# Flight Booking Chatbot

This is a chatbot that is used to provide accurate flight prices based on information provided by the user.

* Tientcheu Ngaleu Larry Jordan
* Kameni Kameni Olivier
* Ej-Jaidi Aymane

# Project Structure

The project consists of two main parts. A backend and a Frontend.

## Backend:

The backend is the backbone of this project. The backend consists of the AI model used to predict flight prices and an API used to interact with this model and save the resulting prompts into a database.

### AI Model:

In order to build the AI Model, we went through three major processes. These processes are well defined each in their respective .ipynb files. These files are:

* Flight\_prices\_prediction.ipynb:
  + This file contains all the data wrangling work we have done. It is here that we perform data assessment, data cleaning and data visualisations in order to get valuable insights.
* Model.ipynb
  + This is where we train our AI\_model on the preprocessed data obtained from the previous file. Here, we train 3 models and based on the results, we chose the best of 3.
* Model\_eval.ipynb
  + This is a notebook where we test our model to make sure everything works as expected.

#### Flight\_prices\_prediction.ipynb:

The goal of this section is to generate flight prices based on information provided in the flight.csv dataset. Here, we wrangle and analyze the dataset. At the end of this, we provide a new dataset with accurate flight prices based on various factors.

* We start with data assessment. In this section, we check the dataset for quality issues and tidiness issues. We also use this section to get a global feel of the dataset. After the assessment, we were able to make some observations:
  + Our assessment showed that the dataset contains only flights for the year 2015.
  + This dataset contains flight information only for the US territory(Domestic flights).
  + We identified information that is needed in our process. i.e (Unique airline identification number, Date of flight, Origin airport, Destination Airport, Flight Distance, Scheduled departure)
* After the data assessment section, we continued with data cleaning. Here we cleaned the dataset by removing the unnecessary information for our analysis and keeping only what we needed. We started this section by making a copy of the data so as not to inadvertently modify the original dataset.
  + We formatted the scheduled departure time and the different dates in a single entity. The new format for the scheduled departure date is now YYYY-MM-DD HH:mm:ss. This proved to be better because when booking a flight, we always have a date and a time.
  + The original dataset contained distances in miles. We proceeded to convert it into kilometers.
  + In order to calculate the cost of each flight;
    - We get the average price of the origin airports multiplied by the price per km due to the length of the flight. The price per kilometer is obtained through a google search. This is the mean of the lowest price and highest price 0.15 and 0.40 for domestic us flights.
    - The flight price varies according to the time of the year. To rank the months for booking domestic flights within the United States in ascending order from the best (most affordable) to the worst (most expensive), here is my recommendation: `**January, February, September, October, November, March, August, April, May, June,July, December**`. The months with the lowest fares for domestic flights in the U.S. are typically January, February, and September, which fall during the off-peak travel season. These months offer the best deals as demand for air travel is lower. October and November can also be relatively affordable months for booking domestic flights, as they fall just after the peak summer travel season. March, August, and April are considered shoulder months, with fares being moderate compared to the peak and off-peak seasons. The most expensive months for domestic flights are typically May, June, July, and December, which coincide with peak travel periods for summer vacations, holidays, and school breaks. During these months, demand for air travel is highest, and airlines can charge premium prices.
  + The average price of flights is obtained from the [transtat website](https://www.transtats.bts.gov/AverageFare/). For each origin airport, the cost of the flight is the average between the inflation adjusted fare and the price per kilometer of that particular flight.
* At the end of this process, a new dataset called flight\_prices is produced.

#### Model.ipynb:

Here we train 3 models; Linear Regression, Random Forest, Neural Network. Between these 3 models, we choose the best and that is what we use for the price prediction with our chatbot. At the end of this process, we decided to go with the neural network, because it had the best performance. The model is called nn\_model.pkl saved in pickle format in the file tmodels.

#### Model\_eval.ipynb:

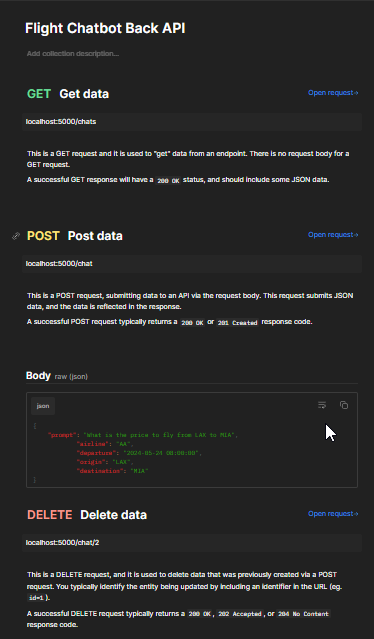
In this notebook, we evaluate our trained model, the neural network model. This notebook tests the model on a single custom input to make sure everything works well.

Ai\_model here is complemented with some helper function in the file helpers/helpers.py. These functions are used to interact with the ai model.

### Backend API

This is a Flask API that provides routes to interact with a model by providing prompts and returning a response. This is the way we have decided to interact with the ai model. This API is made of notable features like:

* The api is built using Flask: A python web development framework.
* PostgreSQL as the database management system.

The code can be found in the backend folder. Inside program.py. This api takes into account database migration and uses SQLAlchemy as Object Relational Mapper. The Api consists of 3 routes:

The response for the post request is a JSON like this:

{

    "price": "660.51324",

    "prompt": {

        "airline": "AA",

        "destination\_airport": "MIA",

        "id": 8,

        "origin\_airport": "LAX",

        "prompt": "I want to fly from LAX to MIA on 24 May at 8 am",

        "scheduled\_departure": "Fri, 24 May 2024 08:00:00 GMT"

    },

    "success": true

}

The loading of the model for the prediction is cached. The actual loading is only done the first time the request is processed.

* Precise instructions on how to run the backend api can be found in the Readme.md file.

## Frontend:

# How to run