**Vision College Logo**

# Diploma in Software Engineering and Design

# Assignment Cover Sheet

|  |  |
| --- | --- |
| Course name: Diploma in Software Engineering and Design | **Student’s name:** |
| **Assignment title and/or number**: Mobile app development using a local database | |
| **Assessment weighting** | *15 credits,*  *12.5% of the overall programme.* |
| **Due date**: 1st April 2016 | **Date submitted**:  (late submissions incur 10% penalty, after 7 days late, the assessment will not be marked) |
| **Assessment conditions:** | This is a resource-based assessment. This means that you may have access to any relevant resources to assist you. This could include, for example, your learning materials, information on the Internet, and so on. However, all work must be your own with no assistance from any other person. |
| **Submission requirements:** | You’re required to submit the following into your assignment submission directory:   * This document, completed where appropriate * SQLite Database file * Visual Studio or Xamarin project files * Testing Sheet attached   Also upload your assessment on Github and share your repository link |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Assignment Checklist:** | |  |  | | --- | --- | | **Requirement** | **Completed** | | Database | ✓🗶 | | User interface | ✓🗶 | | Functionality | ✓🗶 | | Coding | ✓🗶 | | Testing | ✓🗶 | |

# Disclaimer of Plagiarism and Collusion

I declare that, to the best of my knowledge, this assessment is my own work, and has not been copied from any other student's work or from any other source.

Enter your name here to indicate you agree to the above statement.

|  |
| --- |
|  |

**Requirement**

This assessment consists of building an app that displays today’s weather and the forecast for the important cities of New Zealand (Auckland, Christchurch, Hamilton and Wellington)

**API**

**Open Weather Map API**

You can use the open weather map API to obtain the current weather information.

For example to find the weather for London city you can give a request as follows.

[http://api.openweathermap.org/data/2.5/weather?q=London&mode=xml](http://api.openweathermap.org/data/2.5/weather?q=London&mode=xml%20)

Details of the API can found at the link below

<http://openweathermap.org/API>

**Wunderground API**

There is another API by wunderground that gives the same result.

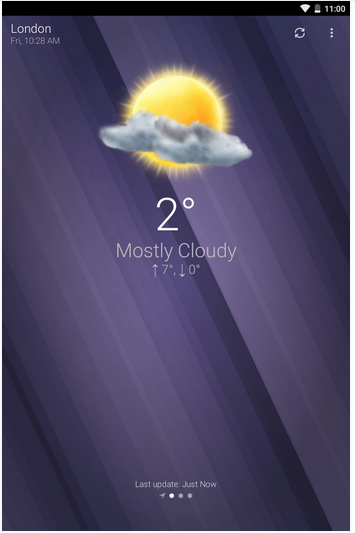
Details of the API can found at the link below

<http://www.wunderground.com/weather/api/>

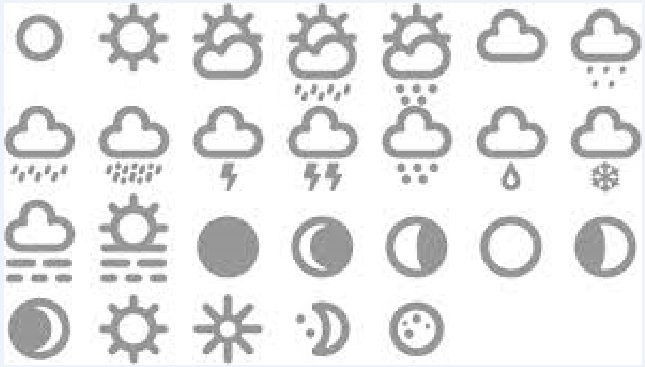
Feel free to search and use any other API that may be better than the ones mentioned above.

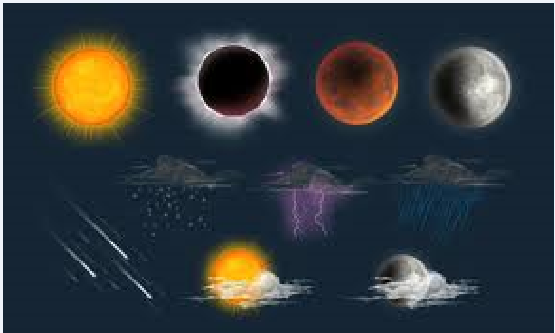
**Some Sample App Layouts**

Some sample layout to get you started.

The image bitmaps for the weather conditions can be obtained on the Internet





**Basic Requirements**

1. Show the current temperature and weather conditions (sunny, rainy, partly cloudy etc.)
2. Show the current date and time
3. Have a selection for the city, and the weather condition changes based on the city selection
4. Show a bitmap for various weather conditions
5. Show 5 day weather forecast
6. Design the app to fit different screen layouts
7. Automatically detect the current city based on the GPS location [**Optional**]
8. Anything fancy you can think of

**Marking**

|  |  |
| --- | --- |
| **Marking Schedule** |  |
| Program Functionality | **60** |
| App design and UI look | **15** |
| User Friendliness | **10** |
| Creativity | **15** |

**Testing Sheet**

Kindly fill in the testing sheet below with your comments. These are the common quality checks you would perform before uploading your app on the Google play store.

Some common test procedures for an android app can also be found on the Google developer website.

<http://developer.android.com/distribute/googleplay/quality/core.html>

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test Condition** | **Result (Pass/Fail)** | **Comments** |
| **App Design** | Navigate to all parts of the app — all screens, dialogs, settings, and all user flows.  **Platforms to Test**  Android 4.4  Android 5.0  Android 5.1 | Pass |  |
| Navigate to all parts of the app — all screens, dialogs, settings, and all user flows  **Screen sizes to Test**  4’ inch  5’ inch  7’ inch  10’ inch   1. App displays graphics, text, images, and other UI elements without noticeable distortion, blurring, or pixilation. 2. App provides high-quality graphics for all targeted screen sizes and form factors, including for [larger-screen devices such as tablets](http://developer.android.com/distribute/essentials/quality/tablet.html). 3. No aliasing at the edges of menus, buttons, and other UI elements is visible. 4. Composition is acceptable in all supported form factors, including for larger-screen devices such as tablets. 5. No cut-off letters or words are visible. 6. No improper word wraps within buttons or icons are visible. 7. Sufficient spacing between text and surrounding elements. | pass | The sizes of the Data objects don’t change so when you go for a bigger screen size the app will look smaller |
| **App State**  App preserves user or app state when leaving the foreground and prevents accidental data loss due to back-navigation and other state changes. When returning to the foreground, the app must restore the preserved state and any significant stateful transaction that was pending, such as changes to editable fields, game progress, menus, videos, and other sections of the app or game. | From each app screen, press the device's Home key, then re-launch the app from the All Apps screen.  When the app is re-launched from Home or All Apps, the app restores the app state as closely as possible to the previous state | pass |  |
| From each app screen, switch to another running app and then return to the app under test using the Recent’s app switcher.  When the app is resumed from the Recent’s app switcher, the app returns the user to the exact state in which it was last used. | Pass | The app maintains its state even when another app has been opened |
| From each app screen (and dialogs), press the Back button.  On Back key presses, the app gives the user the option of saving any app or user state that would otherwise be lost on back-navigation. | Fail | There is no prompt as the app functions in a single state so a save is rendered redundant |
| From each app screen, rotate the device between landscape and portrait orientation.  The App should correctly preserve and restore user or app state. | fail | The app will change but there will be cut off areas such as the five day forecast |
| **Stability and Performance** | App does not crash, force close, freeze, or otherwise function abnormally on any targeted device.  App loads quickly or provides onscreen feedback to the user (a progress indicator or similar cue) if the app takes longer than two seconds to load. | pass | My app functions correctly |