- **1.1 Introduction**

The application built for this assignment is designed to take orders for teas, coffees and beverages whilst entertaining guests. Growing up, my parents always had folks over and the question was always asked; who wanted tea and who wanted coffee? Further, there was always the awkward person who wanted something else. Even at meetings I've been to, this issue crops up. Even if you get the right number there's always the confusion over who had milk and who had sugar. I thought that this could be done in a much simpler way, through an app. The app could be passed around the group with individuals filling in their order and no fuss would be needed.

- **1.2 Scope and Content**

The application should be able to handle the input of orders for a given event. The user will name the event (i.e. "Family get together", "meeting") and the app should allow either Tea, Coffee or 'Other' drinks to be selected. There should be an option for the amount of sugar required and an option for milk level. The latter should be shown graphically and there should be good user feedback given throughout usage. When all orders have been done, the host should be able to see the information in a clear way and this should end up making the whole process easier.

- **1.3 Resources and Reading**

As well as the Module workbook and notes available through the website, I have extensively used online sources. Most notably 'Stack Overflow' for aid with problems and Google's own android documentation. I have also researched both the IOS App Store and Google's Play Store. On the App Store there is one app available like this, however this app looks basic and not very aesthetically pleasing and the user can only select pre-selected items. I can find no apps specifically like my one on the Play Store however there are plenty of coffee apps and tea brewing ones available for download.

Fig 1.1 – IOS Alternative

In terms of physical resources, I am using android studio to build and design my app. Due to not having an android device personally, I am virtualising one via Android studio. While not gaining all the features of a physical device this works just fine for the developmental phase. I will also be using source control via GitHub and SourceTree throughout.

- **2.1 Software Design**

When designing the app I thought it would be of help to create a flowchart in order to model the software. This aided in clarifying the steps needed and decisions required at each stage of the implementation.

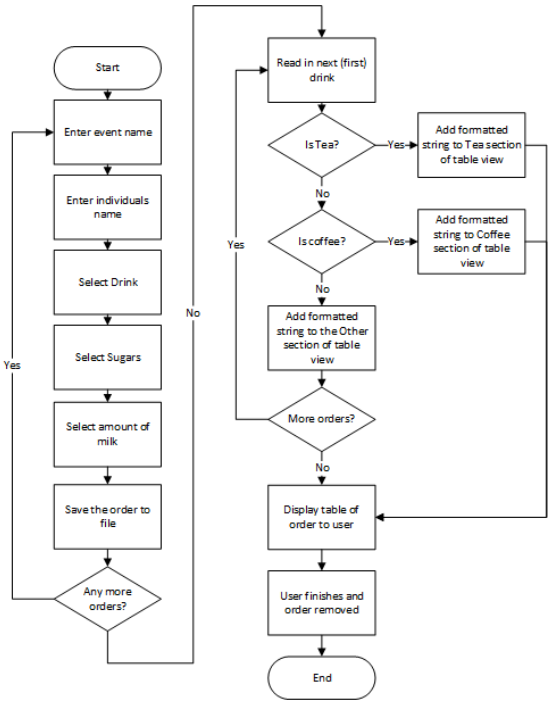

Fig 2.1 Flowchart

As well as the flowchart I also decided upon making a data flow diagram, this clarified how the data would be stored. I would use a file that was named after the event. This would be used in each activity in the app and then when the user finishes the file would be deleted.

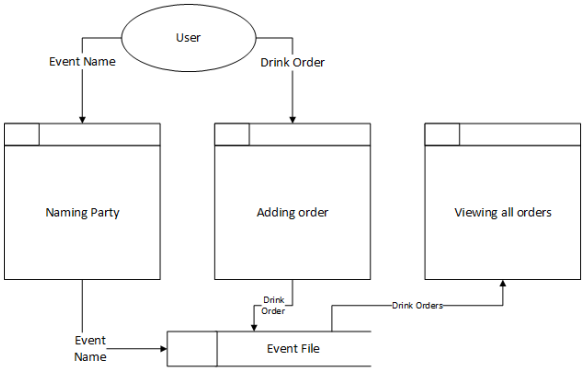

Fig 2.2 Data Flow Diagram

- **3.1 Implementation (Paper Based Design)**

Before any programming and implementation was done, a few short and basic paper based designs were made. As well as this a few notes were taken on implementation ideas. There were a few alternatives and due to the basic detail shown these were adapted in order to fit all functionality in the final app.

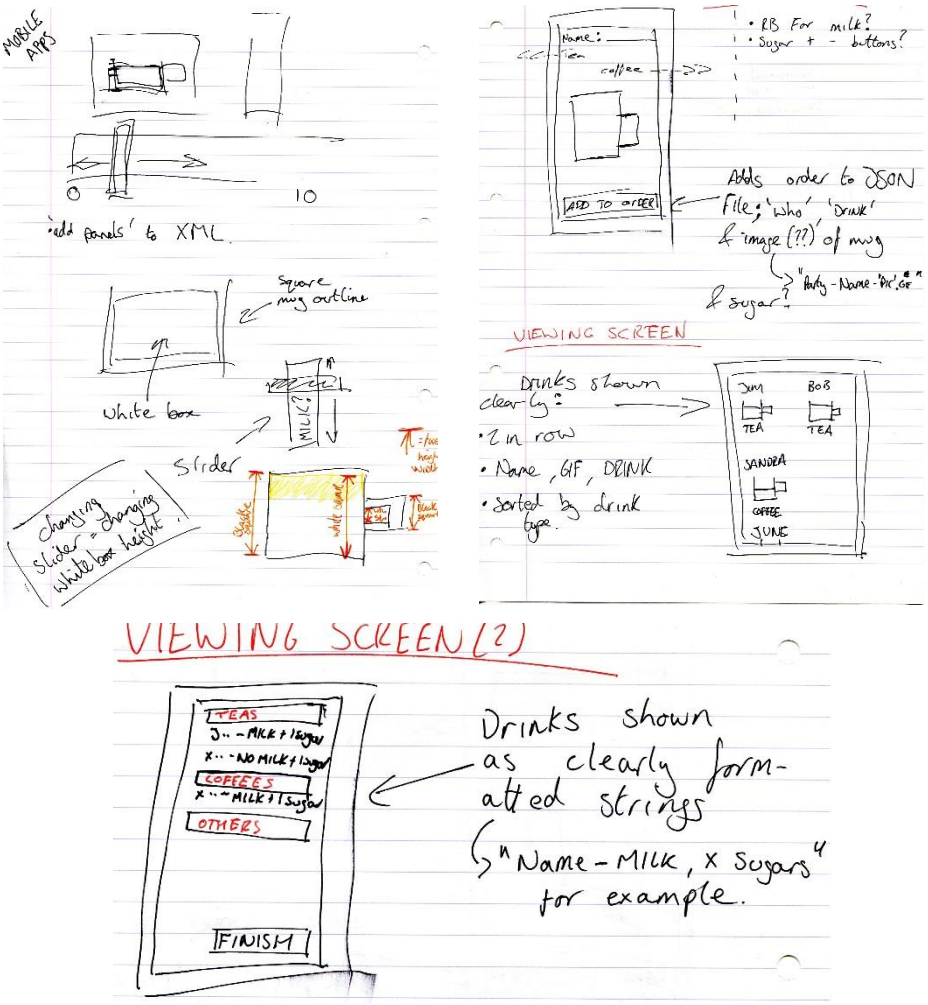

Fig 3.1 Sketches of initial ideas

There were a few major decisions taken at this point that really steered the app in the direction it ended up going in. I decided that I definitely wanted some sort of user feedback based on their milk level. The first way contemplated was using a two image system (one black mug, one white mug) and changing their overlay amount, the more milk the more white was shown. However this seemed like an unnecessary use of data and a complex method that may not even have the desired look. In the end it was decided that using a canvas and the paint method would be simpler and also have a more retro but obvious look as to what it represents.

Another major decision chosen here was how to display the data in the end. As shown in Fig 3.1 I had two different methods in mind. One of them would use a squared layout and display the mug graphic again but individualised based on the person's order. The other would be the orders split into drink type using either an expandable list view or a table. In the end I chose the latter as it was clearer and, if formatted correctly, easier to read.

- **3.2 Implementation (Working App)**

The app required three activities, a panel and a class. The order class holds the name, drink, amount of milk and number of sugars. The panel would display a mug and depending on the user choice the amount of milk required. This was done by passing through a float value based upon the user's choice of radio buttons. There was a thought as to doing this based on the user touching where

there level would be but this was deemed too complex and radio buttons seemed to be the best alternative. The mug was drawn using multiple squares, two black for the main sections. The handle was a square of the background colour and the milk a white square inserted into the main body of the mug.

Fig 3.2 Mug Panel

The first activity that the user comes to asks them to enter their event name. This input will be used as the file name on the SD card of the user. This was purposefully kept simple and the user cannot continue to the next screen until they have inputted an event name. This means that there will never be a situation of a null file name.

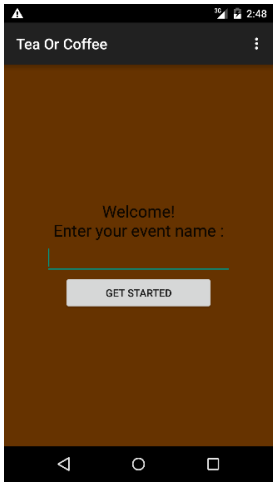

Fig 3.3 Initial Screen

The user is then lead onto the ordering screen. This is clearly laid out and has a simple approach meaning that the user can work their way from the top of the screen down and all the elements get completed. Once the user clicks next person a toast is shown to say that the drink has been saved. This provides the user with feedback, something I deemed very important and useful.

Fig 3.4 Order Screen

Once all the orders have been added and the finished button clicked the user is sent to an activity which shows the orders. The orders get processed, split into 'Teas', 'Coffees' or others, and then displayed in a table. The order has been split up and re-formatted as a string. As stated in the previous section this was decided as the clearest way of presenting the information. Initially I attempted to use an expandable list view, but decided upon a table view instead. This was due to its simplicity and also the speed at which it could generate itself.

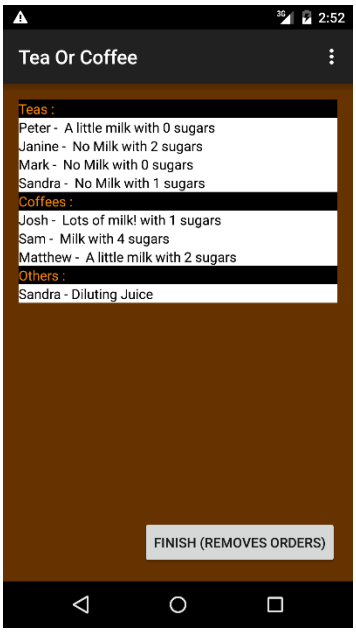

Fig 3.5 Order Viewer

The order is processed into "Name – Milk Statement, with X sugars". This is clear to see and understand and if it is an 'other' drink, "Name – drink". The user can go back to orders via the built in button on android. The user can also click on a finish button which will delete the file, saving space on the user's device, and return the user back to the initial screen. This also creates a pop up ensuring the user is entirely sure they want to delete the order.

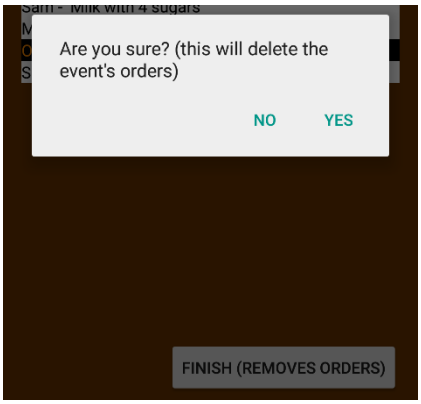

Fig 3.6 Deletion of Event

- **4.1 Comparison to Original 'Scope and Content'**

Through the use of pre-planned software design and paper based designs it was relatively easy to maintain the original scope of the app. The user can name the event, input their choices of drinks and the host can clearly see the orders. The milk level is shown graphically to the user and so all the

requirements set out at the start have been met. The process has been made easier and so the ultimate goal has been achieved.

- **4.2 Comparison to IOS alternative**

When comparing this to the IOS alternative, mentioned in section 1, I feel this displays the orders in a much better and user-friendly manner. On the IOS app it shows the orders by person not by drink. I feel that sorting them by drink means the user can see at a glimpse how many of each drink to serve. The app I designed also shows a much clearer order as to sugar and milk and so avoids confusion. Both apps represent their platforms very well on a visual front, however the IOS app is starting to look dated. This may deter users from using it as it seems not to be as slick a design and not as welcoming.

- **4.3 Real World Evaluation**

In addition to the testing against the specification and other applications, above, I performed real world evaluation. This app has only been simulated on a virtual android device. However it has been used by real world users. Whilst not being scientific, the feedback received was mostly positive. The use of '+' and '–' buttons for choosing sugar has been spotted as a clear and simple way of selecting. One of the issues that has arose is that a user will click on the 'finished' button to input their order not the 'next person' button. This issue has been addressed by the app saving the order if either button is pressed. However the app will search through the order details and not save it if no name has been entered. This means that when the 'Finished' button is eventually properly clicked it doesn't show a blank order.

- **4.4 The Future**

Future developments for this app could be using a wireless network to add orders. An event could be hosted on the hosts tablet/phone and the individuals could add their orders from their own personal app. This could even lead to you saving your personal preference and being able to send your order the instant you are in reach of the network. Another smaller development would be to implement the altering of the milk level based on the user touching where their desired level would be. However these are only theoretical and beyond the scope of this coursework.

- **5.1 Resources/References List**

Tea or Coffee (IOS App)	https://itunes.apple.com/gb/app/tea-or-coffee/id581844386?mt=8
Git Repository	https://github.com/JR1110/T-or-C.git
Stack Overflow (Multiple pages)	http://stackoverflow.com/
Android Documentation	http://developer.android.com/guide/index.html
Expandable List View Tutorial (Unused)	http://www.androidhive.info/2013/07/android-expandable-list-view-tutorial/
Module Website	http://siwells.github.io/teaching_set08114/