# Documentation

GUI

The applications GUI is built using with three main screens

* The main screen: where the user's statistics on how much time she\he spends on every app is seen in various forms. The different options can be seen via "fragments", which appear as tabs on the user's screen. The options in the "Time" options are: day, week and month. Only one of these options is seen on the users screen at a time (radio buttons behavior). The second option changes the type of fragments - the "Stat Appearance Type", which also can only be seen one at a time on the screen. Options are: Pie chart, list and graph.

The different types of graphs (pie chart, line graph) fragments were implemented using a library accessible via Github - [MPAndroidChart](https://github.com/PhilJay/MPAndroidChart) .

* The blocked app screen: here we have three basic options. Adding an app to the blocked list, editing and removing existing rules. The blocked list is a list while the other options are regular buttons connecting the app to the intent in need.
* The "add blocked app": here we have two options picked via radio button and a list of the "blockable" apps. The apps that are System based are obviously "unblockable", and thus do not appear on this list.

Backend

The app is highly based on Firebase. WE give each user a unique identity which is based in his SIM number, phone id, and android id. Every 10 seconds a background service runs that documents the app that is currently being used and the timestamp of the current time.

It is then possible to collect this information via Firebase according to the time that it was used and present the use of all the apps and their respective times.

Each user also has a Firebase based list with all of his blocked apps and the type of rules that are used. These are updated on the app the moment the user leaves the "blocked apps" activity. Thus giving an option to search the currently used app and compare it to the list. Should the currently used app be on the list the phone will show a notification stating the fact that the user has passed the elapsed time allowed.

The application detects the running app by noting the app that is running in the foreground using the service mentioned earlier. In phones using android with versions predating API 21, the information will be accessed using the getRunningTask(0) command which has since been deprecated. The phones after this API, will use a method where the last app running is recorded, using the UsageStats.lastRunning().