# CODE TO CONNECT

## SPONSORED BY GOOGLE MAPS PLATFORM

- Team Name: IndiOne
- Team Size: 3
- Google Maps Platform Account ID: siwoo.jg@gmail.com
- Leader email: 200134kms@gmail.com
- Github: https://github.com/min-1870/IndiOne/

# A MOBILE-FRIENDLY WEB APPLICATION THAT AUTOMATICALLY SCHEDULES ITINERARY BASED ON USER PREFERENCES

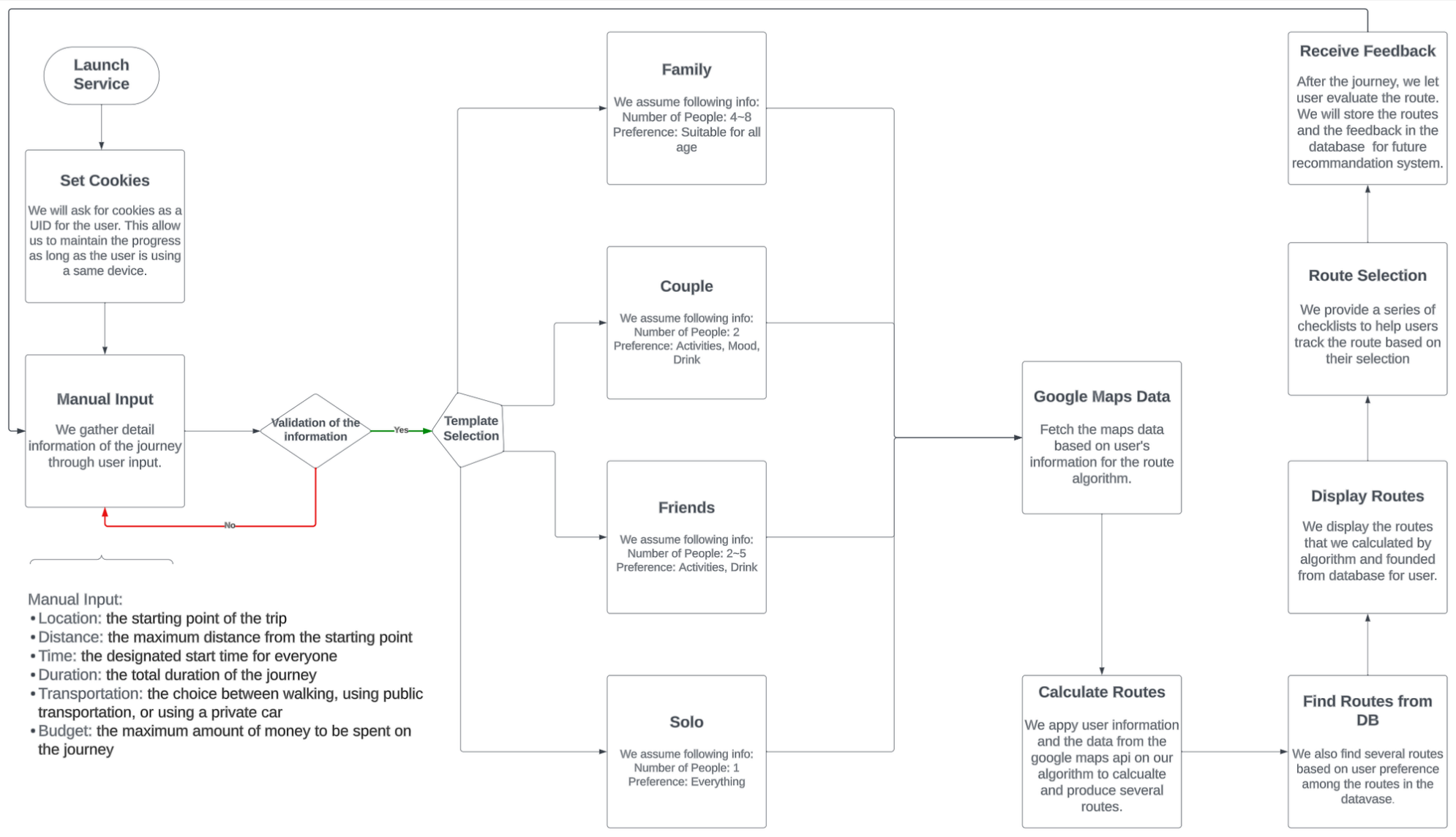
# PROBLEM STATEMENT

- As AI technologies have been developed and made available to the public over the last decade, the demand for a convenient lifestyle has become a growing global issue.
- We organized our ideas and devised a plan to address this trend. As a result, we have created an application with the function of recommending multiple routes for customers with various purposes (dating, leisure trips and business, etc.)
- Selectable options are integrated into the system to allow users to customize their plans accurately, according to their preferences.

# FEATURES

- **Automatic recommendation of travel itinerary**
- **Customization of travelling preferences by a user**
- **Four different themes of recommended itinerary**
- **Shows overall time schedule**
- **Shows information of recommended places**

# UML DIAGRAM


Launch Service
Set Cookies
We will ask for cookies as a UID for the user. This allow us to maintain the progress as long as the user is using a same device.
Manual Input
We gather detail information of the journey through user input.
Validation of the information
Yes
No
Template Selection
Family
We assume following info:
Number of People: 4~8
Preference: Suitable for all age
Couple
We assume following info:
Number of People: 2
Preference: Activities, Mood, Drink
Friends
We assume following info:
Number of People: 2~5
Preference: Activities, Drink
Solo
We assume following info:
Number of People: 1
Preference: Everything
Google Maps Data
Fetch the maps data based on user's information for the route algorithm.
Calculate Routes
We appy user information and the data from the google maps api on our algorithm to calcualte and produce several routes.
Find Routes from DB
We also find several routes based on user preference among the routes in the datavase.
Display Routes
We display the routes that we calculated by algorithm and founded from database for user.
Route Selection
We provide a series of checklists to help users track the route based on their selection
Receive Feedback
After the journey, we let user evaluate the route. We will store the routes and the feedback in the database for future recommandation system.
Manual Input:
• Location: the starting point of the trip
• Distance: the maximum distance from the starting point
• Time: the designated start time for everyone
• Duration: the total duration of the journey
• Transportation: the choice between walking, using public transportation, or using a private car
• Budget: the maximum amount of money to be spent on the journey

# USED API

Backend

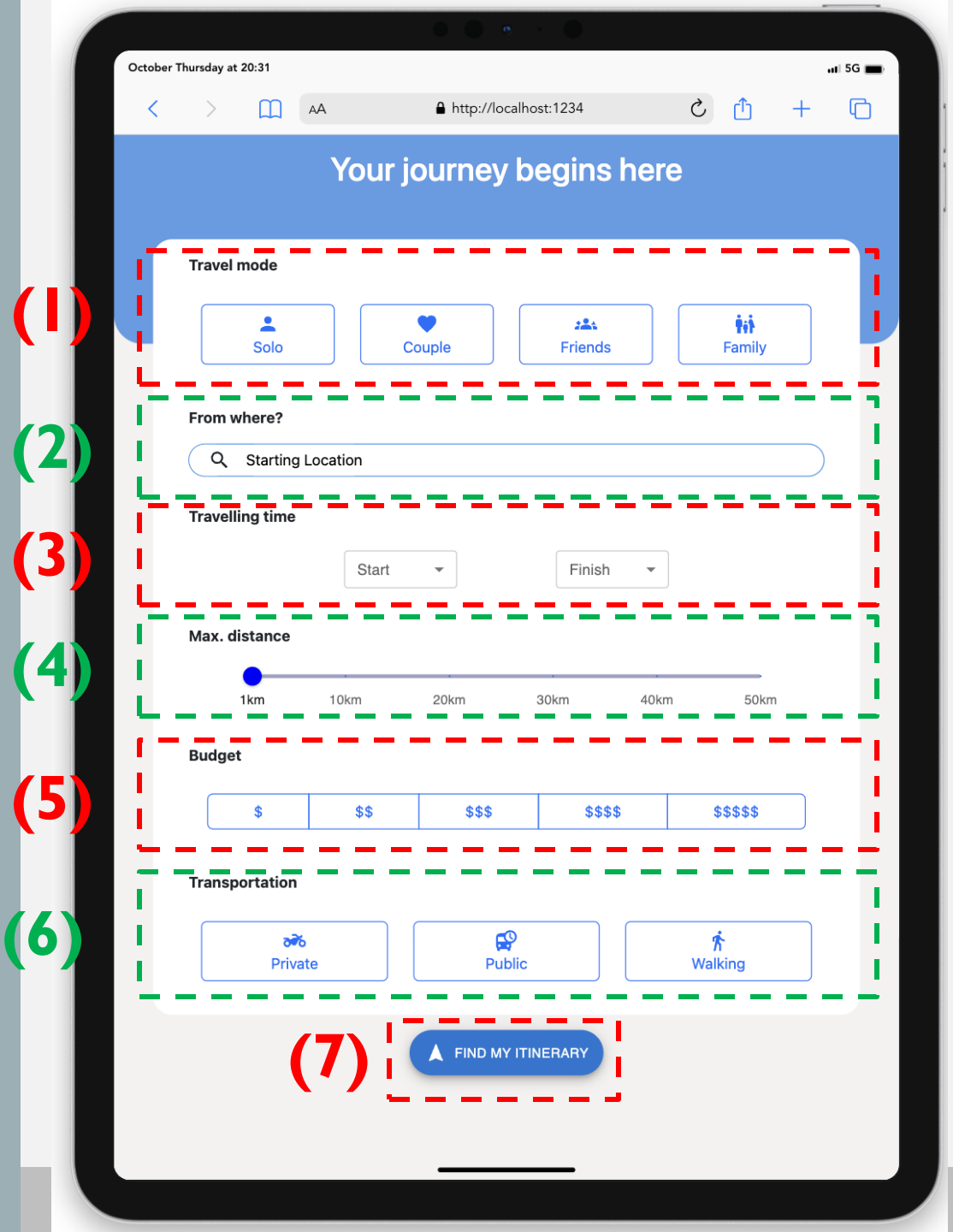
- Nearby Search (Places API)
  - We use this API to fetch the surrounding places based on user preference. We request at least 10 times and merge the result to create the route with diverse type of places.
- Place Detail (Places API)
  - We use this API to check the opening hours of each places in the route.

Frontend

- Maps JavaScript API
  - To render google maps to help a user to search for starting location, and to show users routes of itinerary
- Place Autocomplete (Places API)
  - To help users to search for addresses/places of their starting locations
- Geocoding API
  - To retrieve geographic locations of a user-input starting location
- Directions Service (Maps JavaScript API)
  - To find routes of recommended places

# PROTOTYPE – MAIN PAGE

1. A user can select whom they will travel with. It affects the selection algorithms of recommended places
2. A new swipeable edge pops up and a user can choose the starting location.
3. A user can choose starting and finishing time of travel
4. Maximum distance a user would like to travel
5. A level of available budget
6. Transportation method
7. Once tapped, it goes to the next page. In backend server, it begins calculations and searches for recommendation routes.



# PROTOTYPE – RESULT PAGE

1. Shows the current theme of the recommended itinerary, among casual, local specialty, qualitative and shortest
2. Google map showing recommended places and routes
3. Itinerary information card showing brief (3A) and detailed once opened (3B)
4. Navigator that a user can navigate different themes of recommended itinerary
5. Home icon to the main page

# ALGORITHM

STEP 1

We retrieve places based on the user's preferences using the Nearby Search API. We make multiple requests to the API based on our predefined place types for each template.

STEP 2

We generate 500 routes using the places from Step 1. Every route should have valid distances between places, types of places, and good ratings of places.

STEP 3

We sort the algorithm by different criteria for each category of the route. We priorities the 500 routes for validation in Step 4. This will help us avoid unnecessary API request.

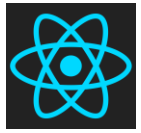
STEP 4

We request detailed opening times using the API and assess the feasibility of the route by comparing the opening times with the visiting times of the places on the route.

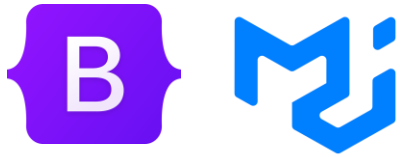
# TECH STACK

- Frontend
  - UI


  - Frontend framework



  - Styling




  - Web application framework


- Backend
  - Backend framework


  - Database

# DEMO

https://www.youtube.com/watch?v=VnM7GrvGW9M&ab_channel=SiwooJung

# FUTURE WORK

**Improved route recommending algorithms**

- Apply machine-learning for users' route choices and apply it for route recommendation algorithms
- A feature that a user can choose unwanted places among recommended routes, and those unwanted places to be replaced by other places while maintaining the rest of the places

**Handling user database**

- Sign-up, login, logout features
- Saved user preferences

**Overall improved UI**

- More detailed description of each selected places
  - Sample photos
  - Link to the official site
- Customization of Google Map - simple, yet informative

# THANKYOU