In the following section the requirements that came from our study of the problem will be discussed as well as the architecture and design we are going to use to satisfy them.

**Use cases**

After considering the problem and different possible solutions we came up with the following use cases.

* User can create an account
* User can log in into his account
* User can log out from his account
* User can delete his account
* User can see his files that are both local and online
* User can upload a local file to the server
* User can download a file from the server
* User can update a file on the server
* User can delete a file both locally and on server
* User is notified if there is a conflict
* User can handle a conflict if it occurs

**Architecture design**

We decided that the best way to deliver the above functionality to the user will be through a client server architecture. In particular the client will communicate with the server in the back to send and receive data. This allows for modularity because we can have different clients in different platforms talking to the same server. Specifically, as shown in the figure we are going to have a mobile app and desktop app clients that would provide users with a graphical user interface to interact with the server. The server which runs on a Linux operating system will store all the file’s and user’s information into a database and the files themselves will be stored in the Linux file system.

A close up of text on a white background

Description automatically generated

**Project Objectives**

We set the following objectives for the implementation of the system. Specifically, what needs to be implemented for the clients and the server and the priority of each objective

A screenshot of a cell phone

Description automatically generated

In the figure we can see each one of the objectives having a colour that states its priority. Green objectives represent the bare minimum functionality of the corresponding component and should be completed first. Yellow objectives represent important functions of the subsystems and should be done immediately after. On the contrary, red objectives represent features that are not essential for the system to be operational but could potentially be added to expand the services it provides. Finally, blue objectives are quality of life improvements that will be implemented only if the quality of the system has been thoroughly tested and time allows for their development.

All the functionality above will be able to handle almost any kind of file with the exception of directories. We decided that it would be best not to support the synchronization of directories as it would be too complex for us to store the structure of directories and implementing it within our time constrains for the project. Also having directories could cause performance issues as a user may create infinite number of sub directories.

**Conflicts Handling**

We discussed thoroughly about what is the best way to go with handling the conflicts and how to detect them in the first place. Firstly, for the detection of a conflict we came out to the conclusion that the following occasion should qualify as a potential conflict. If a user tries to upload a file without already possessing the latest version of that file will qualify as a potential conflict because it means that someone else has updated the latest version before he had the chance to do so and therefore his version will be different.

Now to handle that conflict as soon as a user tries to upload a file that causes the conflict, we will give to him three options. To either upload the file anyway and replace the latest version or just get the latest version first or a third option that a diff file will be presented to him so he can handle the conflict on his own. The diff file will display to him the differences between the version he is trying to upload and the current latest version on the server.

**UI Design**

We have attempted to do the user interface as friendly as possible both for the mobile and desktop applications, implementing the latest guidelines in material design. Also, we made the two client interfaces as consistent as possible to ensure the user can adapt between the two platforms.