Smart City App – Required Research Document

Smart City is a community-driven mobile application built on top of the most high-end technologies in the market. It takes advantage of the brand-new API launched by the City of Ottawa. It is built using the React Native framework, a powerful and well know JavaScript Framework developed by Facebook. Moreover, it uses Google Maps for geolocation data. The purpose of this document is to provide information about the technology and processes used for the application development.

- React Native
    - React Native is one of the most used JavaScript libraries in the market. Launched by Facebook in 2013 and maintained by their community since then, this framework intends to provide cross-platform solutions. From just one development environment it's possible to release both iOS and Android native applications.
    - MGIS has been using this technology since 2016 and will move forward with it since it is flexible enough to hold all their projects in different market niches.
    - Researched Topics:
        - Navigation – React Native navigation stack is a very powerful feature. However, when you first implement it, it must be done carefully while thinking about further updates to the application. The following step will be adding the pages to the navigation stack and customizing their labels and icons if needed. (source: https://facebook.github.io/react-native/docs/navigation.html)
        - Platform Detection – Implementing both platforms at the same time decreases development time and lowers costs. There are some specific features/UI specifications for each platform. Identifying which platform the app is running is seamless and easy with React Native. It provides you with components that can be used with conditionals during implementation. (source: https://facebook.github.io/react-native/docs/platform-specific-code.html)
        - Hardware Access – Regarding the main features of our application, using the camera and geolocation (GPS coordinates) is crucial. Using React Native we found a simple way to implement the camera and GPS features. The standard React Native tool supports most features, however, further development requirements may need some external plugins found on their community forum. (source: https://facebook.github.io/react-native/docs/cameraroll.html, https://facebook.github.io/react-native/docs/geolocation.html)
    - Official Source: https://facebook.github.io/react-native/
- Native Base
    - Researched Topics:
        - Layout Applicable to each platform's guidelines – Native Base is a powerful UI kit for React Native. The best feature they provide developers is the cross-platform guidelines. The only implementation required by the developer is adding components from both Android and iOS into the same View and the React Native detects which platform to switch to. (source: http://docs.nativebase.io/Customize.html#Customize)
        - Iconography – With Native Base, there are a huge number of icons that we can work with. Moreover, it is possible to follow both iOS and Android guidelines with different icon styles. (source: https://github.com/oblador/react-

native-vector-icons)

  - Official Source: https://nativebase.io/
- Open Data Ottawa - 311 API
  - Researched Topics:
    - Service specifications (GET/POST) – The city of Ottawa recently released access to their API, which includes 311 data. The Ottawa 311 API is the core service used by our application. It will be possible to open and query requests for several public services using this API. This service is part of the Open 311 API (source: http://www.open311.org/), which has many cities world wide in their database. Although the Open 311 API has many features, the city of Ottawa is currently using just part of this structure. So far, what we can use is the request system data and some extra statistics available through their calls. Our application will be prepared to work with any of the cities that use Open 311 in the future, which is a great asset for this project.
  - Official Source: http://data.ottawa.ca/dataset/open311
- Google Maps
  - Researched Topics:
    - Custom Pins – Based on the requests posted to the 311 API, our Map feature will show different pins and info according to the category of the requests. This is going to be done using native Google Maps features embedded in React Native Components. (source: https://developers.google.com/maps/documentation/javascript/custom-markers)
    - Marker Clustering – Google Maps does not allow you to make calls with more than 11 points at once. To overcome this we can either split the points into several calls or create Marker Clusters to identify a group of points in the same category. This latter is the standard solution among geolocation service providers. (source: https://developers.google.com/maps/documentation/javascript/marker-clustering)
    - Heat Map – One desirable feature asked for by the client is to have a Heat Map layer. It should show the requests in a higher level on the map, showing city's problems by zones and bigger areas. However, this will be discussed further in the future.(source: https://developers.google.com/maps/documentation/javascript/examples/layer-heatmap)
- MGIS Rest API
  - Researched Topics:
    - Create User
    - Update Profile
    - 2-Way Authentication Process
    - Create Request
    - Update Request
  - These services are internal MGIS' services. They'll be provided or customized according to the project's needs.