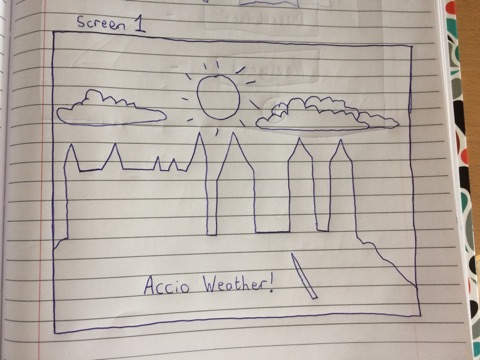
**App Design**

I decided to do a Harry Potter themed (just to make it a bit unique/interesting) **weather application** as my app as it provides most of the functionality and requirements in the assignment specification.

The app will contain six different screens:

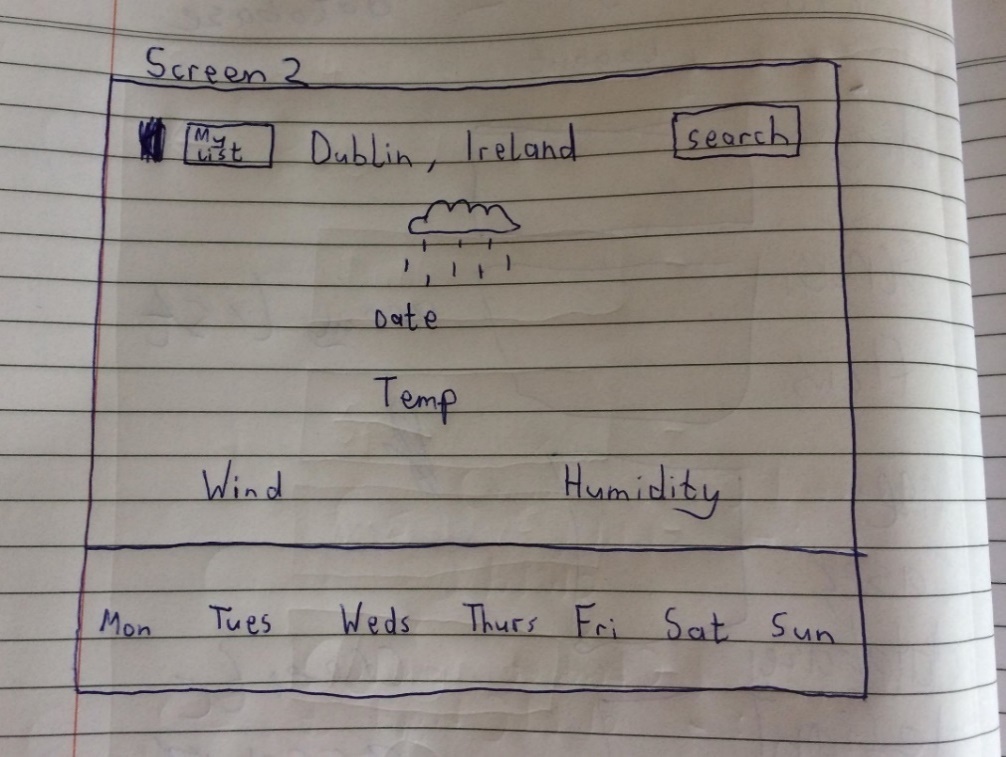
1. Intro Screen - This screen will appear first when the user first clicks on the app. It will simply display the title of the app along with the icon used for it. This screen will shortly be replaced by the Main Screen of the app.
2. Main Screen - Weather activity in the users current location (at the current time).
3. Search Screen – User can search for any location which will display its current weather. This will also show the weather for every day of the upcoming week. User will have option to add this location to a list.
4. List Screen – Displays all locations that the user has put in the list for easier and quicker accessibility.
5. Add Screen – The result from the Search Screen, containing an Add button to add the location to the list.
6. Element Screen – When viewing an element/location of the list.

**Intro Screen**



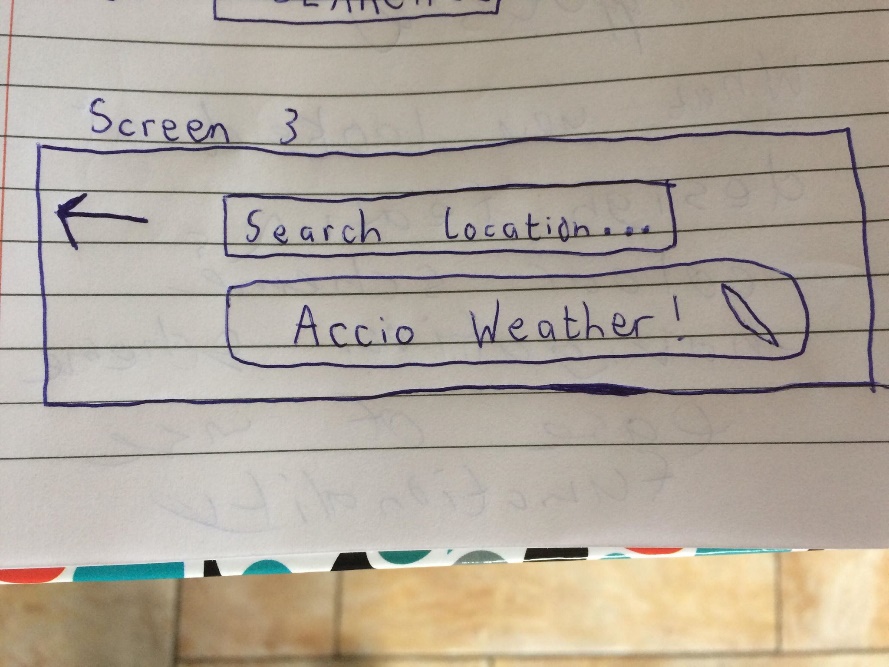
The screen that will appear as an opening before it preceded by the Main Screen. Contains an image of a castle and sunny weather behind it, along with the name of the app.

**Main Screen**



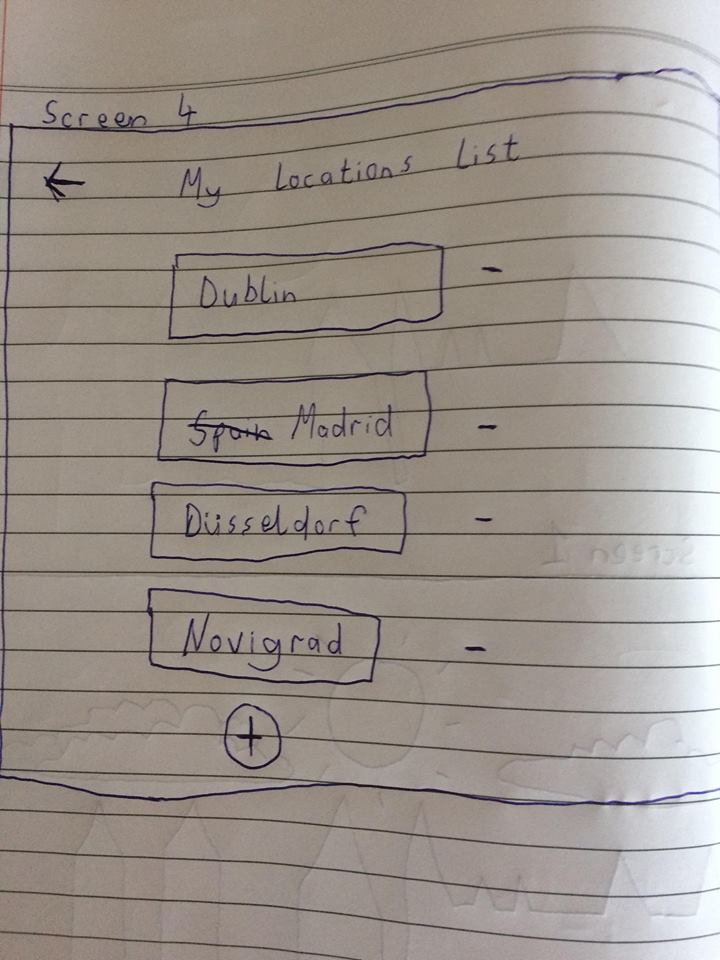
This is like the home page of the app. From here, the user can navigate to the Search Screen and List Screen. As a default, the app will show the current conditions based on the users location, along with minimum information on the upcoming week in the list located at the bottom of the screen.

**Search Screen**



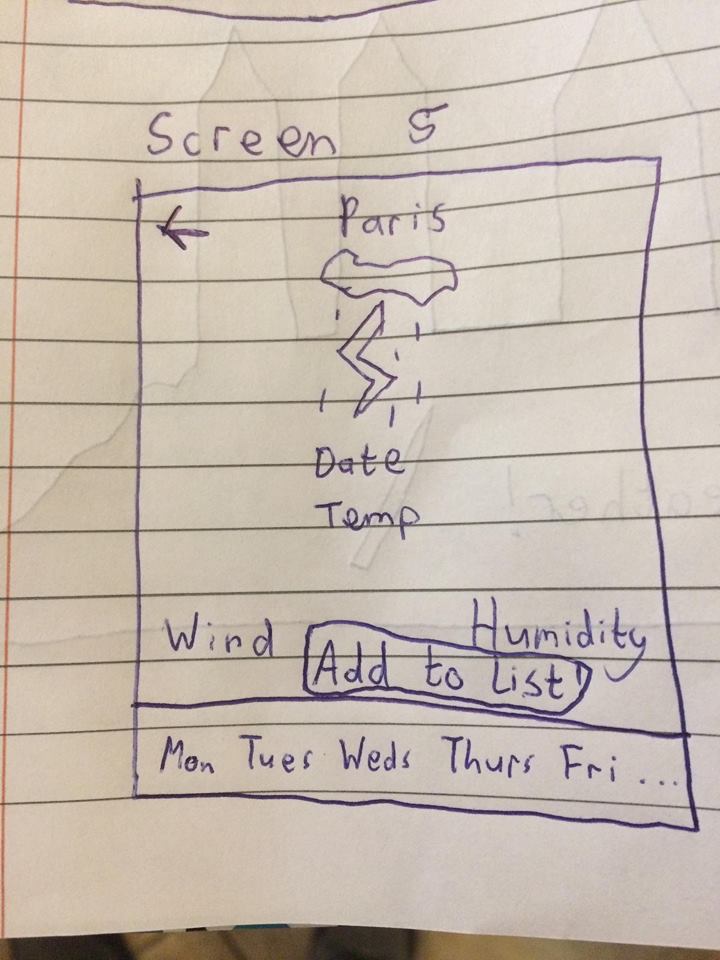
The arrow on the left indicates that the user can return to the Main Screen. Here the user inputs a location in a textbox and clicks the button beneath it to search for any locations.

**List Screen**



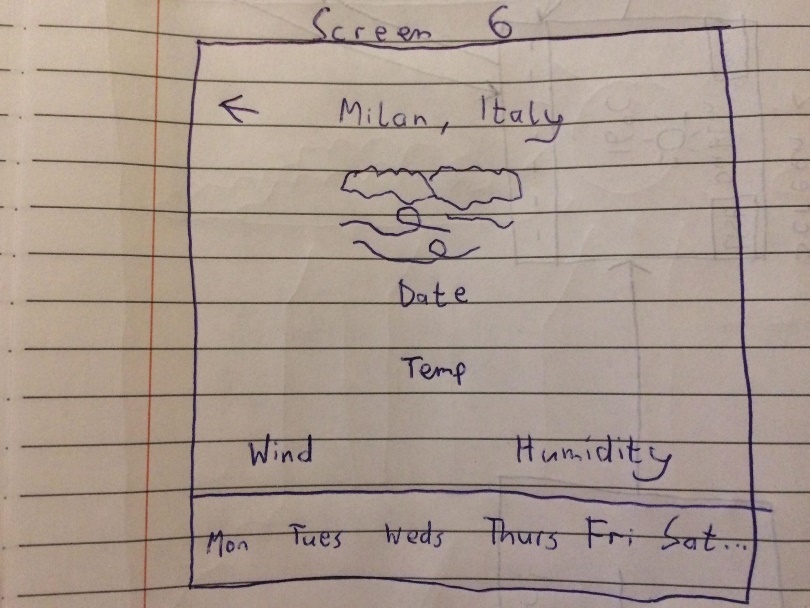
Once again, the arrow on the left indicates that the user can return to the Main Screen. The users list of locations are kept in a list of clickable buttons so that they can access further information regarding the weather depending on the button clicked. To the right of each button will be a red minus sign, where the user has an option to delete a location from the list. The plus sign at the bottom of the list allows the user to add another location which will bring the user to the Search Screen. When the user searches a location, they will have the option to add it to their list.

**Add Screen**



Similar to the Main Screen. However, it does not contain a search button or a button to view the list. It contains an arrow (upper left hand corner) to navigate back to the Search Screen. It also contains a button “Add to List” near the bottom of the page to add the location into the list. This location can then be viewed on the List Screen.

**Element Screen**



Just like the Main Screen and Add Screen, it provides all the weather information of the location. Unlike the Main Screen and Add Screen, its only additional button is an arrow (upper left hand corner) to return to the List Screen.

**An app design review: Weather app on my iPhone 5**

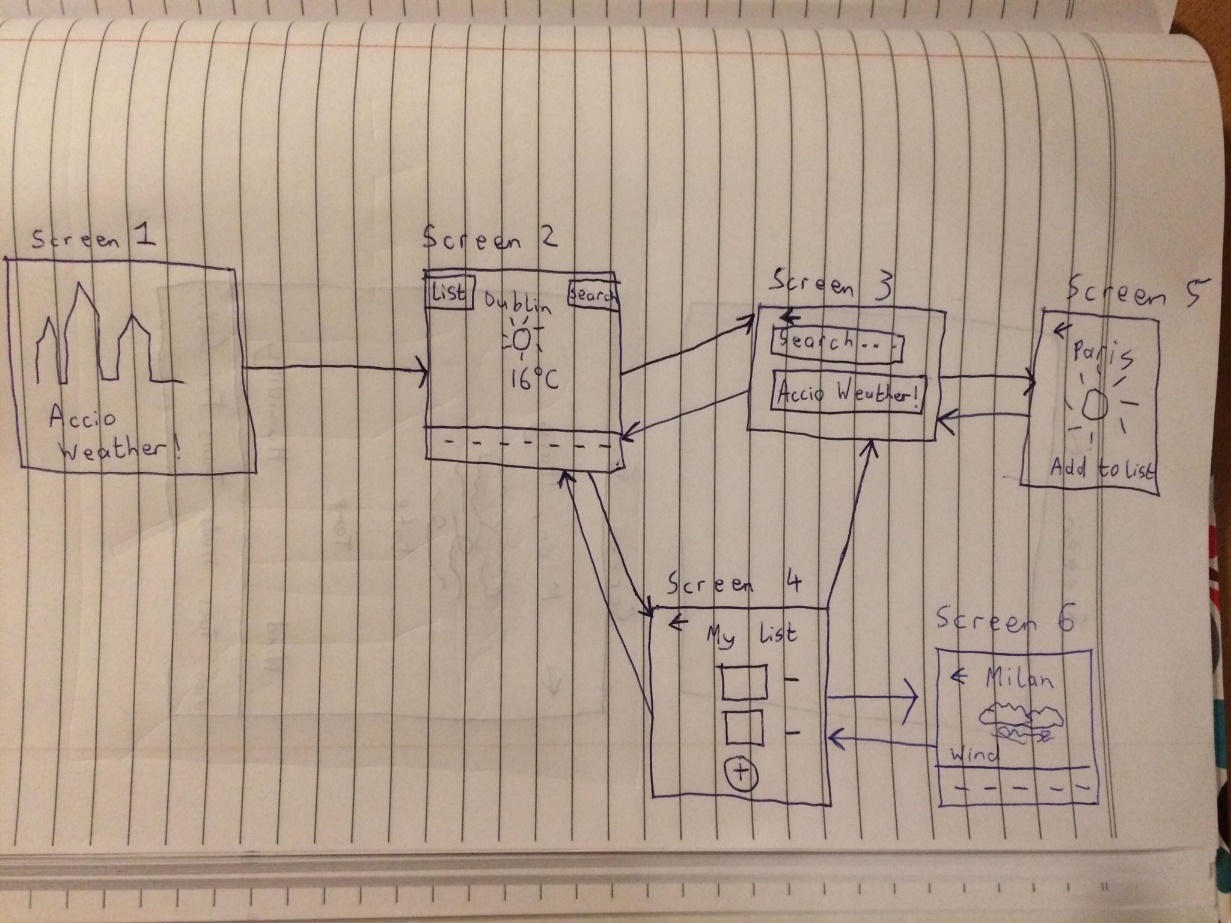


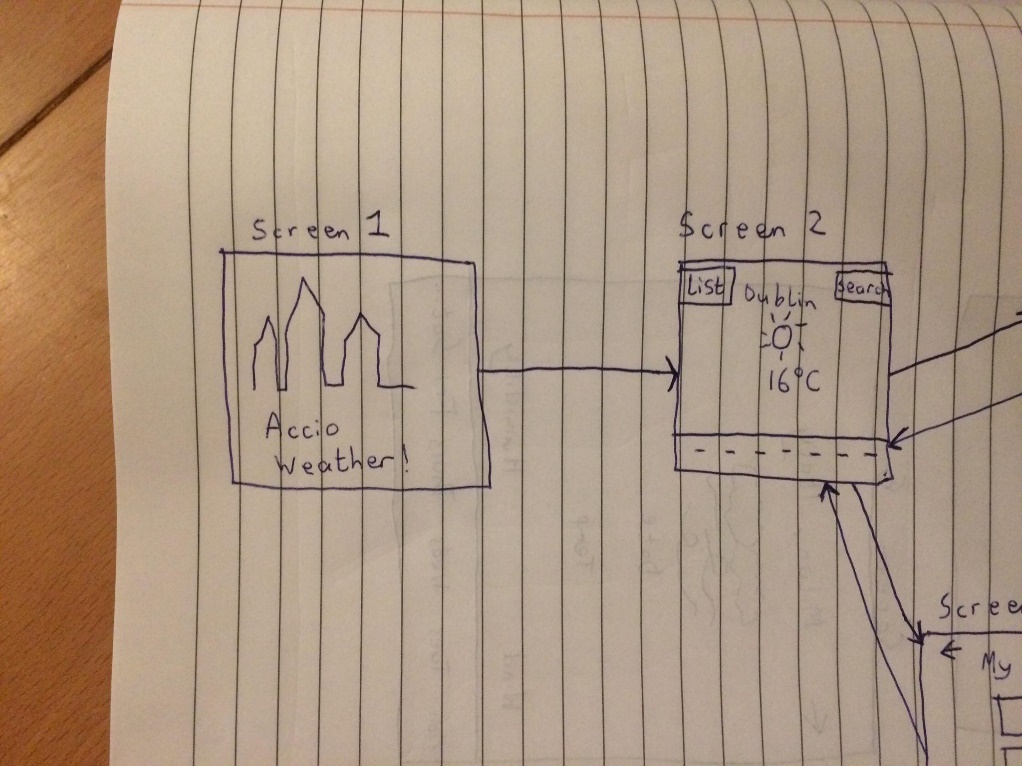
I think this app is standard. It could be improved. For example, in the picture above, the user can make a list. However, there is no way for them to delete an element from that list. Also, navigation is incoherent between pages. There is no arrow or button to return to a screen or main screen. There is no individual screen to search for the weather (unless you are adding a location to the list). The only way to search a location’s weather is if you add a location to your list and there is no option to delete!



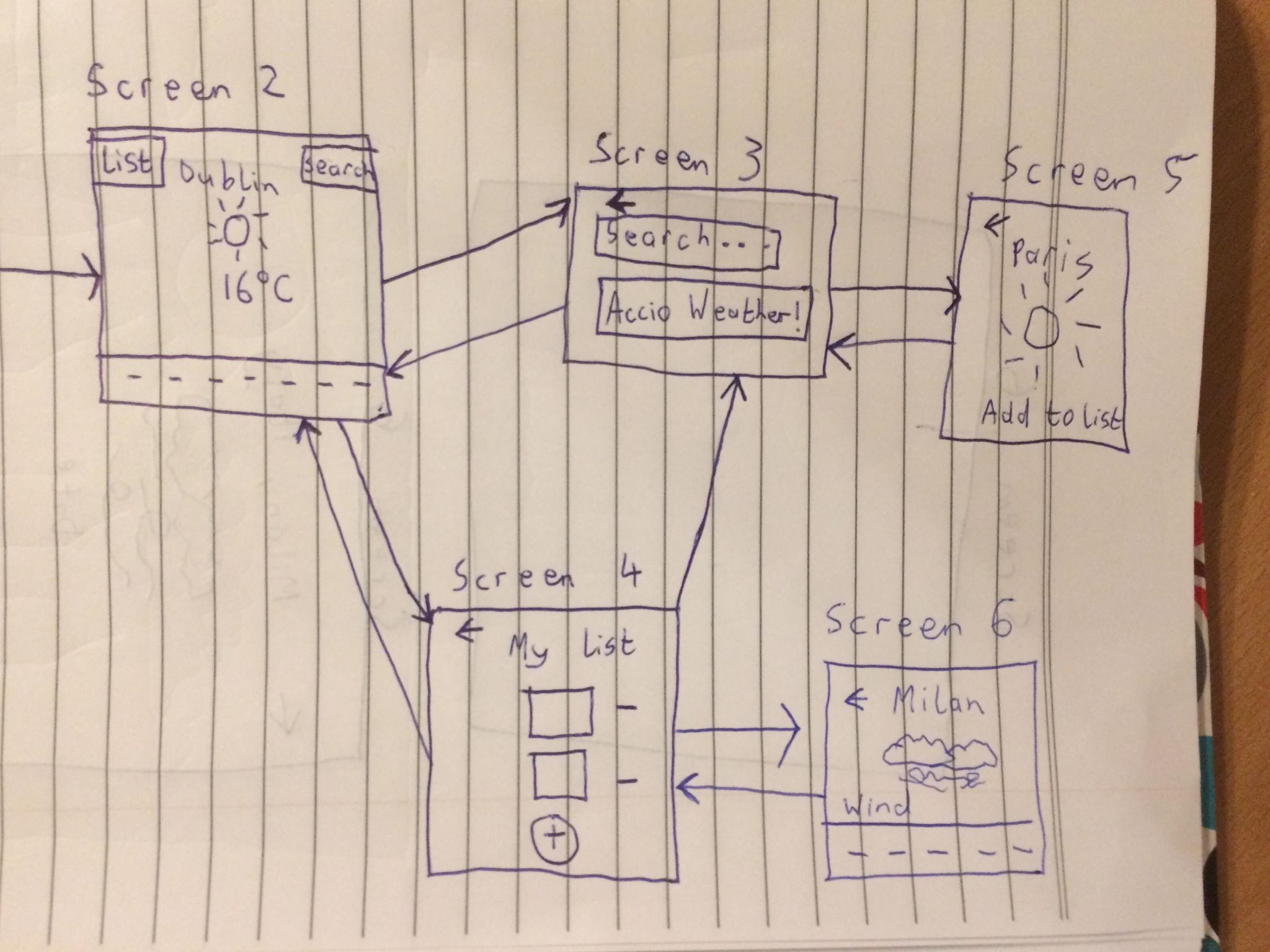
The styling is quite well done. When a location is clicked, it provides all necessary information one could possibly need with regards to the weather. This information is laid out in a very clear and coherent way and the background moving images of the weather add to the clear and simplistic layout when reading about a locations weather.

**Screen Flow**



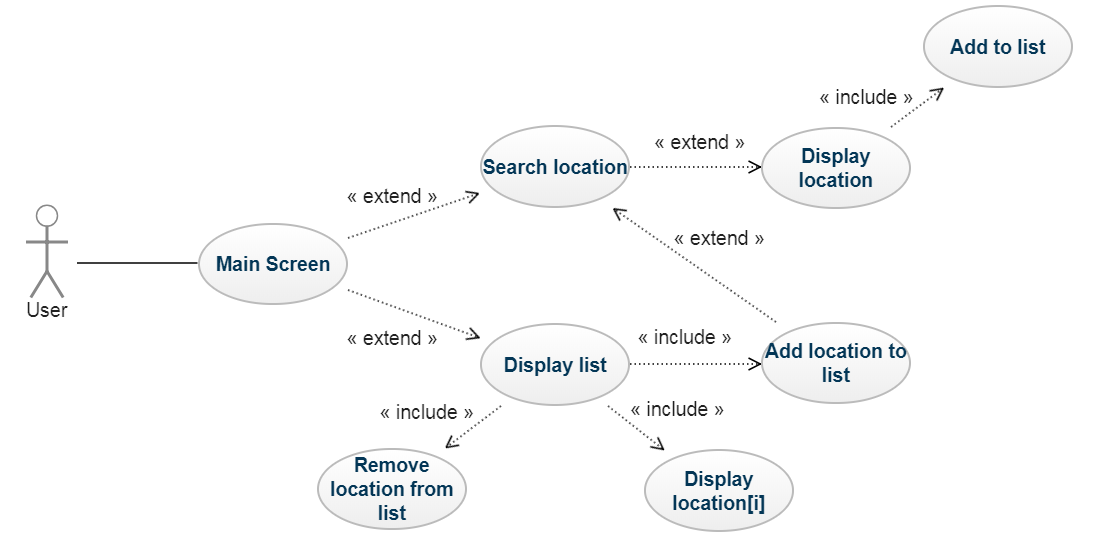


Screen 1 (Intro Screen) is connected to Screen 2 (Main Screen). It is not possible to return to Screen1 one from Screen 2 while using the app. Screen 1 will only be accessible again if the app is shut down.



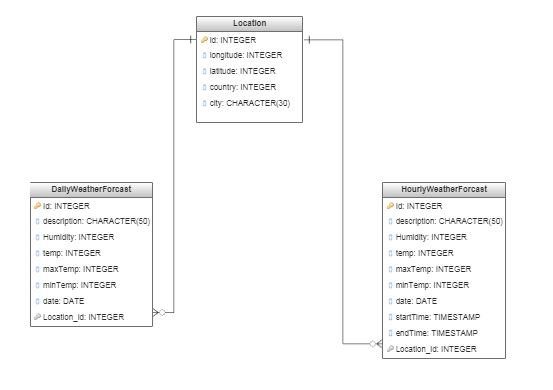
Screen 2 (Main Screen) provides access to Screen 3 (Search Screen) by pressing the search button in the upper right hand corner. It also provides access to Screen 4 (List Screen) by pressing the button in the upper left hand corner. Screen 3 can navigate back to Screen 2 with the arrow button in the upper left hand corner. Screen 4 can navigate back to Screen 2 in this manner too. When a location is searched for in Screen 3, Screen 5 (Add Screen) is prompted, showing the weather information for that location. The user can return to Screen 3 by clicking the arrow. Screen 5 can also add the location to the list in screen 4 by clicking a button located near the bottom of the screen saying “Add to List”. When viewing the list in Screen 4, the user can add a location (by pressing the plus button at the bottom of the screen) which will relocate them to Screen 3. Screen 4 can click on a location within the list to view all the details of the weather. This will bring then to Screen 6 (Element Screen). This simply displays all weather details regarding that location/element in the list. The user can navigate back to their list by pressing the arrow in the upper right hand corner of Screen 6.

**Use-Case Design**



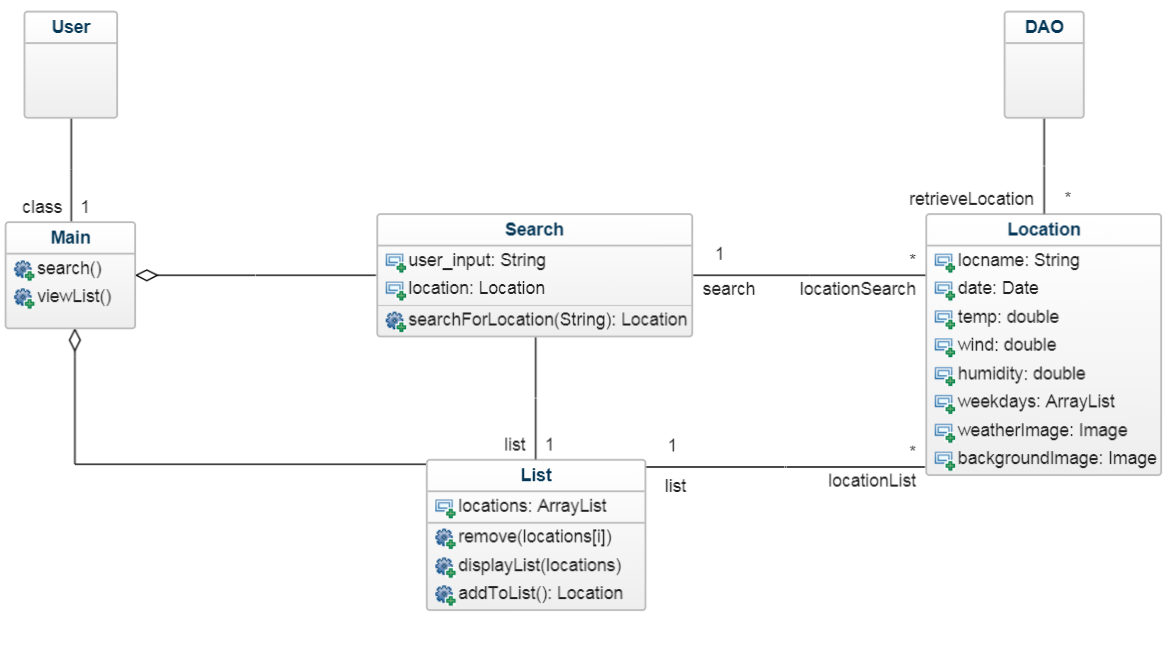
The user directly interacts with the Main Screen. Here the user can search for a location or display the list of locations. When searching for a location, the resulting location will be displayed and the user has the option of adding the location to their list. When displaying the list, the user can optionally remove a location from the list, display a location from the list or add a location to the list. If the user decides to add a location to the list, they will be navigated to the Search Screen to search for a location, and the location will be displayed and the user has the option of adding the location to the list again.

**Database Design**



When the user first opens the app and has not picked a location yet, then they will have to enter in the necessary details to search for a location. If there is a location already selected, a request to the API is sent for the hourly and daily weather forecast for the next seven days and stored in the database. So in the database it should have (7 \* 24) entries every hour for the next seven days in the HourlyWeatherForcast table. The DailyWeatherForcast table will have seven entries (one row for the next seven days).

**Class Diagram**



Much like the use case diagram, the user directly interacts with the main screen (the Main class). It has two functions: search() and viewList(). The associations with the Search class and the List class mean that the Main class ‘has that class’. In other words, Search and List are contained in the Main class. Search takes some user input as a string datatype and searches all the Location types to see if there is a match. A Location will be returned if there is a match. The association between the Search class and the Location class mean that a Search can have many different Locations and a Location can only belong to that individual Search instance. The Location class contains all the details with the regarding the weather. This information will be retrieved from the database via the DAO (Database Access Object). Similarly, the List and Locations classes have a one-to-many association. A list can have many locations, however, the location only belongs to that single list. The locations will most likely be some sort of ArrayList or Array so the functions will loop through the list of locations when removing an element or displaying the elements. Adding a location to the list will return a Location instance and add it to the list.