Appendix D. Test Cases

There are few precondition before the test, user must have their own ftps server set up, and they must have access right to the server. Due to the fact that Android on some devices has a bug blocking application to set last modification date of a file using standard Java function call, root access is needed in order to workaround the issue. I tested each functional requirements set at the beginning of the project, and the results are listed below.

Testing Environments

- *Mac OS X platform:*
  - *Macbook Pro (Retina, 13-inch, Mid 2014), OS X 10.10.3 Yosemite*
- *Android platform:*
  - *Nexus 6, Android 5.1.1*
- *FTPS Server:*
  - *Own FTPS Server located in Hong Kong*

Test Case: 1	**Test Purpose:** The application should store a master copy of selected files on FTPS server.	
Expected Outcome	**Actual Outcome**	**Pass/Fail**
Remote server should store the latest files and directories	During synchronization, the application will upload the latest files from local client and remove deleted directories	Pass
Comments: The server does store the latest copy of selected files. To enhance, the application can implement a backup or versioning system to store multiple copies.		

Test Case: 2	**Test Purpose:** The application should do file exchange and connection using FTPS protocol.	
Expected Outcome	**Actual Outcome**	**Pass/Fail**
All connections and transfer actions between client and server should run using FTPS protocol	Both applications do file exchange and all kinds of data connections under ftps protocol	Pass
Comments: The system works as intend. To enhance to compatibility, the application can implement a lower security level ftp protocol.		

<table>
<tr><td><b>Test Case:</b> 3</td><td colspan="2"><b>Test Purpose:</b> The application should download updates when master copy on the server changes.</td></tr>
<tr><td><b>Expected Outcome</b></td><td><b>Actual Outcome</b></td><td><b>Pass/Fail</b></td></tr>
<tr><td>In a given time frame, the application will check and download latest files from server.</td><td>The synchronization trigger every 5 minutes. By comparing local and server directory list, the application will download updates from server.</td><td>Pass</td></tr>
<tr><td colspan="3"><b>Comments:</b> The system works as intend. However, if client modify files directly on ftps server without the use of applications, detection may be wrong.</td></tr>
</table>

<table>
<tr><td><b>Test Case:</b> 4</td><td colspan="2"><b>Test Purpose:</b> The application should upload files when local copy changes</td></tr>
<tr><td><b>Expected Outcome</b></td><td><b>Actual Outcome</b></td><td><b>Pass/Fail</b></td></tr>
<tr><td>In a given time frame, the application will check and upload latest files to server.</td><td>The synchronization trigger every 5 minutes. By comparing local and server directory list, the application will upload changes to server.</td><td>Pass</td></tr>
<tr><td colspan="3"><b>Comments:</b> The system works as intend. However, if client modify files directly on ftps server without the use of applications, detection may be wrong.</td></tr>
</table>

<table>
<tr><td><b>Test Case:</b> 5</td><td colspan="2"><b>Test Purpose:</b> The application should run in the background without command synchronous action manually or any user interaction after the initial setup.</td></tr>
<tr><td><b>Expected Outcome</b></td><td><b>Actual Outcome</b></td><td><b>Pass/Fail</b></td></tr>
<tr><td>The application will able to start synchronization automatically in schedule.</td><td>User can set the interval to trigger sync action on both applications</td><td>Pass</td></tr>
<tr><td colspan="3"><b>Comments:</b> More testing time is needed. If some bug block or crash the application, the application would not able to trigger sync operation in time.</td></tr>
</table>

<table>
<tr><td><b>Test Case:</b> 6</td><td colspan="2"><b>Test Purpose:</b> The application should handle conflicts like updating the same file at the same time.</td></tr>
<tr><td><b>Expected Outcome</b></td><td><b>Actual Outcome</b></td><td><b>Pass/Fail</b></td></tr>
<tr><td>The application will let user to select which files is the latest one when conflict occurs.</td><td>A popup dialog shown when file conflict occurs in Android application.
User able to choose whether they want to select each files when conflict occurs or set a default action.
Mac user only able to set default upload or download action.</td><td>Android: Pass
Mac: Fail</td></tr>
<tr><td colspan="3"><b>Comments:</b> Current implementation of popup dialog may be good enough to ask user to solve conflicts. A better solution will be implement a queue list to show which files are awaiting user to decide which file is the latest one.</td></tr>
</table>

<table>
<tr><td><b>Test Case:</b> 7</td><td colspan="2"><b>Test Purpose:</b> The application should handle users that offline for a long time of period.</td></tr>
<tr><td><b>Expected Outcome</b></td><td><b>Actual Outcome</b></td><td><b>Pass/Fail</b></td></tr>
<tr><td>The application will still able to synchronise suitable files after offline for a long time of period.</td><td>The application uses version number for each file on each client and server. Client able to tell other client’s version number, and compare which version is the latest one.</td><td>Pass</td></tr>
<tr><td colspan="3"><b>Comments:</b> As mention in the report, the synchronization process is done by using version number instead of timestamp of the file. Therefore, even client offline for a long time, the application will still able to tell which file is the latest one. In addition, if the system fail to distinguish, it will ask user to select. An alternative solution can be implemented to notify user they offline for a significant long time of period, and ask them whether sync or overwrite all files in selected directory.</td></tr>
</table>

<table>
<tr><td><b>Test Case:</b> 8</td><td colspan="2"><b>Test Purpose:</b> The application should consider battery usage on mobile devices.</td></tr>
<tr><td><b>Expected Outcome</b></td><td><b>Actual Outcome</b></td><td><b>Pass/Fail</b></td></tr>
<tr><td>The application gives users control to customise interval of check time.
Android application gives users control to synchronous only in Wi-Fi network, and/or only in charging state.</td><td>Both applications let user to set schedule sync enable or not. In addition, the sync trigger interval. Android user can set sync only in certain network condition and power state condition.</td><td>Pass</td></tr>
<tr><td colspan="3"><b>Comments:</b> A longer running test may be needed to ensure battery usage. Ideally, the design and these preference settings should control the uses of battery.</td></tr>
</table>

<table>
<tr><td><b>Test Case:</b> 9</td><td colspan="2"><b>Test Purpose:</b> The application should avoid data loss while updating local and server’s data.</td></tr>
<tr><td><b>Expected Outcome</b></td><td><b>Actual Outcome</b></td><td><b>Pass/Fail</b></td></tr>
<tr><td>The application do not delete any files before it ensure files updated successfully in order to avoid data loss.</td><td>The application will first rename the old file with current time append to file name. After new file is either upload or download, the old file will be deleted.</td><td>Pass</td></tr>
<tr><td colspan="3"><b>Comments:</b> Current implementation can ensure at least one file is keeped. A better solution will be implement a backup and recycle bin <i>(More details in future enhancement section)</i>.</td></tr>
</table>

<table>
<tr><td><b>Test Case:</b>
10</td><td colspan="2"><b>Test Purpose:</b> The application should be easy to select or remove directories to be synchronised.</td></tr>
<tr><td><b>Expected Outcome</b></td><td><b>Actual Outcome</b></td><td><b>Pass/Fail</b></td></tr>
<tr><td>The application contains a folder picker, instead of forcing user to enter the complete path manually.</td><td><i>Android:</i> Both local and remote directory can be selected from a folder picker activity.
<i>Mac OS X:</i> Local directory can select from native folder picker, and remote directory can select from a scrolling list under remote path entry.</td><td>Pass</td></tr>
<tr><td colspan="3"><b>Comments:</b> Both application work as intended, user can select directory easily. A better solution will be update the folder picker in Android to match material design, and a faster ftps directory picker. Mac OS X solution for remote directory picker should seek a better implementation that feel as native as local directory picker. Both recommendations are for better user experience and user interface <i>(More details in future enhancement section)</i>.</td></tr>
</table>

<table>
<tr><td><b>Test Case:</b>
11</td><td colspan="2"><b>Test Purpose:</b> The application should be easy to add or remove files in selected synchronize directory.</td></tr>
<tr><td><b>Expected Outcome</b></td><td><b>Actual Outcome</b></td><td><b>Pass/Fail</b></td></tr>
<tr><td>The application will add and delete files during synchronization.</td><td>During synchronization, current directory list will compare with the previous to check modified files. New files will upload to server or download from server. Deleted files will remove from both client and server.</td><td>Pass</td></tr>
<tr><td colspan="3"><b>Comments:</b> The system works as intended.</td></tr>
</table>

Test Case: 12	**Test Purpose:** The application should be implemented on both Mac (Desktop) and Android (Mobile) platform.	
Expected Outcome	**Actual Outcome**	**Pass/Fail**
The project produce a Mac OS X desktop application and an Android mobile application.	A prototype application for each platforms, Mac OS X and Android , are done.	Pass
Comments: More testing is needed to ready for prime time. For more future enhancements, check below section.		

Test Case: 13	**Test Purpose:** What happens if directory disappear after last sync?	
Expected Outcome	**Actual Outcome**	**Pass/Fail**
Ask user to choose whether continue synchronise, i.e. delete the whole folder, or act as new directory, i.e. upload/download the whole folder.	Synchronise normally, if directory exist on either local or remote side. Data loss if this is not what user expected.	Fail
Comments: Some precaution should be implemented as states in the expected outcome, in order to ensure no data loss in practical situation.		

Test Case: 14	**Test Purpose:** What happens if network connection unavailable?	
Expected Outcome	**Actual Outcome**	**Pass/Fail**
Notify user they do not have a good network connection to proceed the next action	Android application will show connection unavailable. Mac OS X application will stop working without any notice.	Fail
Comments: Both applications need to implement a better solution to let user know their network connection affects the application to synchronise data.		

<table>
<tr><td><b>Test Case:</b>
15</td><td colspan="2"><b>Test Purpose:</b> What happens if application does not have a valid ftps settings or cannot connect to server?</td></tr>
<tr><td><b>Expected Outcome</b></td><td><b>Actual Outcome</b></td><td><b>Pass/Fail</b></td></tr>
<tr><td>Notify user they do not have a valid ftps server settings.</td><td><i>Android:</i> When synchronization starts, a notification will be shown to notify user. When launching the application, user can only modify preferences or exit the app.
<i>Mac OS X:</i> A notification will be shown. In preference window, after user save the ftps settings, the system will check if user have a valid setting and show a message on the screen.</td><td>Pass</td></tr>
<tr><td colspan="3"><b>Comments:</b> To educate first-time user, a user guide will be great <i>(More details in future enhancement section)</i>.</td></tr>
</table>