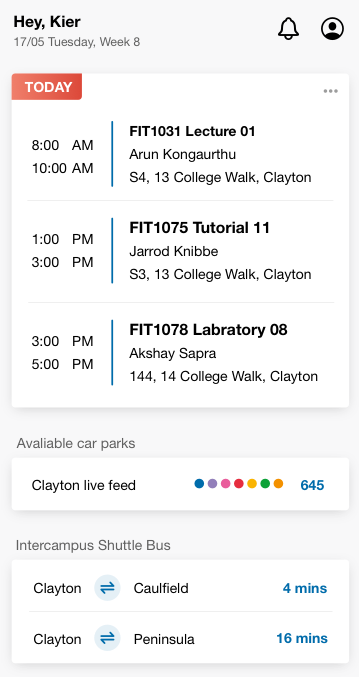
# Monash University Mobile Coding Challenge

Your mission if you choose to accept it is to create the screen provided to you below.

****

**Context**

The picture above is a snippet of an existing application. Think of it as the dashboard for students where important information from the app is summarised in this screen.

You have a custom navigation bar at the top that shows the student’s name and the study week that they are in. A couple of buttons on the right that directs them to different components.

Immediately below that is a summarised list of their timetable for the day

Followed by another section that shows the car parks

Finally a section showing the upcoming buses.

**Task**

Recreate this as a Kotlin based, single page Android app. You do not need to implement any of the button functionalities on the screen and tapping on any of the views do not require any transitions to other pages.

**Requirements**

* The data does not need to be fetched from a service. However the data source needs to be represented locally (i.e do not make these static views with default values). Yes, use fictitious data.
* The list and sections must all be represented in a single recycler view. **Do not nest recycler views**.
* Pay **careful attention** to the design, e.g the shadows, offsets and corners. Hint: There is a slight y offset for the drop shadow.
* You are welcome to use whatever font you like but you must maintain consistency.
* Use git version control and track your commits while developing this app. *We will not be using this for calculating the time spent on the app.* We will need access to this repository to assess your code.
* Randomize the amount of rows in each session. OR give a method for us to specify that we can see 1 row per section, 2 rows per section and 3 rows per section.

**Criteria**

The following will be assessed:

* Attention to detail. This may be an uncommon UI in material design where we are specifying the offset of the shadow. We will be looking at how you are going to achieve this across each of the views in the recycler view and how the components can be reused to build single or multiple item lists.
* Code Reusability
* App architecture
* Documentation
* Unit Testing (some ideas: create a unit test that gives you the next 3 classes based on the current time)
* Functionality
* Use of modern features
* Bugs/Warnings

**Bonus Marks**

The following are not mandatory but if implemented could earn you additional points:

* Data is fetched from a remote service
* Remote service done with Firebase