Daniel Ro
File Syncing Protocol with Merkle Trees

Step 1: Client and Server construct merkle trees from their respective directories.
Step 2: Client establishes web socket connection to Server.
Step 3: Client sends "push" or "sync" to Server to inform Server of what type of action is occurring.

If request is "sync" then the below steps occur. If request is "push" then the same steps occur but the roles of Client and Server are reversed.

Step 4: Client sends the root node hash of merkle tree to server.
Step 5: Server compares this hash to the corresponding node hash of its own merkle tree.
Step 6: If the hashes are the same, the Server does nothing.
Step 7: If the hashes are different and the node we are looking at is a directory, the Server requests the Client's child directory hash values. If the node we are looking at is a file, the Server does not make a request, but simply adds this node to a list it keeps locally called ModifiedFiles.
Step 8: If the Server requests more child directory hashes than there are child directories in the Client, Client adds the extra child nodes requested to a list it keeps locally called MissingNodes. This list contains all the sub directories that the Client is missing. If the Server requests fewer child directory hashes than there are child directories in the Client, Client deletes the child directories that were not requested for. This must have meant the Client had extra sub directories so we need to get rid of them.
Step 9: Client provides the requested hashes to Server.

Continue looping from steps 5-9 until Server makes no more requests to client. At this point, the ModifiedFiles list, which is contained locally on the Server, contains the list of files which have been modified in Client. Also, the MissingNodes list, which is local to the Client, contains the list of directories which the Client is missing. Lastly, any extra subdirectories in the Client have been deleted.

The next step of the protocol is for the Client to get the sub directories it is missing. These are in the MissingNodes list which the Client has. Repeat the steps below until all the missing directories have been taken care of.

Step 10: Client requests missing directory from Server.
Step 11: Server returns the missing directory to Client.
Step 12: Client adds the missing directory in the correct location.

Now we have added the missing files to the Client directory, but we still need to update the files which exist in both Client and Server, but have differences. The ModifiedFiles list, which is contained in the Server, contains a list of these files. Repeat the steps below until all the modified files have been synced.

Step 13: Server sends File Path of modified file to Client.
Step 14: Server sends actual File to Client.
Step 15: Client uses the File Path to go to the correct location in the directory, delete the existing File, and add the new File sent by the Client.

At this point, the directories on both computers should be completely synced, so the Server closes the web socket. Hooray!

If the Client is attempting to push its modifications to the server, it's exactly the same, except in steps 4-15 the Client and Server's roles are reversed.