Journey Jotter: Project Specification

Our application is designed to assist individuals or groups when planning events, specifically travel itineraries.

Product Backlog

Actions

- **Register/Login:** every user has to register or login with Google OAuth in order to see the rest of the webpage
- **View Trips**: the main page upon logging in is a page with a list of each trip you have created/are a part of with links to each individual trip page
- **Add Trip:** you can click a button on the view trips page to create a new trip
- **Click on a trip**: if you click on a trip from the main page, you can view the individual trip page with information such as the liked flights, liked activities, the calendar, and the map
- **Invite people**: you can invite users to your trip
- **Add flight or activity to map**: on the trip page, you can view liked activity and flights and click a button to view them on the map
- **Click on flights page**: on the trip page, you can click on the flights page in order to search for flights and modify liked flights
- **Click on activities page:** on the trip page, you can click on the activities page in order to search for and modify likes for activities, restaurants, and accomodations

Ideas

- **Dynamic Planning**: utilize real-time prices to provide costs that reflect current market prices. Information is refreshed and up-to-date on all pages by refreshing the APIs used on each page

- **Calendar Interface:** We will implement a calendar so users can add and edit travel dates. This will allow us to fetch data accordingly to reflect the prices of the given dates.

- **Cost-Conscious Planning:** Users can select from many different flights/activities. They can create different itineraries that can be compared side-by-side to assist with cost and travel time considerations.
    - Using the Gemini API, users can easily search different activities at different price points. On the same page as the Gemini API interface, users can annotate different activities, leading to this comparison feature that will enhance user experience.

- **Authentication and Authorization:** Users will be able to register accounts and log in using our application as well as through a third-party service (Google OAuth). Google OAuth simplifies the sign-in process and leverages Google's security.

- **Collaboration:** Authenticated users can create and join groups. Among these groups, users can view the itinerary, reservations, etc. Users can also vote on preferred activities, restaurants, hotels, etc. using the “like” button.
    - Google OAuth can be used to fetch the email addresses of potential collaborators to send invitations.
    - AJAX will send requests to the server to record votes on activities

- **Maps:** Users can see their travel destinations, reservations, and trajectories on an interactive map. The map will dynamically update to show these selections.

  Technologies Used: *django-google-maps package*, AJAX

1. Could given length of activity schedule it into the calendar? - this would probably go under either calendar or an agenda

1st Sprint Backlog

Product Owner: Linsey Szabo (lszabo)

Each of us will tackle connecting our webpage to an API

1. Implement Google OAuth (Carolyn Yang): By the end of this sprint, users will be able to log in and register accounts using their Google accounts via OAuth.

   Tasks Completed:
   - Set up Google Cloud Console for Journey Jotter
   - Use Client ID and Client Secret to integrate Google OAuth into Django
   - Configure Django settings to include Google Oauth credentials and redirect URIs
   - Update the Login and Register pages

2. Implement ChatGPT API (Linsey Szabo): By the end of this sprint, users will be able to query ChatGPT API from a webpage.

   Tasks Completed:
   - After some research, I learned that ChatGPT API isn’t free. At the suggestion of a teammate, I turned to the Gemini API to accomplish a similar functionality.
   - I installed the API and generated an API key.
   - Using that key, I was able to test my ability to query the API locally.
   - I added a simple HTML page to query Gemini, one query at a time.
3. Implement Google Maps API (Corrado Govea): By the end of this sprint, users will be able to see an interactive Google map with one location geotag.

   Tasks Completed:
   - Created an account with a new project, and acquired an API key.

- Created a new HTML page to display the interactive map, with its corresponding URLs and navigation links from other pages.
- Wrote relevant models needed for map implementation once other features were added.

4. Research Google Calendar API Requirements (Corrado Govea): Pending working OAuth to include Google Calendar API.
5. Implement Google Flights API (Joshna Iyengar): By the end of this sprint, users will be able to search for flights with the required parameters of departure airport, departure date, arrival airport, and arrival date, and few the possible flights

   Tasks Completed:
   - I found out that using AJAX through Django gives a CORS no header error, and the only solution would be to create a server-side API and talk to the Google flights api through that so I pivoted to searching in the Python views.py function
   - Created inputs for people to enter the required information in order to search for a flight
   - Handled errors if people didn't select anything for the required information
   - Searched for flights using Google Flights API in Python: https://serpapi.com/google-flights-api
   - Parsed JSON to return the possible flights for people to choose from and present in the HTML

Django Models

Flight:

- ID (For. Key)
- Departure Airport (select text)
- Arrival Airport (select text)
- Departure Date (date)
- Arrival Date (date)

Activity:

- Destination (For. Key)
- Name (text)
- Address (text)
- Summary

Destination:

- Arrival (datetime)
- Departure (datetime)
- Activities (one-to-many)
- Stays (one-to-many)

Stays:

- Destination (For. Key)
- Name (text)
- Address (text)
- Check-in & Check-out (time)
- Price (Number)
- Reservation (text / link)

Activities:

- Destination (For. Key)
- Name (text)
- Address (text)
- Summary

Profile Model

Will utilize information extracted from Google Oauth service

- Location (text)
- User (one to one)
- Profile Picture (url field)
- Timezone (text)
- Email (text)

2nd Sprint Backlog

Product Owner: Carolyn Yang

Each of us will build the functionalities we tested in the 1st Sprint into the mockups we presented

1. Continue Google OAuth Implementation and Collaborative Features (Carolyn Yang): By the end of the 2nd sprint, our application with implement models and collaborative features

   Tasks to Complete:
   - Use access token, CLIENT_ID, and CLIENT_SECRET to request user information and integrate that with the profile model.
   - Set up an invitation system that will utilize Google email to invite other collaborators.
   - Enable users to join groups, share itineraries, and modify shared content collaboratively
2. Continue Google Flights Implementation and integration (Joshna Iyengar | jiyengar)

Tasks to complete:

- creating inputs for optional parameters (there are 27 so I wasn't sure the best way to do this)
- Connecting the flight model and forms
- handle date not being in the future
- Handle other errors that google flights API might return in general
- format the flight information nicely
- add a like button for each flight on the flight page
- Show liked flights on the travel itinerary page
- Add an 'add to map' button for each flight on the travel itinerary page

3. Continue Google Maps API (Corrado Govea): By the end of this sprint, users will be able to see an interactive Google map with all chosen destinations and activities

   Tasks to complete:

   - Integrate django models to create geotags for destination cities, activity locations, hotel stays.
   - Generate recommended routes for a given day based on the activities on schedule.

4. Continue Google Calendar API integration (Corrado Govea): By the end of this sprint, users will be able to see their selected activities, reservations, and flights on a shared google calendar.

   Tasks to complete:

   - Integrate calendar API using OAuth.
   - Enable creation of shared calendars from within the django application.
   - Display event details when an event is selected from the calendar.

5. Continue Gemini API integration (Linsey Szabo): By the end of this sprint, users will be able to interact with Gemini in a similar way as on the website and take notes on activities, restaurants, and accommodations they like.

   Tasks to complete:

   - Build out Gemini API interface, allowing older questions and answers to be displayed and also cleared.
   - Add activities display.
   - Add restaurants display.
   - Add accommodations display.

Views

Login to the Journey Jotter:

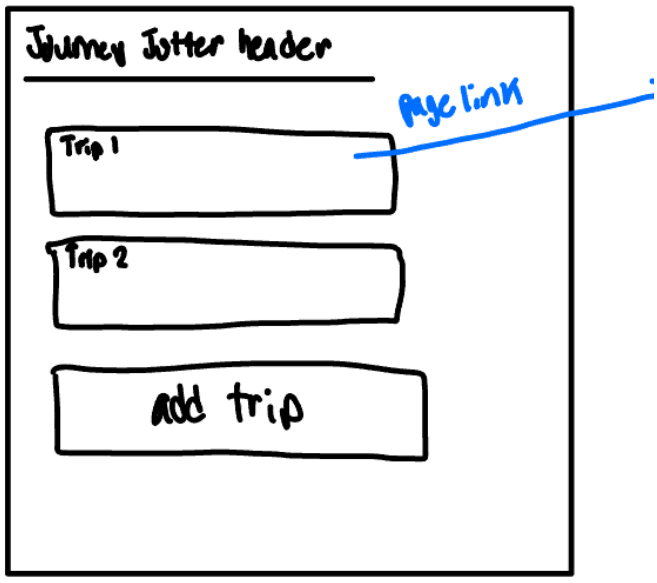
Username:

Password:

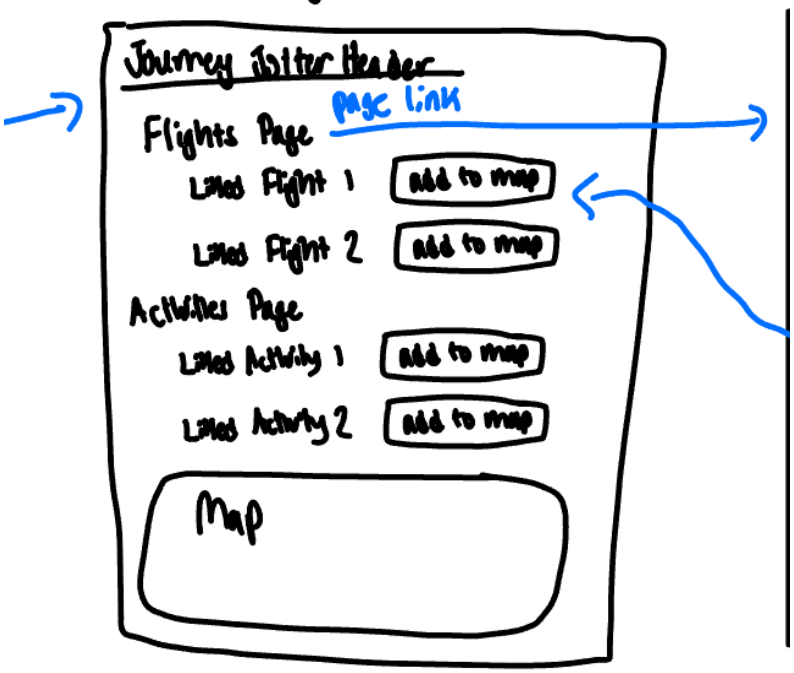
Submit

Sign in with Google

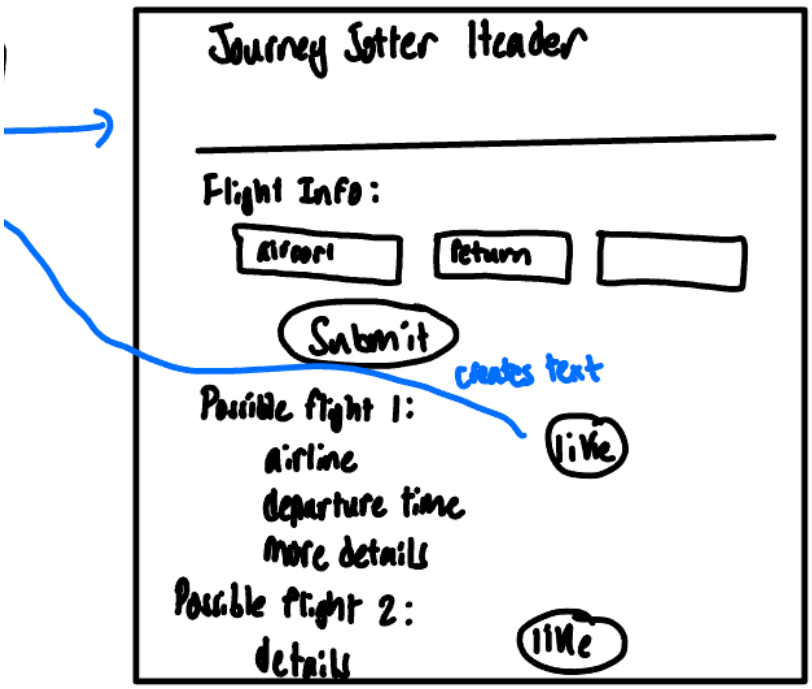
My Trips page



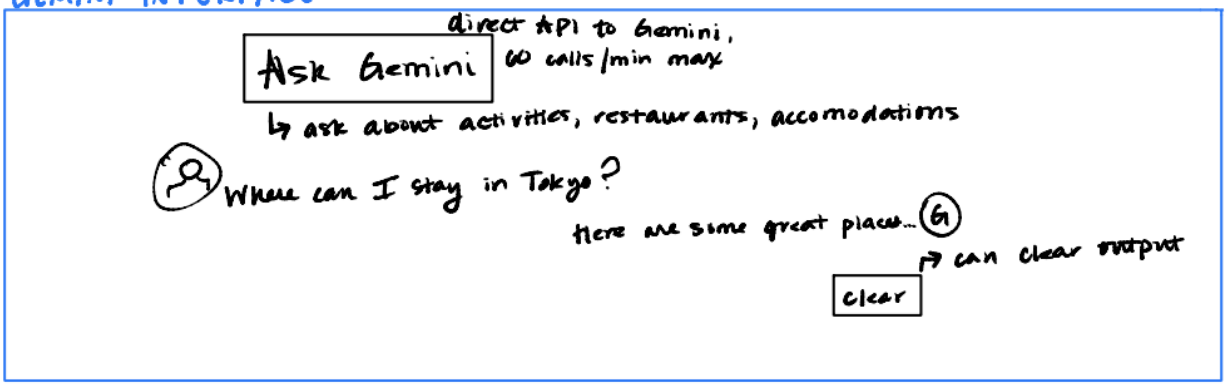
Trip Page



Flights Page



GEMINI INTERFACE



NOTES MADE BY USER

↳ each field inputted by user based on Gemini output

Activities

- Activity 1
  - description (like) ↳ can see if other members of trip like this
  - location [remove activity]
  - price [add to trip] ↳ add to trip page
  - ⋮
  - [add activity] ↳ add another activity you're interested in

Restaurants

- Restaurant 1
  - description (like)
  - location [remove restaurant]
  - $ - $$$$ [add to trip]
  - ⋮ [add restaurant]

Accomodations

- Accomodation 1 (like)
  - location [remove accomodation]
  - price [add to trip]
  - ⋮ [add accomodation]

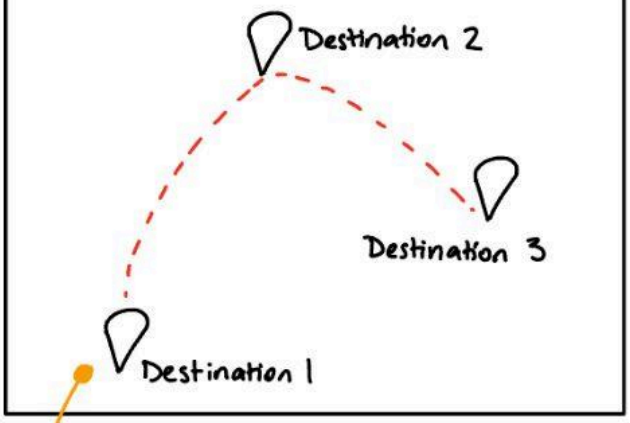
Itinerary

Destination 1

Jan 1st - Feb 3rd, 2024

Destination 2

Des...



Destination Details

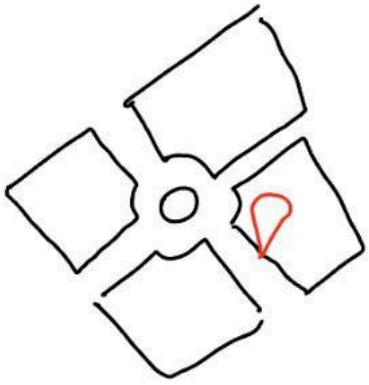
Destination 1

Stays

Activities

Activity 1

Activity 2



Activity 1

Summary....

Date:

Time:

Address:

Trip Schedule