Assessment library choice rationale

For my assessment I chose the JQuery Mobile library to achieve my goal of retrieving and presenting the user data requested.

The reasoning is as follows:

The requirements are fairly simple for this task, and no interaction with the mobile device hardware is required, so advanced library interaction is not required here. Being Dublin, one can assumed that mobile data coverage will be adequately available for the data connection, so offline caching of data is not required. The ability to fetch real-time information should be key as the user would want to have up-to-date information on bus timetables for their stop, and would not want to miss a bus due to inaccurate information provided by the application.

A lot of application to screen interaction and updating of separate user interface (UI) components is also not required – if it was, React or Angular may have been better suited to the task. For instance if the use case was to be real-time tracking of the bus in a map.

If heavier UI interaction with the app and the display components was required I would have chosen React over Angular, due it it's widespread use and growing popularity, and capability in updating the display components independently without refreshing the whole page.

But, in this case, a single file web page with the required accompanying script loaded is lightweight and responsive enough and will be more than adequate to the task. The capability of JQuery Mobile to download and load a single file and using AJAX to render between the pre-fetched "virtual" pages in a single made it the library of choice for this. It is also the lightest framework of the available choices, which is also key.

Due to the nature of the use case for the app, and the fact that it is required to be used by users on a mobile device, it is very likely that it will be used on smaller screens that requires finger-friendly interaction with larger "click" (touch) areas. Minimal entry is required by a user into the app's interface, and the ability to display these on smaller touch-enabled mobile screens is also key.

JQuery Mobile already also has all the capabilities to interact with a remote request due to it's support for XML, and especially the de facto standard now, JSON. It can easily retrieve the JSON text in the API, take the required information and format them into the virtual pages required to present the real-time information to the user. It has very good documentation, fairly easy to understand and debug, and has a rapid development and release path.

The latest versions of JQM has popup support for iframes with third party widgets, with map and video rendering capabilities – however these could make the pages

more sluggish and taxing on the hardware. One of the biggest criticisms against the use of JQM for developing this application is that it looks the same as the millions of other JQM apps (this could also be seen in a positive light).

Future versions of the app could include integration to social media, or could suggest a coffee shop or even vending machines in the vicinity. One could then have a cuppa' with your mates waiting for the bus, munching on a chocolate from the vending machine.