Alex Banning

October 27, 2024

**Checkpoint 2: Simple Prototype Document**

**Features needed to deliver MVP + Breakdown of Tasks:**

* Location 1 on MainPageView needs to be able to take in any data including user’s current location as an option

Tasks:

* + - Edit hardcoded Location 1 data to take in user input using TextField and State variables
    - Enable permissions requests in app and create a popup onAppear to ask for user location service permissions while using the app
    - Pass correct data to subsequent views and methods to accurately calculate meeting points
* Both the location buttons on the MainPageView need to navigate to a new view where suggestions autofill as the user types

Tasks:

* + - Create a navigation link to the new InputLocationView instead of an onTapGesture alert
    - Query an API to get autofill suggestions based on input (maybe Apple MapKit or something else)
    - When the user selects a location, correctly pass that data back to the MainPageView
* The MeetingPointsView currently uses an algorithm that relies on hard coded major city data. For the US, this is generally okay because the data set is not that large and can be used in the app, but as I extend functionality, this should probably be done with some sort of population-based API call

Tasks:

* During the meeting points search algorithm, search along the route polyline for any cities surpassing 10000 population in a radius of 50 km
* Use an API to figure out locality populations to sort with
* I should also display markers on the map for all of the potential meeting points, so the user can see at a glance what the possibilities are
* The NearbyResultsView needs to be easier to interpret

Tasks:

* + - The Map needs to be centered at the meeting point and display a radius that it is searching in and stay relatively static
    - When the user taps on a nearby result it needs to highlight on the map (currently doing that but also adjusting the map view every time is too busy)
    - The MKLocalSearch query needs to be refined to include more relevant results (currently searching for “hotel(s)”, “restaurants”, and “activities”, but the restaurants should be all food and drink options and the activities are vague and sometimes weird like local gyms, schools, and other places that aren’t really relevant for the purposes of this app.
    - I don’t necessarily believe the filter feature is necessary at this point as long as my MKLocalSearch queries are inclusive enough to get some good results and the List shows proper titles of categories
    - When the user taps on a nearby result it should show the Apple Maps details. This functionality is not quite built yet and leaves out some key information the user may want
* The backend structures and flow

Tasks:

* As I update the search algorithms and MKLocalSearch results, I need to be wary of how I use and store this data in the app
* Currently storing serialized JSON data of a structure called FavoriteResults = {

**let** id: UUID

**let** location1Name: String

**let** location2Name: String

**let** meetingPointName: String

**let** meetingPointCoordinate: EncodableCoordinate

**var** favoritePlaces: [String] = []

}, where favoritePlaces is an array of Strings containing the names of the favorited places. In the full app, this needs to include more data of the place in question to display the Apple Maps more details popup.

* UI Updates

Tasks:

* + - The UI needs a lot of work. Looks like something from 2010, so I need to get some users testing it and receiving feedback on appearances. Maybe brush up on modern design philosophy and guidelines.

**Mapping of Features and Values:**

**Value Provided by App: Find a meeting point easily; find things to do near that meeting point; always have a place to stay overnight if needed. Do it all faster and simpler.**

|  |  |
| --- | --- |
| **Feature** | **Value** |
| Custom location entry | No matter where you or a friend lives, you can find a place to meet up |
| Find Meeting Points | Automatically finds a meeting point, including major cities near the route, that has at least a place to stay and 5+ things nearby. Lists options so the user can contemplate where they want to be. |
| Display Nearby Results | Find things to do automatically near a meeting place. Points of interest are searched automatically no user input required, speeding up the process. |
| Map Display of Nearby Results | Quickly see where a nearby result lies in comparison to the region you’re searching in. Gives user a sense of distance between things for planning purposes. |
| Favorite an Item | Favoriting things allows a user to save it for viewing again later. That way if they want to leave the app and come back to planning another time they can pick up where they left off. |
| View Saved Trips | Allows the user to view all their saved trips from previous searches. That way if they ever need to go back to an old search for reference that data is available. |