups in sequence
  + Delete them and store reminders to prompt the user to deal with them manually
  + Have an setting to automatically deal with them in a certain way

**Ambiguities in RFP or suggested changes/simplifications**

* If we back up data from a new location, does it overwrite the old backup data?
* By “changing the location of the current data”, does it mean moving the data in the directory you want to back up from one folder to another or changing the directory you need to back up?
* Do we keep a backup of each separate state or do we have to note how the backup changed for each state before the most current one?