MegaMax

TM352 – EMA

Question 2

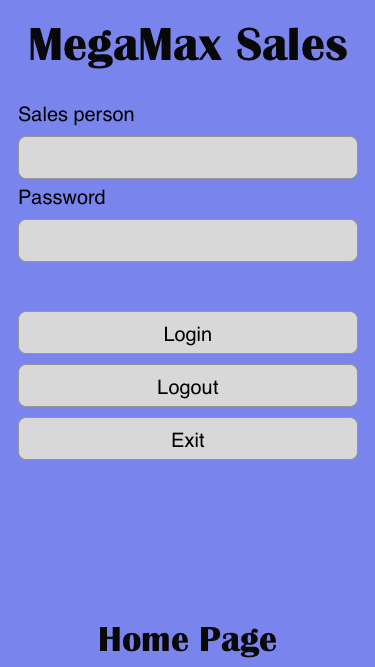
Darren Tynan (B7739482)



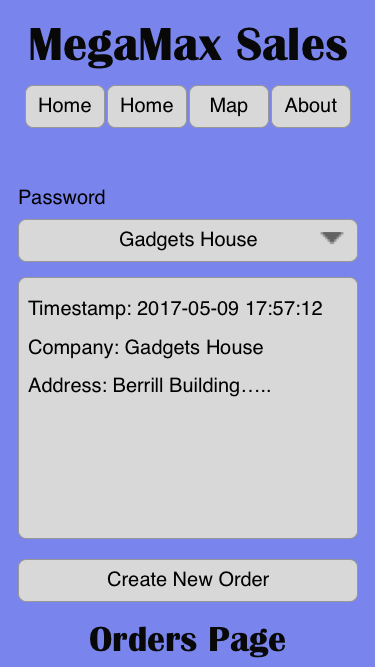
**Question 2 (a)**

|  |  |  |  |
| --- | --- | --- | --- |
| Functional Requirements | Name | Description | Obstacle |
| FR.1 | Logout | The application should allow the current user to logout for reasons of security in regards to unauthorised users being able to utilise the MegaMax api. | This would require a redesign of UI and implementation of methods. Additional functionality for ‘password reminder’ would need to be implemented. |
| FR.2 | Client selector | The application should have the ability to obtain a current list of clients from the api rather than hard coding existing clients and needing an application update for new clients. | None already implemented. |
| FR.3 | About | The app should have the added functionality of an about page. This page should simply but clearly describe the UI elements on each page of the application and the order of operation. | None already implemented.  Update need if changes were made to UI. |
| FR.4 | Stock checker | The app should have the ability to check stock levels of widgets prior to order being placed with possible functionality of a back order option. | There is no currently implantation within the MegaMax api. With several sales people using the application there might be a delay in updating stock levels. |
| FR.5 | Previous/Next Order | Navigate and display history of orders placed by selected client id. | A redesign of the current UI would be required and method implementations. |
| FR.6 | Summary Page | A more detailed summary form with further options of edit price and quantity, cancel widget or cancel order. | Would require a complete redesign of the current UI. The current revision of the api implementations restrict usage. |
| FR.6.1 | Edit quantity | The summary form would allow the sales person to edit this field with entries of zero would be the same as cancel order of this widget. | Would require a redesign of the application and implementation of FR.4 Stock Checker. |
| FR.6.2 | Additional widget | The application should have the ability for additional widgets to be appended to current order. | Would require a redesign of application and implementation of FR.4 Stock Checker. |
| FR.6.3 | Cancel order | The application should have the ability to cancel the current order. | None |
| FR.6.4 | Place order | The application should have the ability to place the current order. | A complete redesign of the existing application business logic would be needed. |

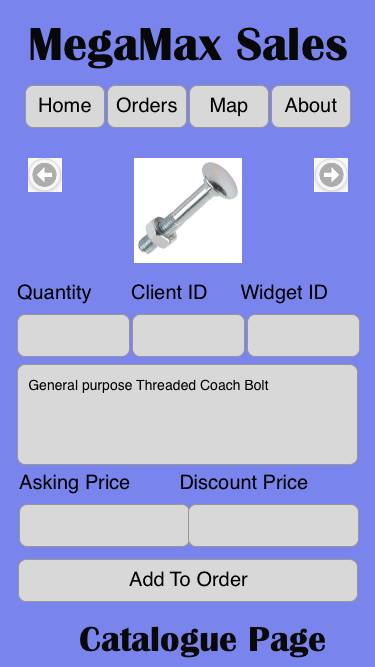
**Question 2 (b) Mockups**

****

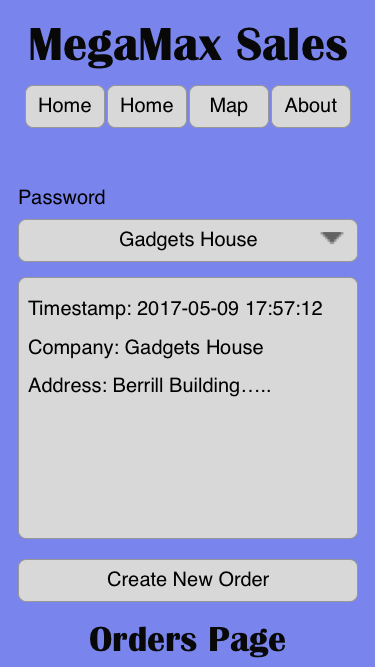
**Figure 1** Mockup of Home Page



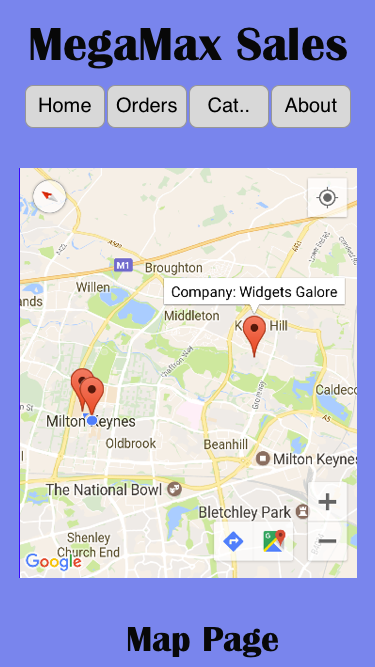
**Figure 2** Mockup of Orders Page

****

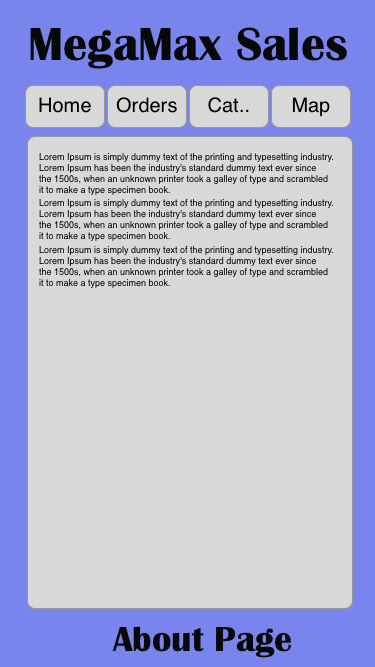
**Figure 3** Mockup of Catalogue Page

****

**Figure 4** Mockup of orders Page after Create New Order

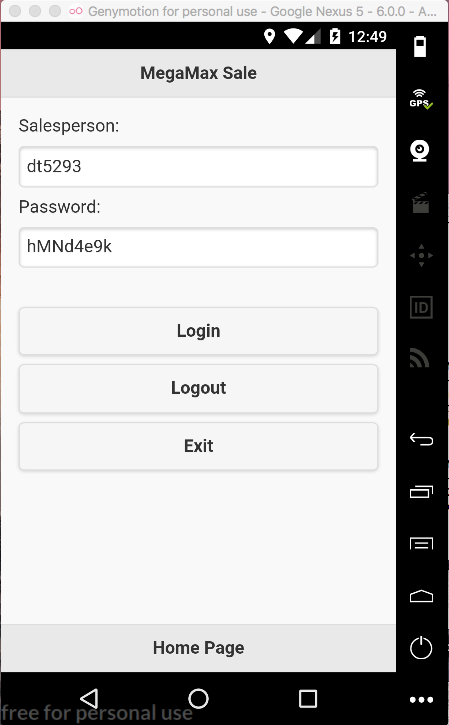
****

**Figure 5** Mockup of Map Page with order markers

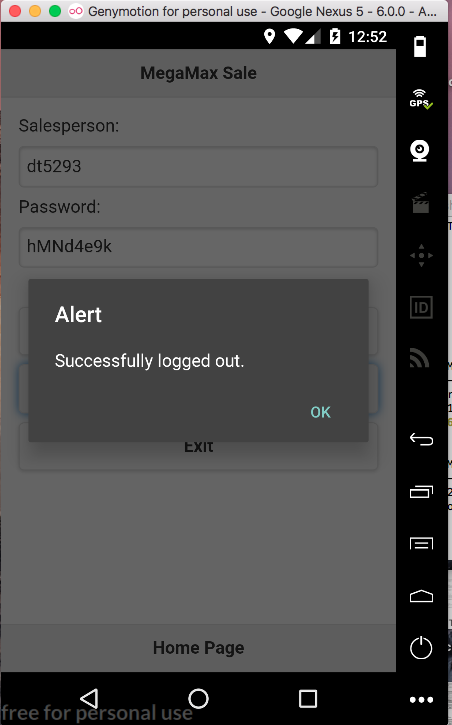


**Figure 6** Mockup of About Page

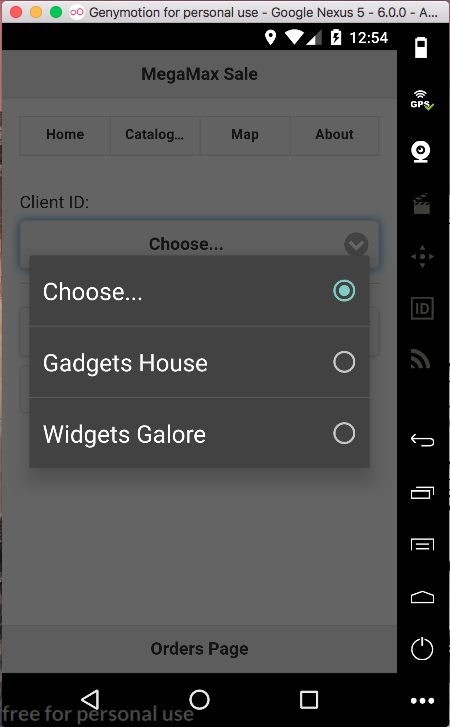
**Question 2 (C)**



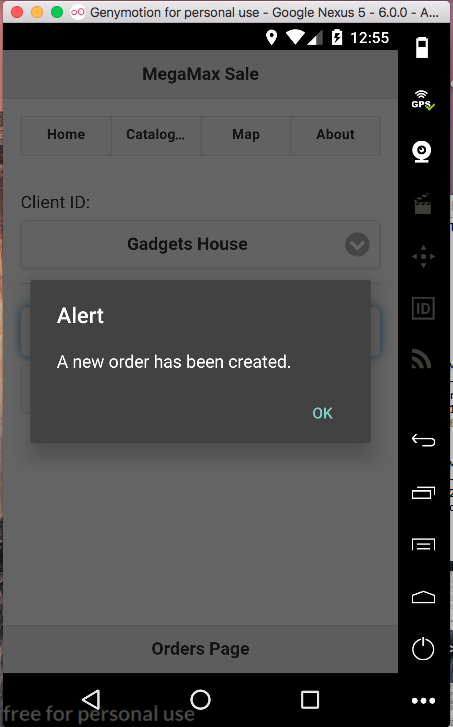
**Figure 1** Screenshot showing the Home Page



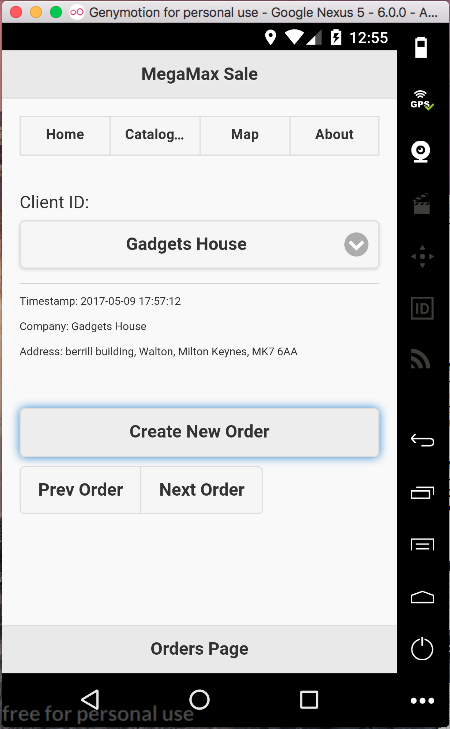
**Figure 2** Screenshot showing successful logged out.



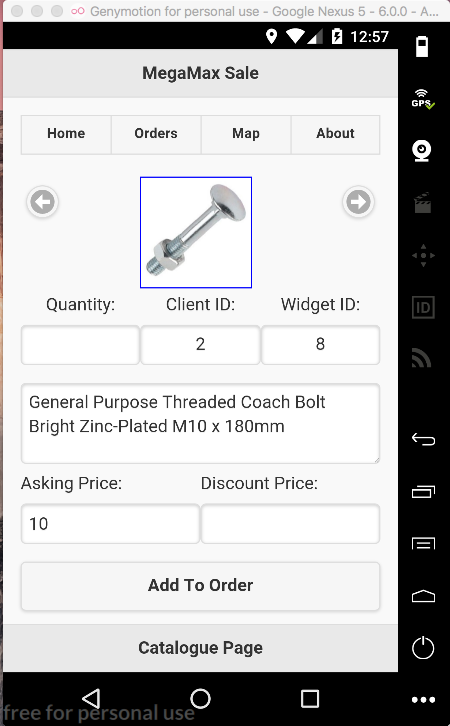
**Figure 3** Screenshot showing the Orders Page with dropdown for client select.



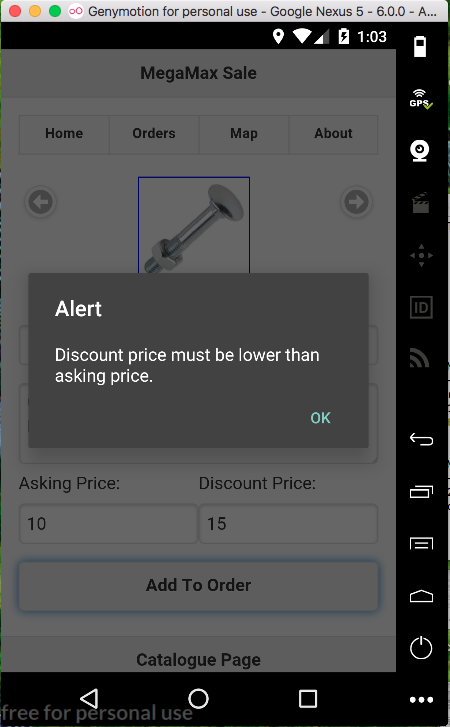
**Figure 4** Screenshot showing the Orders Page with a new order being created.



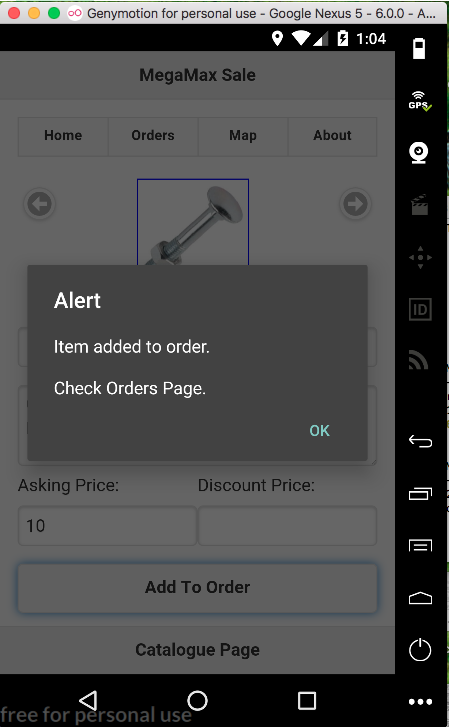
**Figure 5** Screenshot showing the Orders Page with a new order created for the selected client.



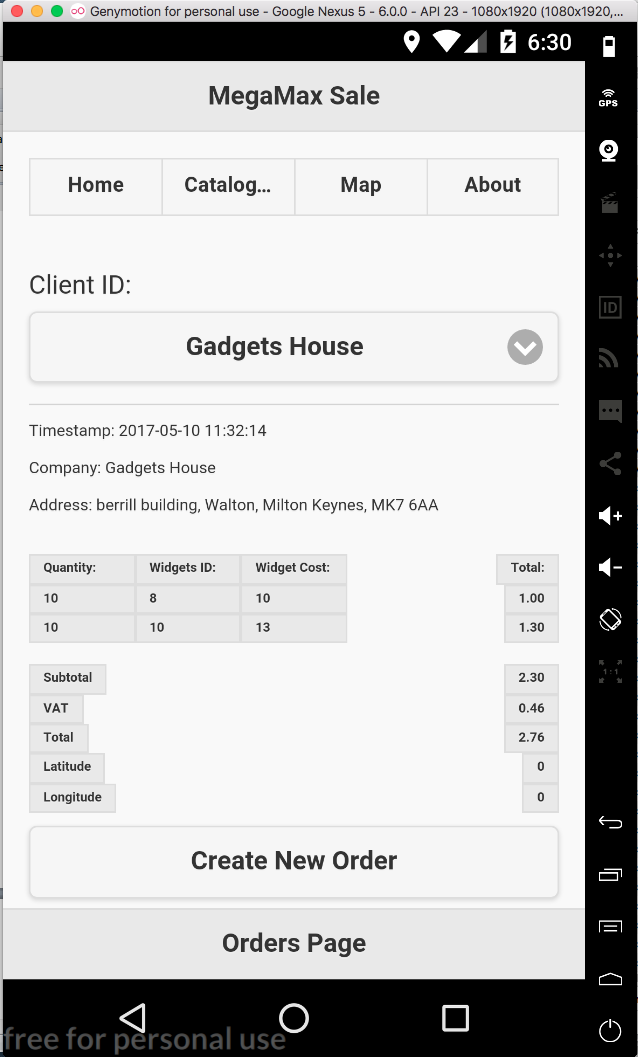
**Figure 6** Screenshot showing the Catalogue Page.



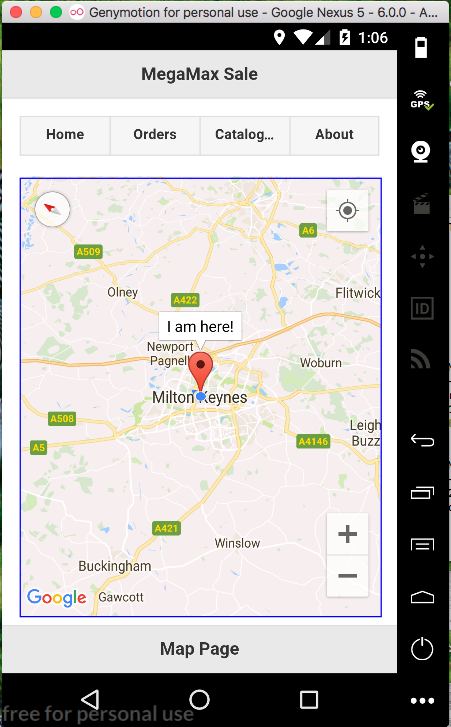
**Figure 7** Screenshot showing the Catalogue Page with error detection.



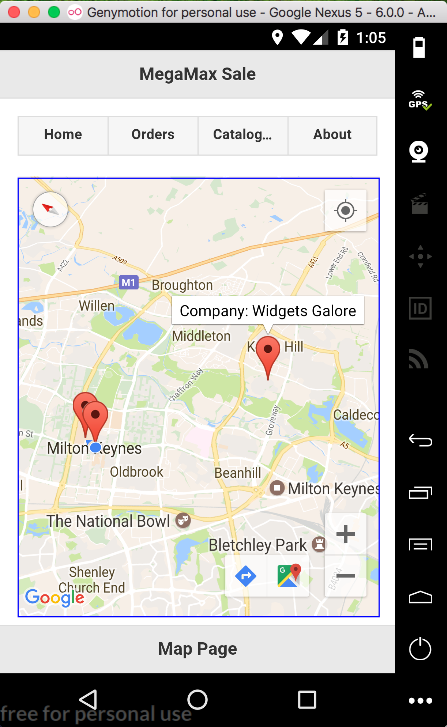
**Figure 8** Screenshot showing the Catalogue Page with item ordered.



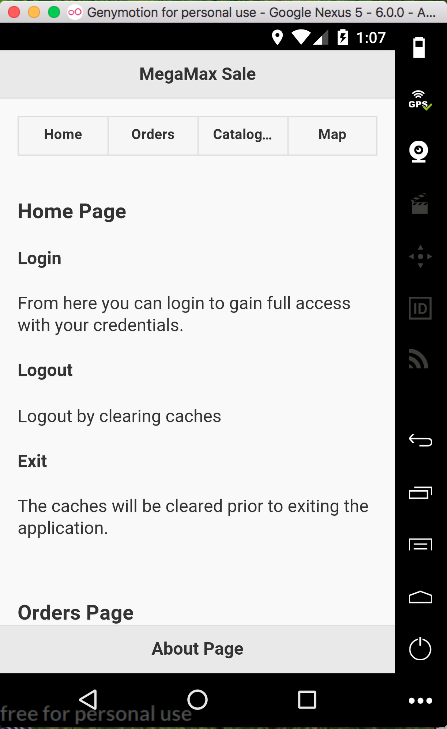
**Figure 9** Screenshot showing the Order Page with widgets added to order.



**Figure 10** Screenshot showing the Map Page with marker for device location prior to order being placed.



**Figure 11** Screenshot showing the Map Page with markers for orders location.



**Figure 12** Screenshot showing the About Page with top half of instructions shown.

**Application outline and navigation**

As can be seen from the above screenshot’s the application is complete (well at least my interpretation).

**Functional Requirement 1.1**

From **Figure 1**, after the salesperson credentials have been entered and upon clicking Login; the username is validated according to FR1.1 if the credential are valid then the user is directed to the Orders Page as show in **Figure 3**. If not than an alert box is shown not allowing you to continue.

**Client Selection**

In the essence of separation **Figure 3**, allow you to select a valid client from the dropdown list prior to creating a new order.

After selecting a client from the dropdown you can Create New Order and the timestamp, company name and company address is added to the order summary section as can be seen in **Figure 4.** If you have not selected a client from the dropdown then you will be shown an alert if you try to Create New Order.

**Functional Requirement 1.2 and 1.3**

As can be seen from the Catalogue Page you can iterate over the widget list via the arrow buttons. The Client ID along with the Widget ID, description and Asking Price updated to reflect the current index.

As shown in **Figure 7** I have implemented error checking for Quantity less than 0, or Discount Price greater than Asking Price. If all checks pass then the widget is added to the order as can be seen in **Figure 8**.

**Functional Requirement 1.4**

On each Add To Order event, the selected widget is added to the order summary with the subtotal, vat and total be calculated in real time. As can be seen in **Figure 9**.

**Functional Requirement 2.1**

From the Map Page as can be seen in **Figure 10** then device’s current location can be seen.

**Functional Requirement 2.2**

From the Map Page as can be seen in **Figure 11** after orders have been placed then device’s current location can is show but also markers for the days placed orders.

**Plugin’s used**

* [cordova-plugin-geolocation](https://cordova.apache.org/docs/en/latest/reference/cordova-plugin-geolocation/)
* [cordova-plugin-nativegeocoder](https://www.npmjs.com/package/cordova-plugin-nativegeocoder)
* [cordova-plugin-googlemaps](https://www.npmjs.com/package/cordova-plugin-googlemaps)

All installation and use was covered by instructions in the above links.

**Cordova plugin geolocation**

This plugin provides information about the device's location, such as latitude and longitude.

To install from terminal of project folder:

cordova plugin add cordova-plugin-geolocation

The implementation can be found in the index.js file.

navigator.geolocation.getCurrentPosition(onSuccess, onError, {enableHighAccuracy: true });

**Cordova-plugin-nativegeocoder**

Call nativegeocoder.reverseGeocode() to transform a latitude and longitude into an address or nativegeocoder.forwardGeocode() to transform an address into a latitude and longitude.

To install from terminal of project folder:

cordova plugin add cordova-plugin-nativegeocoder

The implementation can be found in the index.js file.

Nativegeocoder.forwardGeocode(succsess, failure, address);

**Cordova-plugin-googlemaps**

To install from terminal of project folder:

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

The api key was obtained from my Google developers account portal.

The implementation can be found in the index.js file.

All code can be found in the ‘function UpdateMap(address)’ method near the top.

**[WORD COUNT : 487]**