MeetMeHalfway

Alex Banning  
Version # 1.0

# Summary of Project

*MeetMeHalfway is an app that enables maintaining relationships. As people get older their social circles tend to get smaller; friends get new jobs in different cities, families separate, and it becomes more difficult to find the time to plan meet ups. MeetMeHalfway seeks to solve these problems by providing an easy-to-use platform for finding meeting points in between two locations. Whether across town or across the country, MeetMeHalfway’s simple UI allows the user to enter a location and will find a middle point between the user’s current location and the user’s input. From there, the app will search nearby for food and drink, living accommodations, events, and other activities. If there is not much to do in the middle, MeetMeHalfway will search nearby along the route to find the nearest bustling town or city that has more things to do.*

# Project Analysis

## Value Proposition

*One problem with having friends in a different city is deciding where to meet up. I have lots of friends that live in Seattle, and I often don’t want to drive all the way to them in traffic and struggle to find parking. Sometimes they will drive to me, but there are only so many things to do in Woodinville, and after a year or so it starts getting boring always doing the same things. Instead, we sometimes suggest a mutual meeting location, like Bellevue or Redmond, to cut down on the driving time. The tradeoff, however, is we typically spend a couple hours deciding where that location should be and searching Google for activities and food in that area. Sorting through those results can often take an additional 30 minutes to reach a final plan.*

*This problem is exacerbated when the distance between friends is greater. One person could fly to the other person’s city, but depending on how much there is to do in those places, we run into the same boredom problem as before. It is also quite expensive for one person to pay for their travel.*

*Instead, MeetMeHalfway seeks to provide value by solving these problems with a simple tool. We can cut way down on the time it takes to pick a meeting location and find things to do. The app can also filter results based on common interests and categories to make it easier to reach a decision.*

## Primary Purpose

*The purpose of this app is to provide people with a useful tool that can enable better friendships. I want to help bring people closer together. As I have gotten older and subsequently busier, it has been very hard for me to maintain my friendships and supply the time necessary to be social. I have seen others on social media echo this feeling, and I believe my app is the necessary tool to make getting together over distances easier than ever. Facetime and texts only go so far in a friendship, and I believe enabling easier and faster physical closeness between friends is the key in forming and maintaining stronger friendships.*

## Target Audience

*Typically, this app will be useful to adults seeking to engage with their friends more, especially those who are further away. However, it is not exclusive to those people. I would imagine that most younger people (i.e. college-age and younger) already have a large pool of nearby friends due to their common experiences in school and activities and don’t typically need this app. Working adults lack this physical proximity to and shared experiences with many of their friends, so this tool could benefit them more.*

*Ultimately, it will serve anyone struggling to figure out a good place to meet up with someone.*

## Success Criteria

*This app’s success is determined by two metrics: number of people actively using the app, and the time it takes to plan a meet up on the app vs. other methods. If even one person uses the app and considers it helpful or easier or faster than other methods, I would label that a success. I plan on running tests to time the app in comparison with other methods (manual Google search, genAI, etc.).*

## Competitor Analysis

*There are a few competitors that are popular on the app store right now called “Whatshalfway” and “Meet Halfway.” Both these apps are relatively simple to use but lack nuance in the results.*

*Whatshalfway allows the user to enter two locations and will always show the midpoint as equal driving distance from both locations. This is helpful for smaller drives, but it starts to break down with longer distances that may take 3+ hours to drive. Furthermore, since the result is always static there are sometimes midpoints with absolutely nothing to do. For example, when entering Seattle and Boston, the app gives a result in the middle of North Dakota with one restaurant and nothing else in sight. This app does offer extensive filters and search categories, but the user must pay to upgrade to a premium version.*

*Meet Halfway adds additional functionality by allowing the user to enter multiple starting locations then uses geometry to find a center point. Because of the precision of the math though, the results have the same problem of showing nothing around the midpoint. This app is also littered with ads, which makes the user’s workspace and map view too small to coherently read and interpret.*

*MeetMeHalfway seeks to fix these problems by removing ads and using a smarter algorithm to determine a meeting location that has at least five results in core categories like “food and drink,” “living accommodations,” and “nearby activities/entertainment.”*

## Monetization Model

*I am strongly against the grotesque use of advertisements in everything, so I will refuse to use that in an app that relies on being fast and simple.*

*Instead, this app will cost $0.99 and allow users to donate more if they would like to. I am not necessarily worried about making money on this app, but I do understand that publishing apps on the iOS app store requires purchasing a developer license annually. The app itself does not use any paid APIs; it mostly relies on a free API for geocoding location information called Positionstack. This API can be upgraded to a paid version if the number of users/API requests increases beyond the free threshold, which is a problem I can tackle in the future.*

# Initial Design

*The MVP for this app at its core must provide the value I have described above. It needs to be faster than alternative processes and it needs to be simple.*

*An app where a user can enter two addresses (or one if location permissions are granted), be shown a map with potential meeting locations, and filterable nearby results for each meeting location would be sufficient for the MVP.*

*A wireframe diagram of the initial design can be found in this project folder.*

## UI/UX Design

* *Location permissions request on startup*
* *TextField element(s) for the user to enter location(s) information*
* *Map view that displays results to the user, with each potential meeting point shown as a marker*
* *On tap of each marker, zoom in and show nearby results on the map*
* *By default, all results will be shown in the area, but there will be a filters button to select*
  + *Food and Drink*
  + *Activities and Entertainment*
    - *Movies*
    - *Games*
    - *Comedy*
    - *Plays/Musicals*
    - *Concerts*
  + *Temporary Living Accommodations*
    - *Hotels*
    - *AirBnBs*
  + *Outdoor Lifestyle*
    - *Hiking*
    - *Beaches*
    - *Mountain resorts (skiing or mountain biking)*
    - *Natural sights*
* *Upon filter selection, the map will reload with only related criteria*

*See wireframe diagram for more information about layout of elements.*

## Technical Architecture

*In the MVP, I want to limit data storage as much as possible to keep cost as low as possible (no reliance on a server or database to be maintained). With the nature of the geocoder API I am using and free Apple MapKit services, I believe I can get away with using only memory to do the operations necessary to display information to the user. However, some data structures and local storage may be necessary in the future if a user wants to save some trip information to check again later.*

*Data structures: Lists, potentiall some custom data structures for storing “My Meet Ups” locally (potentially an array of Lists or something similar).*

*Storage considerations: Maybe some local storage for favorite places or “My Meet Ups page,” but I am currently undecided on that.*

*Web Interactions: Positionstack API, Google Places API, Apple MapKit native API, potentially Yelp API for review information.*

# Challenges and Open Questions

Some potential challenges could include the cost of API use for things like Google Places and Yelp. Limited API requests are free, but if the user base of this app increases to a large enough size, I would need to adjust the monetization model to account for the increased cost of using these services. One monetization model I would consider that seems less intrusive than pop-up or block ads would be to allow sponsors to pay me to push their results higher on the list. I could also consider adding additional features to the app based on user reviews that customers could pay a premium for.

Another challenge is going to be the performance. Because this app relies on doing location geocoding, results gathering and sorting, and map displays with lots of data processing in between, I need to ensure my code is as efficient as possible. On the initial search for midpoints, I can have the app already gather a list of nearby results for each meeting point possibility and cache that information in a data structure in memory for the fastest possible access. Another potential problem with the performance is the limiting speed of the API calls I am making. Some testing will need to be done during the building process of this app to figure out the impact of those API calls and exactly what is necessary for the app to succeed in creating the value I have outlined.

Some questions I still have

* How well does this app design work in comparison to other methods?
* How easy and intuitive are the controls and design?
* How could I introduce genAI to this app idea to make the results more relevant and complete?