Project Proposal: Cheap Flight Finder

Group 9
Hao Qiu

Introduction:
The travel industry has seen significant growth in recent years, with more people traveling for both personal and business reasons. One of the most significant expenses of traveling is the cost of flights, making it crucial for travelers to find the most affordable options. The purpose of this project is to create a platform that allows users to find the cheapest flights available, taking into consideration their preferred travel dates and destination.

Project Goals:
The primary goal of this project is to provide a user-friendly platform that allows travelers to find the cheapest flight options quickly and efficiently. The platform should be able to search multiple airlines, compare prices, and provide users with a range of options that fit their budget and travel needs.

Features:

Flight search: The platform will allow users to search for flights based on their preferred destination, departure date, and return date.

Price comparison: The platform will compare prices from multiple airlines, including budget and full-service carriers, to provide users with the best available options.

Itinerary customization: The platform will allow users to customize their itinerary by adding or removing stops, selecting specific airlines, and adjusting the departure and return dates.

Price tracking: The platform will provide users with the option to track the prices of their preferred flights, alerting them if the price changes or if a better option becomes available.

*User account: Users will have the option to create an account, where they can save their travel preferences, flight itineraries, and receive price alerts.

Technologies:

Backend:
We will use python to design a program that can read a dataset and user input and return the final cheap flight information.
Use python to design a web crawler to crawl relevant flight data from the booking website and store it as a dataset.

*Frontend: The platform will use React.js to develop the user interface, making it responsive and easy to use on any device.

*Database: The platform will use MongoDB to store user data and flight information.

Timeline:

Planning and Research (week 2-3): This phase will involve conducting market research, defining the project scope, and creating a detailed project plan.

Development (week 4-8): This phase will involve developing the backend services, integrating them with the database, and testing the platform for any bugs or issues.

Testing and Deployment (week 9-10): This phase will involve conducting thorough testing of the platform, fixing any issues found, and deploying the platform for users.

Maintenance (Ongoing): This phase will involve providing ongoing support and maintenance for the platform, ensuring that it continues to meet the needs- of users and provide them with the best possible experience.

Conclusion:
The Cheap Flight Finder project is an ambitious but achievable project that will provide travelers with a user-friendly platform to find the best flight options that fit their budget and travel needs. With its combination of powerful technologies and a user-focused design, the platform is poised to become a leading player in the travel industry.--

(*Ideally, it may not be implemented in this project)

Project Update:

Dataset：
This dataset is a CSV file where each row is a purchasable ticket found on Expedia between 2022-04-16 and 2022-10-05, to/from the following airports: ATL, DFW, DEN, ORD, LAX, CLT, MIA, JFK, EWR, SFO, DTW, BOS, PHL, LGA, IAD, OAK.

(Source: https://www.kaggle.com/datasets/dilwong/flightprices)

Update(week4):
1. Cleaning this dataset, choose important information used by program. (Airline, Country, Start and Ended destination, date, price, etc). Done
2. Design the python program to obtain user's input as parameter pass through the search function. Done
3. Design the search functions can use a few of parameters and return the result from cleaned dataset.
4. Design a python web crawler that can crawl the flight information of the specified ticket website and store it in .csv format and return it.
5. Using the data from step 4 and combine the step 3 to finish the whole search process.
6. Do the testing of the application.
7. Design the simple UI to show the application.

Update(week8):
1. Cleaning this dataset, choose important information used by program. (Airline, Country, Start and Ended destination, date, price, etc). Done
2. Design the python program to obtain user's input as parameter pass through the search function. Done
3. Design the search functions can use a few of parameters and return the result from cleaned dataset. Done
4. Design a python web crawler that can crawl the flight information of the specified ticket website and store it in .csv format and return it. (Alternative way: Use the airline or ticket 's API to obtain the air ticket information) Fixing bug and consideration alternative way
5. Using the data from step 4 and combine the step 3 to finish the whole search process. Not yet
6. Do the testing of the application. Not yet
7. Design the simple UI to show the application. Not yet

Update(week10):
1. Cleaning this dataset, choose important information used by program. (Airline, Country, Start and Ended destination, date, price, etc). Done
2. Design the python program to obtain user's input as parameter pass through the search function. Done
3. Design the search functions can use a few of parameters and return the result from cleaned dataset. Done
4. Design a python web crawler that can crawl the flight information of the specified ticket

website and store it in .csv format and return it. (Alternative way: Use the airline or ticket 's API to obtain the air ticket information) Fixing bug and consideration alternative way

5. I changed the mind to get the flight data by web crawler to API, I can find a API and connect it to get the flight data.
6. Using the data from step 4 and combine the step 3 to finish the whole search process. Done
7. Do the testing of the application. Done
8. Design the simple UI to show the application. Done
9. Packaging the flasks app and deploy to aws eb, and make it run on the web. Done