Trek

Geographic Information System (GIS)

Last Update 2024-07-07

Version 1.0

Table of Contents

Document Information

Revision History

Date	Version	Status	Prepared by	Comments
2024-07-07	1.0	Approved/Internal	Matthew Kang	

Approval

Role	Name	Signature / Initial	Date
Developer	Jacob Zhu	JZ	2024-07-07
Developer	Justin Lieu	JL	2024-07-07
Developer	Kevin Xu	KX	2024-07-07
Developer	Matthew Kang	MK	2024-07-07
Developer	William Xiao	WX	2024-07-07

Introduction

A **Geographical Information System (GIS)** is crucial for delivering requirements for Trek. A user must be able to interact with geographical data with Trek when building a trip. The goal of this document is to explore different GIS options Trek can integrate, while considering the following:

Compliance — *"Are we complying with GIS' Terms of Service? Are we stealing data and potentially committing academic misconduct?"*

UI/UX — *"Does the UI Look Good?"*

Usability — *"Does it provide good usability with sufficient information of places and provide quality search and personalization?"*

Sustainability — *"Will the system be financially sustainable with the integration costs? Can costs be lowered by caching or storing data?"*

Maintainability — *"Is the information easily manageable and does not require frequent extensive maintenance on Trek's end?"*

Dependency — *"Can we manage our own information without relying on the GIS? Are we able to migrate from this GIS to another system easily"*

What a GIS Offers

A Geographic Information System (GIS) for Trek should offer the following features or services:

Maps - Various visual dynamic maps imagery. Displays maps for 'Map View' and location markers, 'Pins' within the map, and geographic information (longitude and latitude) to provide context when searching for places.

Routes - Provides routing information in 'driving', 'transiting', 'cycling', or 'walking'. Find best routes from place to place in different transportation methods.

Places of Interest (POI) - Provides information on places and ability to search for places. Gathers and provides place information including place ID (identifiers), name, location, description, address, photo, and reviews.

Comparison of GIS API Integrations

Some GIS services do not provide all three packages (Maps, Routing, POI). Some GIS services do not allow caching or storage of data. Some GIS services are outside our budget.

Summary

API	Maps & Routing	Places of Interests	Caching & Storing	Pricing	Description
Google Maps Platform	Best	Photos, Reviews	Restrictive	High	Best up-to-date POI information
FourSquare	N/A (POI only)	Photos, Reviews	Restrictive	Medium	Provides decent POI and places 'personalization'; best for tourist POI
Mapbox	Good	Integrates FourSquare	Restrictive (non-enterprise)	Medium/ Low	Alternative to Google Maps. Good UI. OSM-based. Second most popular
LocationIQ	No transit routing	Only GeoCoding	Allowed	Low	Fully OSM Data packaged as API. Comes with only Geocoding data (no POI)
MapTiler	Good	Basic	Client-side	Low	
HERE	Good	Basic	30 days or Response Header	Low	No permanent storage of location IDs
MapQuest	No transit routing	Basic	Restrictive (non-paid)	Low	

POI Data: Image, Description, and Reviews Data

Apart from Google Maps Platform and ForeSquare, OSM-based GIS APIs only provide basic POI data. Some GIS APIs do not provide contact information or opening hours. We can integrate the following technologies to provide end-users with these data:

Image – For landmarks, well-known businesses (e.g. McDonalds), and attractions, use Wikipedia API to fetch images that are shareable for commercial purposes.

Description – For landmarks, well-known businesses (e.g. McDonalds), and attractions, use AI generated descriptions.

Contact, Hours, and Reviews – Use TripAdvisor or Yelp API to link review data, open hours, and contact information. Only load review data if the user clicks on it to limit unnecessary API calls. Yelp allows caching up to 24 hours but has no free tier.

In-House User-Contributed POI Data System – Support an in-house user-contributed POI Data system, where users can upload images, description, and reviews to a place.

Proposed Solution

Create an OSM-based **In-House User-Contributed POI Data System** with integration with **Mapbox** for additional non-cached POI attributes such as contact information and hours of operation. The goal is to integrate the best of both worlds: MapBox and OSM.

Benefits:

- Integrates MapBox's context-based searching capabilities, better than OSM queries
- Extensive POI database, better than OSM database
- Storage of Data with OSM-provided attributes, more control over data
- Less dependency on Mapbox.
    - In case Mapbox changes their Mapbox ID or expires, users will still get the correct basic geographical data.
    - In case we migrate from Mapbox to another GIS provider, process will be smoother

Drawbacks:

- Complexity of data with using two sources alongside user-generated content.
    - Keeping three different IDs and maintaining synchronization
- Requires handling of cases where Mapbox POI information cannot be matched with OSM POIs
    - Since no attributes except Mapbox ID are stored, it requires API calls for every look-up.
    - Possible mitigation: for popular POIs, add POI to OSM ourselves and help contribute to OSM.

Internal POI Object with Mapbox and OSM

Our internal POI system will store POI objects in the following list of attributes (non-exhaustive):

POI Object		
Attribute	**Source**	**Description**
POI ID	**internal**	**identifier for POI**
Mapbox ID	Mapbox	Mapbox place identifier (limited use but storable)
OSM ID	OSM	OSM place identifier
Place Name		Display name in OSM
Latitude		Latitude in OSM
Longitude		Longitude in OSM
Description	AI generated	AI generated description of place
Photos	Wikipedia	Fetch from Wikipedia API
Other Attributes	User-Generated	Examples include: Reviews,

Flow of Data

User Looks-Up Place on a Map or Search Box

- Uses SearchBox API on Mapbox to return **Mapbox ID** and other POI information such as place name, image, etc

User Adds the Place to their Trip (Mapbox ID is not in our database)

- **Mapbox ID** is stored into a POI Object in our database.
- **If and only if the OSM-equivalent of the Place exists,** Match POI information from Mapbox API results to OSM and fill in:
    - OSM ID from OSM
    - Place Name from OSM
    - Latitude and Longitude from OSM
    - Other useful information

- POI Object is uniquely identified by **our POI ID**.
- **A list of POI IDs are stored for User Trips.**

User Views a Trip with the Place they Added (e.g. Itinerary/Timeline or Map)

- Identify Places (POIs or POI Objects) by **our POI ID**, not **Mapbox ID**.
- Display data in our POI Object for basic information (Place Name, Latitude, Longitude) of the place
    - If our POI object does not store OSM data (it couldn't find an equivalent OSM POI data), then fetch from Mapbox using **Mapbox ID**.
- When and only when the user wants the details of the place, using **Mapbox ID**, fetch Mapbox data (Phone #, Website, Operating Hours).

Compliance with Mapbox Terms of Service

Mapbox does not allow storage of API results except for Mapbox ID. The proposed solution only stores Mapbox ID from the API results.

Mapbox does not prohibit using its services with other third-party data sources. I couldn't find anything that says I can't use it with other services, unlike Google Maps or HERE.

Both Mapbox and OSM require attributions for its data. It must be clear where a specific piece of data comes from.