TEAM 2: WILL CODE FOR A'S
SYSTEM REQUIREMENTS
MIRROR APPLICATION

CSC-478B Fall 2015

TITLE:	**SYSTEM REQUIREMENTS**
AUTHORS:	**GREG PALEN, ZACK BURCH, ASHLEY ROBERTSON**
DATE:	**DECEMBER 14, 2015**
TYPE:	**FINAL**
VERSION:	**1.0.0**
DOC ID:	**REQ-01**
REVIEW DATE:	**DECEMBER 13, 2015**
ACCEPTED DATE:	**DECEMBER 13, 2015**

INTRODUCTION

1. Project Scope

- The software shall run on a minimum of Windows 7 Home Edition inclusive of the general requirements to run Windows 7 Home.
- The software shall run on a system meeting the minimum requirements for Windows 7 Home from a CPU, storage, and memory point of view. It is not anticipated that this application should require any “measureable” additional compute resources beyond what Windows 7 requires.
- The software will require at least Oracle Java 8 to be installed on the user’s computer. Further, Java 8 will need to be configured such that access to the local file system is allowed.
- As is evident from the above, the application will be written in Java.
- The user will need to have sufficient disk space to hold the copy of the selected file set.
- If the user chooses to use the application with a cloud storage provider, the user shall install the appropriate client application to enable the local file system to interact with the cloud storage like it was a local drive or folder (i.e., similar to a “shared drive”).
- The application shall have a graphical user interface.
- The application shall function as a self-contained, standalone application that has no external requirements other than as noted above.
- Detailed project requirements with version assignment are presented in Appendix A of this document.

2. Definitions, Acronyms, & Abbreviations

Item	Description
Cloud storage	A network-based, remote storage product capable of storing files and folder generated by a user. Generally requires a subscription or account. See https://www.google.com/drive/ as an example.
Desktop client	An application software that runs on the user’ s desktop and provides an interface to a cloud storage service for the user’ s native operating system. See https://www.google.com/drive/download/ as an example.
Mirror	Multiple definitions: 1) This application 2) To duplicate, as in a file or folder, to an alternate location
Backup	Synonym to definition #2 for Mirror above. To make an identical copy of an existing file to an alternate location for safekeeping.
Shared drive	A Microsoft Windows feature that allows a hard drive on a different computer to be accessed by the local user as if it were physically connected to the local user’ s computer. See

	http://windows.microsoft.com/en-us/windows/create-shortcut-map-network-drive#1TC=windows-7.
UI	User interface – the screens the user sees when interacting with an application or system.
FileSet	Mirror’ s basic definition of the user’ s preferences for the Mirror application. The user interface provides access to the default FileSet. A FileSet describes the mirroring operation settings as a list of source files to be mirrored, a mirror name (used to name both the FileSet descriptor itself as well as the name of the folder the mirror will be created within), and a destination (the named mirror folder is created inside the destination directory).

3. References

Oracle Java: Copyright © 1993, 2015, Oracle and/or its affiliates. All rights reserved. (http://docs.oracle.com/javase/8/docs/legal/cpyr.html)

SwingX is a subproject of Swinglabs, was started in January 2011 and has 1867 members. The project administrators are rbair, swinglabs, kleopatra, Alexander Potochkin, Karl Schaefer, and Jan Haderka. - See more at: https://swingx.java.net/#sthash.

4. General Description

Product Perspective

Cloud computing continues to grow in popularity amongst both consumers and business users. One of the early applications of “the cloud” was for centralized storage of files for both primary storage as well as for backup purposes. Mozy (www.mozy.com) and Google Drive (drive.google.com) are both examples of early entrants into this space.

As these services have evolved, users enjoy the large amounts of storage available to them at low prices but to date, none of these services offer any kind of easily-used “mirroring” functionality that would allow you to keep your files locally but quickly and easily make a backup or “mirror” to a cloud storage service.

Mirror was designed to address these shortcomings by providing a simple application that users can easily set up to mirror specific files to some alternate destination. Mirror takes advantage of the fact that the majority of cloud storage providers offer desktop client integration where the cloud storage appears to the local file system to be locally attached, just like a USB drive or internal hard drive. This simplifies the implementation of Mirror by allowing development to focus on ease of use and feature set rather than trying to implement the public API of each and every storage provider out there. By installing the desktop clients provided by companies like Google (Drive for Windows, etc.), Mirror can copy source files and folders to the Google Drive folder on the local file system and Google’s client application takes care of the actual upload to the cloud. Thus, Mirror works out of the box with virtually any storage medium that appears as a local resource (it can appear as either a local drive, like a Windows “shared drive” would, or a local folder, like Google Drive does).

Product Functions

Mirror Version 1 allows the user to define a list of files that the user would like to be mirrored (backed up) as well as a destination of where to mirror the files to and a name for the backup “set” (this becomes the name of the directory the backup goes into as well). The original file system hierarchy is replicated inside the destination directory making it easier for the user to put the backed-up copy back into its original location if need be. Future versions of Mirror include functionality such as running on a schedule as well as encryption and compression.

User Characteristics

The typical end user for Mirror is a standard computer user with no particular set of skills or knowledge. Basic understanding of how to install a program is required, and knowing how to navigate a File Dialog Box (in order to find the files to be backed up and the destination for the backup) is about all that is required. To use with a cloud storage provider, the user would also need an account with that provider and to install the requisite desktop client software offered by that provider.

General Constraints

Mirror relies on Java 8 and as such, the user must have Java 8 installed on their computer or the application will not run (though installation is permitted even if Java 8 is not present). Mirror has been tested on both recent Mac and Windows operating systems and functions properly on both platforms. At this time, the program allows the user only to define files to back up and not entire folders (to back up an entire folder, the user simply needs to select each and every file contained in that folder).

In order to work with a cloud storage provider, there is also a constraint that the cloud storage provider offers a client (which the user must acquire and install separately) that makes the cloud storage look like locally-attached storage.

Assumptions and Dependencies

Requires Java 1.8 to be installed on the user’ s computer.

Tested and supported on at least Windows 7 or Mac OS X (note: prior versions of those operating systems may in fact work as long as they run Java 1.8; likewise, Linux operating systems should work as well assuming Java 1.8 is installed. The developers consider Windows 7 the “required” operating system).

The application makes use of the “swingx” library for some user interface functions. This dependency is included in the installer so no user action is required.

- The rest of this page intentionally left blank -

SPECIFIC REQUIREMENTS

1. Non-functional Requirements

1.1. The software shall run on a system meeting the minimum requirements for Windows 7 Home from a CPU, storage, and memory point of view. It is not anticipated that this application should require any "measureable" additional compute resources beyond what Windows 7 requires.

1.2. The software will require at least Oracle Java 8 Runtime (JRE 1.8) to be installed on the user's computer. Further, Java 8 will need to be configured such that access to the local file system is allowed.

1.3. As is evident from the above, the application will be written in Java.

1.4. The user will need to have sufficient disk space to hold the copy of the selected file set. The software shall validate available space in the destination before allowing the user to save the configuration or start an on-demand mirror/backup operation.

1.5. If the user chooses to use the application with a cloud storage provider, the user shall install the appropriate client application to enable the local file system to interact with the cloud storage like it was a local drive or folder (i.e., similar to a "shared drive").

1.6. The application shall have a graphical user interface.

1.7. The application shall function as a self-contained, standalone application that has no external requirements other than as noted above.

1.8. Installation of the software will be accomplished by an installer program to minimize complexity for the user.

1.9. The software shall operate in such a way as to not excessively utilize CPU resources.

1.10. To work with a cloud storage provider, that provider must offer a desktop client for Windows that allows the cloud storage to appear as a folder or drive to the local file system.

2. Functional Requirements

	Version 1
1.1.1.1	The user must be able to create a list of files that the application is to operate on.
1.1.2.1	The user shall be able to specify a destination folder or drive for the backup operation to store the backup into.
1.1.3.1	The user must be able to modify the list of files to operate on prior to backup. After backup, the user can modify the list and save as a new backup.
1.1.4.1	The user shall have the capability of executing a backup on demand.
1.1.5.1	The user must be notified of the status of any backup (failure or success).
1.1.6.1	The installer shall check the target machine for compliance with the minimum system requirements defined in Section 1 above.
1.1.6.2	Is at least Java 8 installed?
1.1.6.3	Is the host operating system at least Windows 7?
1.1.6.4	Does the installation location have enough space to hold the installation?
1.1.7.5	The user shall name a backup, and the backup shall be saved in a folder with the chosen name at the root of the destination.
	Version 2
2.1.1.1	The system shall log the successful copying of each individual file.
2.1.2.1	Each log entry will contain a date/time stamp.
2.1.3.1	The application shall record in the log any file system errors that occur when trying to copy. If the error is a file error, it shall be logged and the system will continue with the next file in the backup set (meaning an error should not abort the entire backup if at all possible).
2.1.1.1	The user shall be able to view the logs in the user interface.
	Version 3
3.1.2.1	The application shall prevent the backup operation if there is not sufficient space to store the source files in the destination location.
	Version 4
4.1.1.1	The user must be able to create a list of folders that the application is to operate on.
4.1.1.2	The system shall log each successfully copied folder. A folder shall only be deemed completely copied if all of its contents (recursively) have been successfully copied.
4.1.2.1	The user must be able to modify the list of folders to operate on.
4.1.3.1	If the user specifies a folder, the complete contents of the folder shall be copied including hidden files.

	Version 5
5.1.1.1	The user must be able to schedule a backup operation to occur on some recurring interval.
5.1.2.1	Backup operations shall happen in the background and should minimize the impact to the user’s machine.
5.1.3.1	The application shall use the file modification date and file size of the source file to determine if the file has been changed since the last backup. ONLY changed or new files will be copied. Existing files that have had no change shall be skipped.
5.1.4.2	The log shall note if files/folders were skipped due to no change detected.
	Version 6
6.1.1.1	The system shall support the capability to apply various optional transformations to the backup during the operation (such as encryption and compression)
6.1.1.1	The user shall be able to optionally encrypt the backup.
6.1.1.2	The backup will be encrypted using a user-defined passphrase
6.1.1.3	The encryption will be handled via open-source encryption libraries
	Version 7
7.1.1.1	The user shall be able to optionally compress the backup.
7.1.1.2	Compression will be handled via open-source compression libraries
7.1.1.3	Transformations must be able to be combined (ie, ability to both compress and encrypt the backup)
	Version 8
8.1.1.1	If a backup operation does not complete entirely, the user shall be given the option to repeat the backup. If the user elects to complete the backup immediately, the backup will only copy those files that did not successfully complete on the immediate prior backup.