# MOBILE PROJECT PART B
# DESIGN

SUBMISSION 1

DAN
GRIZLI777

# Contents

# Introduction

This document contains the outline of our application and details of the design we intend to follow. It includes the functional requirements of the app as well as some user interface details.

The document does not contain information about the specific implementation of the functions and features and does not give code examples.

# Domain Modelling and App Idea

Domain modelling is where we build a conceptual model for our design. It helps us to identify important objects and the relationships between them. A lot of the objects defined during this stage will end up becoming classes in our application, though not necessarily all of them.

My app is called VapeRater. It is a simple app which will allow the user to store details about different flavours they have previously purchased so that it is easy to check which flavours scored the highest for re-purchasing. The main page will be a scrollable list containing all entered flavours. Tapping on a flavour will open a new activity with further details and provisions for edits will be made. It will also include a settings screen and a random joke screen, because after all, who doesn't love a random joke?

**Functions:**

- Create a new juice entry
- Delete juice entry
- Edit juice entry
- Persistent data storage
- Adjust screen brightness
- Adjust text size

**Features:**

- Clear layout
- Easy to use
- Compatible with multiple devices

# Specifications

The application is to be built using Android Studio, and it will utilize the Androidx libraries as well as the Okhttp3 and the Gson libraries.

The application will be written to target SDK version 30 which is currently the highest SDK version and runs on Android OS 11. The minimum SDK version is 21 which runs on Android OS 5.0.

The device will be tested inside an emulator using a Pixel 2.0 with a screen resolution of 1920 x 1080, as well as an actual Samsung S6 which has a resolution of 2560 x 1440.

The app will create a local database to store information about juices, and to ensure data persistence.

The app will connect to a server to retrieve jokes when the user opens the joke screen.

The app's navigation pattern will be a simple home screen which is a list, and tapping an item will move to a different screen displaying details and allowing edits. There is a separate navigation from the toolbar to get to the settings and the random joke.

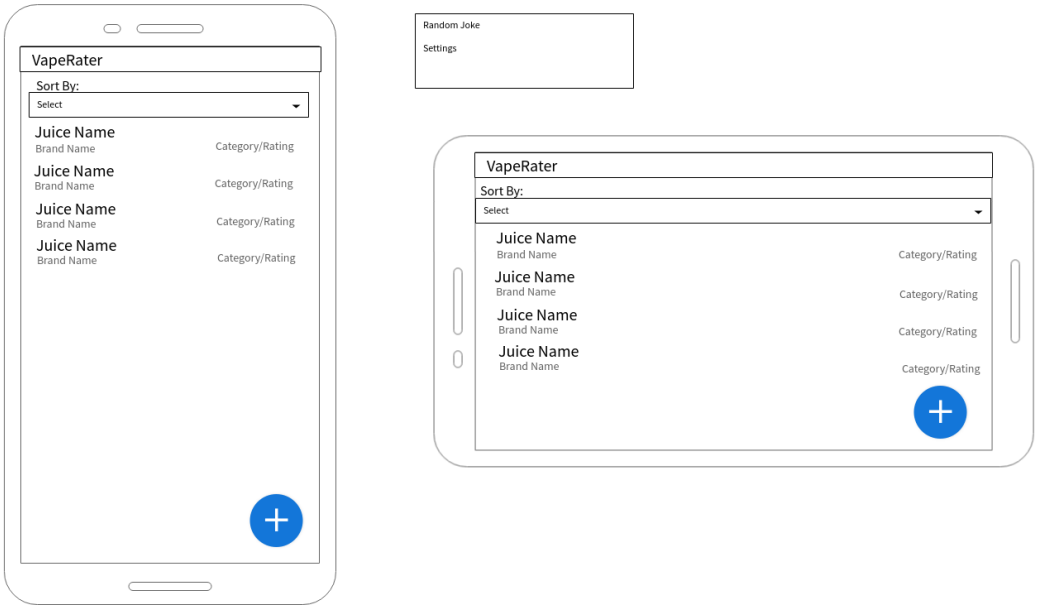
## Design Images

The app will be tested on a minimum of two different devices to ensure that the layout works effectively and looks good on multiple devices and both in landscape and portrait modes.

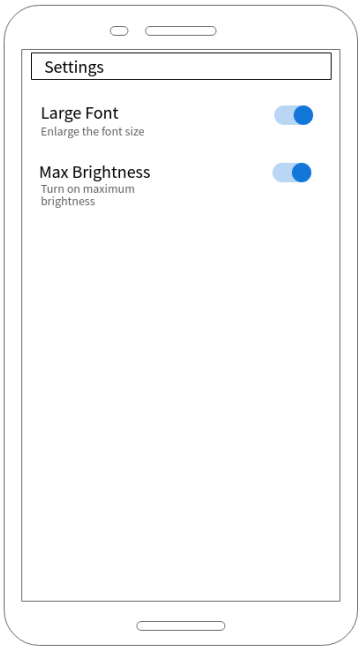
Because the UI design is relatively simple, it should be straightforward enough to make minor adjustments as necessary so that it looks good on multiple devices.

Juice Detail Screen:


Landing Page/Juice Listing:



Settings Screen:





Joke Screen:


## Navigation and User Input

Navigation will work by tapping on a juice item will load a new activity displaying details. The back button will be used to return to the main list. A menu bar will be included for settings, random jokes and also for editing and deleting entries.

To enter new data, the user will be able to tap on a data field and use the Android keyboard to input new information into the application.

## Web Services and Database

Web services will be included in an activity separate to the two main activities. The user will be able to navigate to the random joke activity where the application will send a request to a server and if successful, the server will return a joke which will then go on to be created into a Java object and used to populate views.

A local SQL database will be employed for data persistence.