SWEN325 – Software Development for Mobile Platforms

SWEN325 – Assignment 1

# Contents

[Contents 2](#_Toc48301328)

[Introduction and Task Description 3](#_Toc48301329)

[Core: Pre-processing of COVID-19 cases in a given area [40 marks] 4](#_Toc48301330)

[Select a dataset and analyse (10 marks) 4](#_Toc48301331)

[Describe the results of each technique used on the one dataset (10 marks). 4](#_Toc48301332)

[Pipeline used for Classification with Naïve Bayes 4](#_Toc48301333)

[Pipeline used for Clustering with Simple K Means 6](#_Toc48301334)

[Pipeline used for Linear Regression 7](#_Toc48301335)

[Identify how these aspects of the techniques are different (10 Marks) 10](#_Toc48301336)

[Revisited Business understanding (10 marks) 10](#_Toc48301337)

[Part 2 Completion: Feature Importance to COVID19 cases [40 marks] 10](#_Toc48301338)

[Business Understanding, question to be answered and data sources (10 Marks) 10](#_Toc48301339)

[Merge the Data sources using WEKA (10 marks) 10](#_Toc48301340)

[Use Dimensionality Reduction (10 Marks) 12](#_Toc48301341)

[Analyse the output of the data (10 Marks) 15](#_Toc48301342)

[Challenge: Visualisation of results [20 marks] 16](#_Toc48301343)

[Visualise the most interesting results (10 marks) 16](#_Toc48301344)

[Consider the consequences and ethics of reporting your findings (10 marks) 16](#_Toc48301345)

[Possible consequences 16](#_Toc48301346)

[Background - supplemental 17](#_Toc48301347)

[Features in the original files 17](#_Toc48301348)

[Using suitable techniques Merge the files together (10 Marks) 17](#_Toc48301349)

[Features in the combined file - output-data.csv 18](#_Toc48301350)

[Features of secondary combined file output-hospital-data.csv 19](#_Toc48301351)

[Dimensionality Reduction 19](#_Toc48301352)

[CRISP-DM Steps 21](#_Toc48301353)

[CODE 25](#_Toc48301354)

[Main-countyHealthSummary.ps1 25](#_Toc48301355)

[Main-hospital.ps1 26](#_Toc48301356)

[Get-canary 28](#_Toc48301357)

[Get-FutureStats 28](#_Toc48301358)

[Get-MaximumIncreaseInCasesOverDays 28](#_Toc48301359)

# Introduction and Task Description

## Written Report (To Be Done Individually) (Worth 60%)

The reports are assessed individually and therefore if you are in a group of two or three people, you will need to write it yourself and present a different usability test plan from your other group members.

### What to submit.

Submit a file called 325-a1-report-username.pdf that contains:

* Description of the overall architecture of your application. 2 Pages of A4, Worth 10% out of 60%
* Include a description of how you organised your source code to match your architecture.
* Description of how you utilised at least 1 major existing external component (e.g. Firebase but hopefully much more interesting one than that) to provide either persistent data storage or other major functionality with a description of how it was integrated into your App architecture. 1+ Page of A4, Worth 10% out of 60%
* Reflective report on Ionic as a framework for App design including its advantages and disadvantages. 2 Pages of A4, Worth 10% out of 60%
* Usability test plan 3+ Pages of A4, Worth 30% out of 60% that includes:
  + Goals for the usability test
  + Format of the usability test
  + Target users (and how many)
  + Tasks that users have to perform with the App (including screen shots)
* Finally, include an appendix with as many pages as there are separate screens in your app, with each page containing a screenshot of the screen and a one paragraph description of its design.

# Application Architecture – TO DO (2 pages)

## Source code organisation

src

│ App.tsx

│ index.tsx

│

├───components

│ │ Default.css

│ │ ExploreContainer.css

│ │ ExploreContainer.tsx

│ │

│ ├───assetListItem

│ │ AssetListItem.tsx

│ │

│ ├───currentSession

│ │ CurrentSessionContainer.tsx

│ │

│ ├───home

│ │ HomeContainer.tsx

│ │ IntroductionContainer.tsx

│ │

│ ├───map

│ │ map.css

│ │ Map.jsx

│ │

│ ├───menu

│ │ Menu.css

│ │ Menu.tsx

│ │

│ ├───newAsset

│ │ NewAssetComponent.tsx

│ │ NewAssetContainer.tsx

│ │

│ ├───previousSessions

│ │ PreviousSessionsContainer.tsx

│ │

│ ├───settings

│ │ SettingsContainer.tsx

│ │

│ └───upload

│ UploadContainer.tsx

│ UploadInformationContainer.tsx

│

├───models

| AssetsContext.tsx

│ ServerConverter.js

│ SettingsContext.tsx

│

├───pages

│ Page.css

│ Page.tsx

│

└───secrets

googleMaps.js

### UI Components

The general application architecture splits the application into 6 major sections, four on the device and two located in the cloud. The code on the phone is separated into UI and business/data.

The UI is built from the app holding a single Page element, and within that page element containers are turned on and off as the user navigates the UI.

All of the containers sit within the ‘Components’ folder within their own named folder, this helps to keep the files organised in what might otherwise be a large list of files without context. Some of the elements are reused and so have been brought to the top level e.g. assetListItem, which is used in both the “Current Session” and “Previous Session” containers.

Other elements are only used in a single component and so are kept in the same folder, such as the SerialNumberPhoto and AssetIdPhoto elements. This helps to avoid a cluttered folder structure where all components have their own folders. Creating a separate element is only done if it is significantly complex – so things like Toasts are kept within the main component.

### Data

#### Models

Data connections and models are kept in the same files, with the specific needs of each of the data types kept with the data type itself. In this way state, local storage and upload to the cloud are all kept with the type Asset and Assets.

Settings are the second data type and have their own Context. Settings are saved locally and use state as well to improve performance and ensure a useful experience in the case of an app being closed/reopened

#### Secrets

Secrets are a special type of data that includes API keys and data that needs to be kept from replication. In an ideal situation this would be kept as part of the environment variables and so not included with source code.

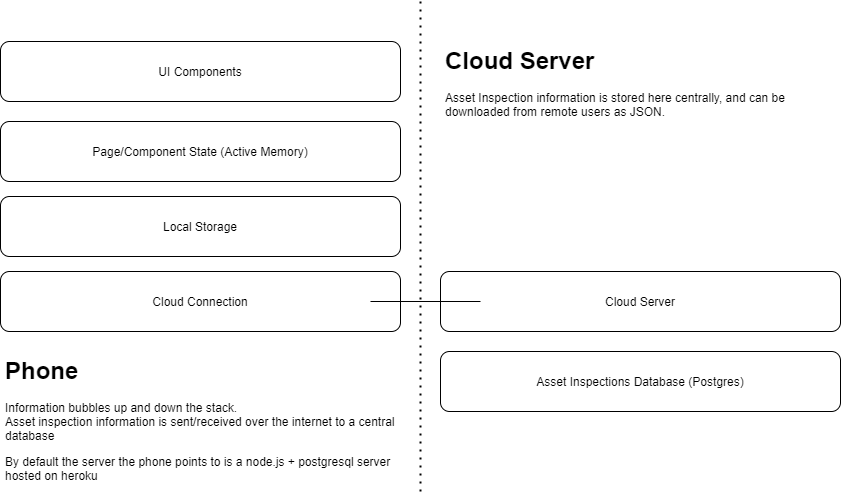


Figure 1: Data flow within the application

# Major External Components – TO DO (1 page)

For External Component, you **need to** describe how your way of implementing it fits into your proposed architecture and provide technical details of how it works and **justify** your choices and why it is the best way to fit the component into your app architecture and design.

Description of how you utilised at least 1 major existing external component (e.g. Firebase but hopefully much more interesting one than that) to provide either persistent data storage or other major functionality with a description of how it was integrated into your App architecture. 1+ Page of A4, Worth 10% out of 60%

## Map

The map is a useful visualisation of the GPS coordinates which are pulled from the device. Without this visualisation identifying a problem would be difficult and this improves the experience as the user inputs data.

## Camera

One of the major benefits of using a phone to take this type of information is that a camera can be used to capture information and make manual processing easier later on.

The camera is only used within one of the screens (New Asset), it includes taking the photo and allowing the user to review the photo when inputting the manual data later on.

## Remote Server (using attached postgresql server)

The remote server is a node.js server with an attached postgresql server attached, allowing for asset inspections to be saved and reviewed later. As the aim for this application is to act as a handset for gathering information the central and remote server is a very important part of the architecture.

In many cases companies performing asset inspections will have existing lists of assets, so being able to export the asset information out in an easy to use format is important. This is done with a JSON to response to GET requests to the /assets URL.

# Framework Reflection (2 pages)

For Framework Reflection, you are expected to provide a lot of technical details, examples, and justifications for how the Ionic or React Native helped or hindered your development. It is essential to not just describe how it works and whether you liked it but rather justify and show technical examples of how it affected your application design and what further improvements or insight you could have hypothetically provided to framework developers.

Reflective report on Ionic as a framework for App design including its advantages and disadvantages. 2 Pages of A4, Worth 10% out of 60%

# Usability Test Plan (3 pages)

## Application flow diagram

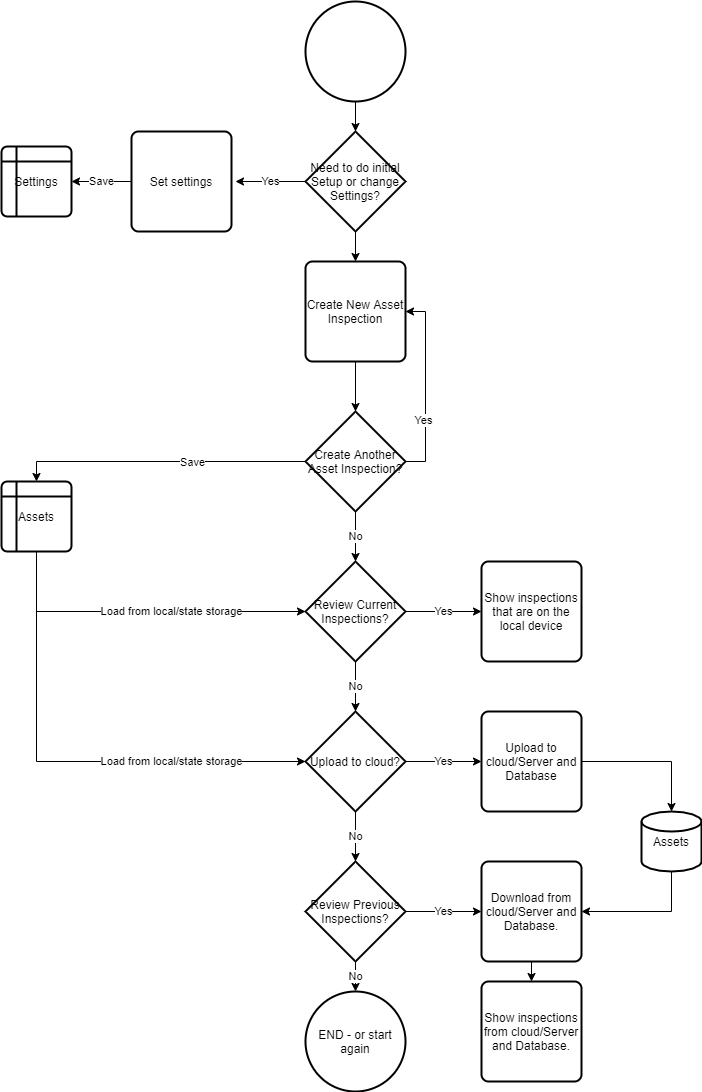


Figure : Application flow diagram, showing choices and flow through the application

## Goals for the usability test – NOT COMPLETE

The goal of the usability test is to identify design flaws, to allow for them either to be corrected in their entirety or to be mitigated with other elements.

## Format of the usability test –TO DO

## Target users (and how many) –TO DO

## Tasks that users have to perform with the App (including screen shots) –TO DO

# Appendix with screens

Finally, include an appendix with as many pages as there are separate screens in your app, with each page containing a screenshot of the screen and a one paragraph description of its design.