Bilkent University

Department of Computer Engineering

Senior Design Project

Project short-name: AeroCast

Project Specifications Report

Ahmet Alparslan Çelik, Ömer Berk Uçar, Muhammed Yusuf Satıcı, Yasin İlkağan Tepeli, Ahmet Taha Albayrak

Supervisor: Çiğdem Gündüz Demir
Jury Members: Selim Aksoy and Mustafa Özdal

Progress Report

Oct 9, 2017

1. Introduction

The purpose of the project is to provide a forecasting service to different kinds of business areas by using time series data. Since the project will require to process specific data for making predictions on different industries, designing a system that can achieve this purpose needs to be built in multiple phases. Therefore, the project aims to divide the process of building the final forecasting service into three phases in which the complexity of the technical infrastructure and the variety of data will increase in each phase. The first phase will consist of developing a forecasting system for commercial flights. In the next two phases, the project will be extended to the real-estate and e-commerce industries. In the senior project, only the first phase will be completed. The rest of the introduction explains the motivation behind aiming for the commercial flight industry.

While planning a holiday or a business trip, everyone once in a while tries to find the best airfare deal according to their needs. In general, it is a tedious work due to the dynamic changes in prices. Airlines change their prices according to various criteria such as supply and demand, how far the departure date is, which day of the week the flight is, etc. Thus, finding the cheapest ticket by searching manually can be cumbersome. Furthermore, it is necessary to optimally find the best purchase date for a particular route within the travellers' budget.

With the current technologies, it is possible to track airfares, get notifications in case of a drop or increase in price and even get rough predictions if the price will increase or decrease for a particular route. Nevertheless, the existing methods do not provide user specific airplane predictions. Therefore, in order to purchase the ideal cheapest ticket, travellers will need to use separate services. Moreover, if they were to look for different travelling plans, the number of different routes they need to check will increase making things even more time consuming and complicated.

Aerocast aims to minimize the efforts put into finding the best airfare deal available. It will be a web service, which collects the available data scattered through different web services and analyzes them in a user specific way. Therefore, it will assist travellers to find the best airfare in the minimum possible time.

In the report, a description of Aerocast is provided. Then, it is followed by the constraints regarding the economy, time, implementation, ethics and sustainability. Afterwards, potential professional and ethical issues are brought forward. Finally, the functional and nonfunctional requirements of Aerocast are provided in detail.

1.1. Description

Think of a scenario where a traveller plans to purchase cheap plane tickets and yet have a decent flight experience. They can check services like Skyscanner to list airfares for a specific departure date. However, they might not know when exactly they want to fly so they need to look for different departure dates. Also, they can add price alerts to the flights that suit their program but the price alert service does not say anything about the purchase time that the ticket is the cheapest. Thus, they need to use another service named Hopper to get suggestions on whether it is better to wait for a while or to buy the ticket now. However, Hopper does not offer any date, time and flight flexibility in its suggestions. That is to say, travellers cannot select range of dates for the departure and they cannot define time of the day or any flight preferences. Also the notifications sent by Hopper alert its users several time of the day with every minor change in the price and there is no way to change the intensity of the alerts. Hence, the travellers have to consult again to another application like Skyscanner to list and filter the tickets after they get notification about their flights from Hopper. Finally, if travellers have not yet decided on the specific day of the travel, they have to do all of these things for other alternative flight dates as well. As seen from above, coming up with a decent flight to travel can take a lot of effort and an application that provides an effective medium to track airfares is essential.

On the side of Aerocast, given outbound and inbound location, range of time and dates, Aerocast predicts the most suitable purchase and departure dates of the flights based on the traveller’s time and budget constraints and it shows statistical data about the flights. It will regularly collect the required information about the flights from services like

Travelpayouts, Make My Trip and Google QPX Express and it will collect the restrictive personal data by asking for preferences to the users. It will then analyze the airfare history and predict the best range of days in near future to buy the ticket. Aerocast will also visualize history of price fluctuation on a graph, where users can see prices in the past for their choice of flight. Hence, they can get better insights about their flights.

Overall, Aerocast will provide extensive assistance to travellers for their flight booking process so its users will never need to switch between different services for the information they need while using the most trusted services Travelpayouts, Make My Trip and Google QPX Express to find the tickets.

1.2. Constraints

1.2.1. Economic Constraints

- Hosting service is needed to deploy this project and the payment for hosting is 20 USD per month.
- Domain name payment for the website is 5 USD per year.
- All framework and libraries to be used will be available and free for commercial-use.
- Travelpayouts, MakeMyTrip and QPX Express API will be used to fetch flight data. Using QPX Express API will cost 0,035 USD per query above 50 query.

1.2.2. Time Constraints

- The first version of this project will be launched in 8 months.
- The system needs to start collecting the data for each flight twelve month before the departure date of that particular flight.
- The system needs to gather at least 10 days of flight data for each flight before displaying the statistics graphically regarding that particular flight.

1.2.3. Implementation Constraints

- The software will have server-client architecture.

- Git version control system together with GitHub will be used to manage project by the development team.
- OOP paradigm with the basics of SOLID [*] principles of will be followed.
- The web platform is required to be responsive and the targeted web platforms are Google Chrome, Opera, Firefox, Safari and Edge.

1.2.4. Ethical Constraints

- The Software will not share users' data with third-parties in order to protect the user privacy.
- The data we collected from web services will not be sold to any other company for any kind of profit.
- We will abide by the IEEE Code of Ethics in the development of the project [**].
- Users are considered to accept the Software as a Service agreement when they use the service.

1.2.5. Sustainability Constraints

- Accuracy of randomly selected predictions will be checked on a daily basis and in case of a drop in accuracy, the admins of the system will be notified.
- There will be advertisements to compensate the spendings.
- The users may send feedback to the program.

1.3. Professional and Ethical Issues

First of all, the data gathering is a professional issue for us and the programs that we want to collect data from. Since we do not have sufficient data for the system to function properly, we need to gather data from various sources and we need to abide by the professional and ethical criteria of those sources. Also, being a student and not having a company to make people trust you have some drawbacks such as no one shares the data with you. While it is a professional issue from our side, it is also an issue for the company which we want to collect data from since we will send enormous number of daily requests. Since the number of requests that we need to send to the websites and companies is extremely high, the high number of request might slow down their systems and cause them to reject our requests.

In addition to that, the system might have professional impacts on the airline companies such that it can change the people’s habit of buying ticket and it may damage their policies and future plans. It may also cost them a lot of money since people can use the program to benefit from the promotions and the events of the airlines.

There is always an ethical issue, if there is a membership feature. The issue is that the people with accounts will have password information that will be saved into database. One issue is that their password will be available in our database, even though it will not be used by anyone. The second issue is that when they signup they have to read Privacy Policy, Content Policy and agree to our terms. It provides that they allow us to collect their data and store it in our database.

2. Requirements

2.1. Functional Requirements

2.1.1. User Profile

- The user should be able to register into the system by providing password and email address information.
- The email address should be unique for each user.
- The user should be able to sign in into the system by using email and password information of his previously created user account.
- The user should be able to sign in by using his facebook or gmail accounts.
- After signing in, the user should be able to update his profile information.
- The user should be able to receive notifications from the system and should be able to view or remove the notifications from his profile.
- The user should be able to change the email notification settings of the system from his profile.
- The system may support SMS notifications for the future development of the project.

2.1.2. Time Prediction

- The user must be able to send requests to the system by specifying the origin, destination and departure dates as well as the number of adult and child passengers information.
- The user must be able to choose one way or return flight tickets in his request.
- The user should be able to specify his flight type as direct, indirect or non-stop.
- The user should be able to specify the departure hour intervals as morning, afternoon and evening.
- The user should be able to choose cabin class and carrier code of the flight.
- The carrier code, departure hour interval, cabin class and flight type information is optional.
- The system should be able to return responses for the departure and purchase date forecasts based on the criteria specified in the request.
- The user should be able to filter the responses according to the departure hour interval, the number of stops, cabin class and carrier code.
- The user should be able to display the responses in different orders based on the prices of the flights and the duration of the flights.

2.1.2.1. Purchase Time Estimation

- The system should estimate the purchase dates with minimum prices and the expected values of the prices.
- The system should return purchase date responses for each departure date which the flight ticket prices are predicted to be the lowest.
- The system should return the expected price and other available flight information as a part of the response.
- The user should be able to track the purchase date responses of the system.

2.1.2.2. Departure Time Estimation

- The system should be able to rank the departure dates as having low, medium or high prices based on their price intervals and stability of their prices.

- The system should display the departure dates of the flights on a colored graph in which the blue lines represent the departure dates with low and stable prices whereas the red lines represent the departure dates with high and unstable prices.

2.1.3. Route and Flight Tracking

- The user should be able to track the routes and flights only if they log in.
- The user should be able to display his saved routes from his profile.
- After selecting the saved route, the user should be able to display his favorite tracked flights on that particular route.
- The system should keep updating the tracked searches and flights as the new data arrives.
- When the departure date of a flight expires, the system should remove the route from tracked routes.
- When the estimated dates for a saved request expire, the system should update the response with a new estimate.
- The system should notify the user about the changes in the tracked responses.

2.1.4. Flight Statistics

- The users should be able to display the flight statistics and graphs for their routes and flights.
- By selecting the flight, the user should be able to display the past changes in the prices of the flights as a solid line on a graph along with the predicted prices of the flight as a dashed line on the same graph.
- The user should be able to display the exact price and date values on the graph by moving the cursor on top of a particular point on the line.

2.1.5. Data

- The system should store the profile information and the saved routes and flights of the user.
- The system should keep the data of the previous flights such as price, departure date and time, origin, destination, flight type(direct or transit), carrier code, cabin class.
- The system should store all airport names, codes and cities in the world.

- The system should search and suggest the destination and origin cities as well as the airport names as the user types.

2.1.6. Automatic Data Update

- The system should collect 600,000 flight data from the most frequently used 10,000 flight routes worldwide daily.
- The system should update its price analysis based on the collected data.
- The system should regularly check the results of the saved routes and flights and update these responses based on the changes in the price analysis.

2.1.7. Feedback

- The user should be able to send feedbacks to the system in order to evaluate the application.

2.2. Nonfunctional Requirements

2.2.1. Availability

- The system should provide service continuously other than the system maintenance hours.
- The system maintenance should be scheduled between 20.00 - 22.00 on the first Monday of each month.

2.2.2. Maintainability

- Since the modules are working independently, the addition of new functionalities and the changes in the existing functionalities should not affect the system.
- A crashed component should not affect other components.

2.2.3. Response Time

- The system should predict the purchase and departure dates of the new request and list them at most 5 seconds.
- Pages should be loaded under two second.

2.2.4. Scalability

- The system should fulfill 10,000 users requests daily.

- The system should be able to store at least 10 GB of NoSQL data for the first year which continuously increase 100 MB in size for each day.

2.2.5. Security

- The system should store passwords after hashing with SHA-256.
- The number of requests will be limited to 200 per IP address in order to protect the flight data from malicious actions.
- The web service should be protected from DDOS attacks using the Cloudflare service.

2.2.6. Reliability

- The system should accurately collect data of the flights on a daily basis and accurately predict the departure and purchase dates of the flights based on the collected data.

2.2.7. Usability

- The interface should be user-friendly and the data is neatly shown.
- The system should display graphs smoothly and continuously.

2.2.8. Portability

- The system should be run on desktop web platforms and mobile web platforms.

2.2.9. Extensibility

- The system should allow developers to add new components and services on it.

3. References

[**] "IEE Code of Ethics". http://www.ieee.org/about/corporate/governance/p7-8.html.

[Accessed: Oct 10, 2016].