Sarah Walker B4050530 TM352 EMA

Question 2

a)

|  |  |  |
| --- | --- | --- |
|  | **Functional Requirement** | **Possible obstacle** |
| **FR1** | The app must display details of available products with their description, picture and standard asking price. | Multiple users of the application and web portal will be potentially accessing the product data and placing orders, and therefore reducing the available amounts while others are viewing this data. This means there is potential for this information displayed to the user may not correspond to the data in the database. This would probably be a consideration for the database developer. |
| **FR2** | The app should allow the salesperson to compile an order, including data of the included products, prices agreed, current time and location. | The user of the application may not have their location services switched on on their mobile device, or may have not allowed the sales app permission to access current location information. This means the location information will not be available when sending the order. |
| **FR3** | The app should allow the user to review places that previous orders have been made | The provider of the service being used to view and display the map and pointers corresponding to the places orders may be experiencing a service outage. In this case the app will not be able to use this service, and therefore will not be able to provide details of the orders made. |

Application guide

**Functionality**

On application load the only items initially displayed to the user are inputs for the user name and ID of the client that the order is being made for. I made this design choice as all of the API calls for information for the UI require the user name, and it seemed prudent to hide these UI features until the data for them could be retrieved. Once the user name and client ID have been entered the app firstly checks that the entered user name is of a valid format, and then various API calls are made using this information. The previously hidden portion of the user interface (UI) is then displayed using the information returned from the API calls. Also, at this point, a new order is created behind the scenes, ready for products to be added, and the location API is called to get the current location which is stored ready for use throughout the app. The retrieved client data is used to display their name, address and location to the user so that it is clear who the order is for, and the time is displayed using a native JavaScript function.

The product catalogue is displayed to the user, allowing them to select the currently visible product to add to their order by using the 'Add to order' button. The 'Agreed price' and 'Amount' fields must also contain a value when this button is pressed, or the call to the API will fail. Once the API returns with a success response, the added order item is displayed to the user in the order items list. The sub total, VAT and total for the order are updated to reflect the newly ordered item price and quantity. Finally, a map is displayed to the user to indicate the location of all orders made.

**Installation instructions**

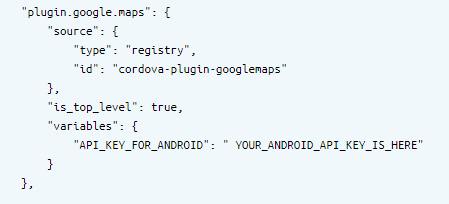
To allow the full functionality of the application, it is required that two additional Cordova plugins be downloaded and referenced within the code. These plugins allow the location and map functions of the app to operate. Firstly, an API key must be obtained from Google to allow the use of their map APIs. This can be done at this url [https://developers.google.com/ maps/ documentation/ android-api/ signup](https://www.open.ac.uk/libraryservices/resource/website:110050&f=29137). Once this is obtained, the follow two commands should be run n the command line, in the root folder of the application:

cordova plugin add cordova-plugin-googlemaps@1.4.5 --variable API\_KEY\_FOR\_ANDROID=YOUR\_ANDROID\_API\_KEY\_IS\_HERE

cordova plugin add cordova-plugin-geolocation

The 'YOUR\_ANDROID\_API\_KEY\_IS\_HERE' string in the first command should be replaced with the API key that was obtained from Google.

It is then advised to check the project's plugins/fetch.json file contains your API key for google maps. The code in this file should include



Where 'YOU\_ANDROID\_API\_KEY\_IS\_HERE' is replaced with your API key.

These steps should allow the app to run fully.

**Word count: 463**