IllumiMate Development Diary

## 28/10/2022:

* Researched best development software for cross platform mobile apps.
* Researched mapping platform that will give us the ability to add what we want and use it in a way that would reduce cost to clients. Discovered OpenStreetMaps.
* Researched ways of adding route logic to OpenStreetMaps and found Graphhopper and ORSM. Graphhopper has easier to understand API.

## 30/10/2022:

* Decided to use Google developed Flutter and Dart to be able to bring Illumimate to Android and iOS, the primary targets for delivery.

## 5/11/2022:

* Installed Adobe XD for Illumimate app wireframing.
* Installed Adobe After Effects for video animation and editing
* Signed up for Postman to test responses with Graphhopper
* Signed up for an API key to Graphhopper

## 6/11/2022:

* Researched how to get coordinates for street lights into Graphhopper as attributes/custom Encoded Value. Which will then be used to calculate the weighting for edges(parts of route).
* Installed JOSM (Java OpenSttreetMap Editor). Seeing if this would yield any clues as to how to make this integration of street light information. Also installed JDK 19.
* Using geograbrik to download an offline copy of openstreetmap pbf file of Australia. File is 594 MB.

## 7/11/2022:

* Cloned github repository for Graphhopper 7.0
* Tried add tags to OSM through JOSM and then have them read in as Encoded Attributes is Graphhopper.
* Edited the Java files in Grasshopper to included
  1. Implemented a ‘Lit’ enum
  2. The Lit enum was added to the DefaultEncodedValueFactory class
  3. Implemented a ‘OSMLitParser’ TagParser class
  4. Added the ‘OSMLitParser’ TagParser to the DefaultTagParserFactory
  5. Installed Maven and built Graphhopper 7.0 Java binary.
  6. Using the endpoint /info on the localhost server found that ‘Lit’ was now an available tag to be used for calculating edges.

## 8/11/2022:

* Adding Lit = yes to some roads in local neighbourhood, Clifton Beach. Will use this to create two paths to same destination and then create a weighting that looks for lit roads.
* Latest version of graphhopper 7.0 search not working. Had to use git checkout for older hash/version to graphhopper 6.x branch.
* Setup testing profile where priority weighting for lit == YES was 1000 and ‘distance influence’ was 0. Added tag of lit:yes to roads and lit:no or null to others for testing. Worked perfectly.
* Setup profile for most\_lit\_route, fastest\_route, and shortest\_route

## 9/11/2022:

* Investigated options for implementing graphhopper on mobile. Graphhopper offers a repository for android called graphhopper-maps, but this hasn’t been updated in at least two years. Found a different repository boldtrn/graphhopper-maps-capacitor, which has updates 9 months ago and an rebuilt app on F-droid.
* Installed F-droid on my android device to test the graphhopper-maps-capacitor.
* Discovered a new cross-platform native runtime called Capacitor that helps build modern web apps that run on iOS, Android, and the web. Capacitor needs Node v8.6.0 or higher and NPM 5.6.0 or higher. For iOS development we can use Ionic Appflow. We also need to install CocoaPods and Xcode Command Line tools. For Android all we need is Android Studio or Android CLI tools.
* Graphhopper-maps-capacitor has a Unix bash shell build script. Installed WSL Ubuntu on Windows.
* Edited the Graphhopper-maps-capacitor config.js file to use graphhopper localhost api.

## 11/11/2022:

* Tried to build the Graphhopper-maps-capacitor on WSL (Windows Subsystem for Linux) but it kept failing. Installed VMWare Workstation Player 16 and installed a copy of Lubuntu. Will attempt to build the package on this.

## 12/11/2022:

* Installed and launched Adobe XD to build the user interface for Illumimate. Looked at the most common smart phone resolutions in Australia as at October 2022. Found that 390x844 px was the most common with 14.47%. The most common smallest resolution is 360x800 px, at approximately 6% according to <https://gs.statcounter.com/screen-resolution-stats/mobile/australia> Therefore we will build for that smaller size and allow elements to slightly adjust up to the larger resolutions. Mobile-up responsiveness pattern.
* Downloaded the UI kits Tactiv and Navigo, which is specifically for a navigation app, for Adobe XD. From these we can use elements for our wireframe prototype.
* Found issue with Graphhopper-maps-capacitor, the Graphhopper-maps submodule did not have permission to be cloned within the Graphhopper-maps-capacitor repository. Manually cloned Graphhopper-maps and then proceeded to run build.sh.

## 17/11/2022:

* After talking to the developers of Graphhopper I finally figured out how to configure the routing and geocoding options for Graphhopper in the options.js file.

## 20/11/2022:

* Graphhopper-maps-capacitor was dead end with lots of bugs. Moved onto using GraphHopper Maps as a backend UI renderer.
* Re-installed entire VPS server to Ubuntu 20.04 because 18.04 LTS could not build Node v 18.x, necessary for building GraphHopper Maps.
* Modified the GraphHopper Maps config file to use local server for routing server and entered GraphHopper API key for geocoding server access.

## 21/11/2022:

* Found GraphHopper Maps server kept crashing. Installed nodemon-webpack-plugin from NPM. Setup Webpack to use nodemon-webpack-plugin so that when running the server it would restart automatically if the server unexpectedly crashes.
* Managed to use Webview stateful widget in Flutter/Dart to get access to our GraphHopper Map server at http://191.96.57.73:3000/