Flights Assistant Chatbot

using WeChat

Team Member:

Gong Yifei (A0198495E)
Jiang Yanni (A0201097M)
Ng Siew Pheng (A0198525R)

Table of Content

1. Introduction

According to IATA, airlines carried 4.1 billion passengers by air in 2017. Before booking a flight, passengers would have to spend time searching through various platforms to find the best flight deals. That is a lot of time spent searching for flight deals. The search for flight becomes a repetitive task that a passenger would need to perform over a period of time before they find the flight they want.

With the above, we see a potential to develop a service that helps passenger monitors the flight deals on their behalf and send them updates on deals, until the request ended.

We have developed a chatbot that user can interact. We are using WeChat app to build our front end-interface to users. This decision is driven by the number of WeChat users worldwide and the exposure to both Android and iOS users, as compared to Google assistant.

Users can either send us a voice recording with the type flight request or they text us with the details of their flight request. Users can also specified the number of days that they would like us to monitor the request for.

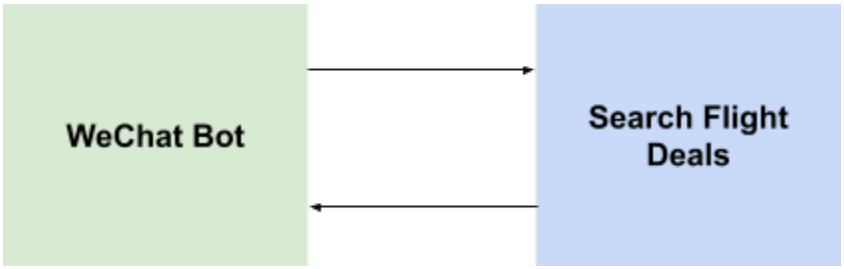
With the above, our system will perform daily search and send the results to the users' WeChat account. Users can also request to extend the monitoring for additional days.

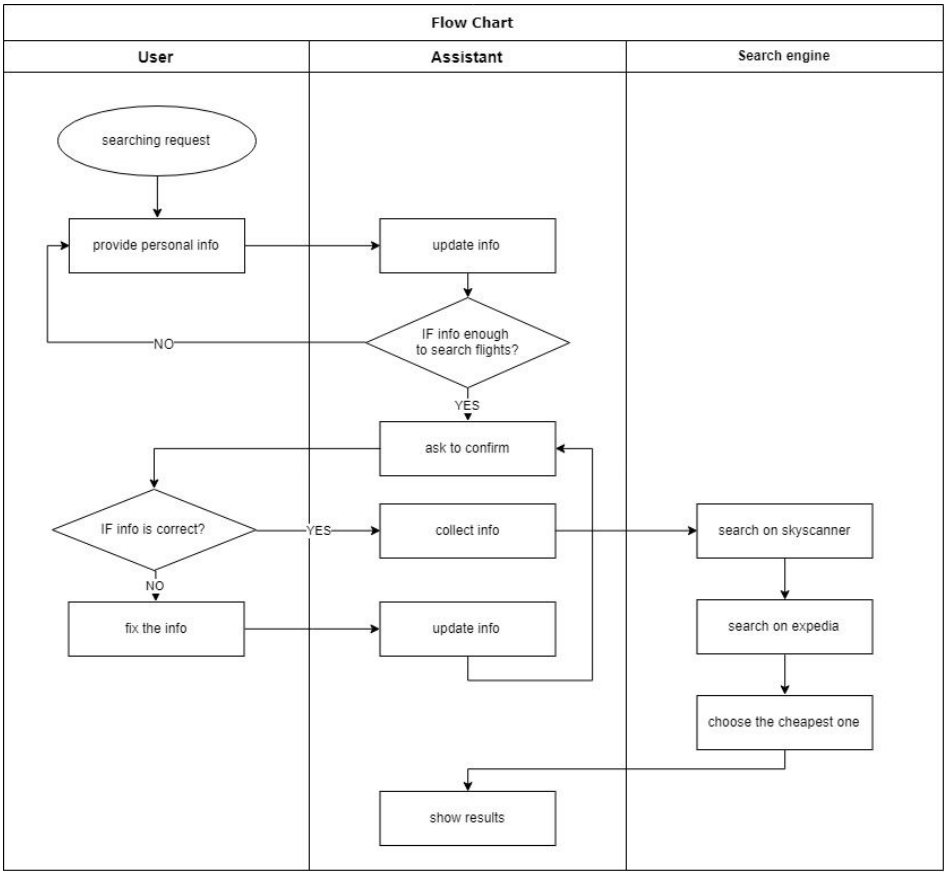
2. Solution

Skyscanner is the initial platform that our system is using to search for flight deals. For every search result, the system will extract the top 2 best deals at the top of the search (based on price).

Using these best deals, it will search for the same in Expedia to perform a price comparison. Depending on which platform offers the best price for the deal, the system will extract its price as well as hyperlinks to access this deal and save them into the database.

MongoDB was chosen to store flight request and search results, as the data structure will be different due to different legs for each flight deal.

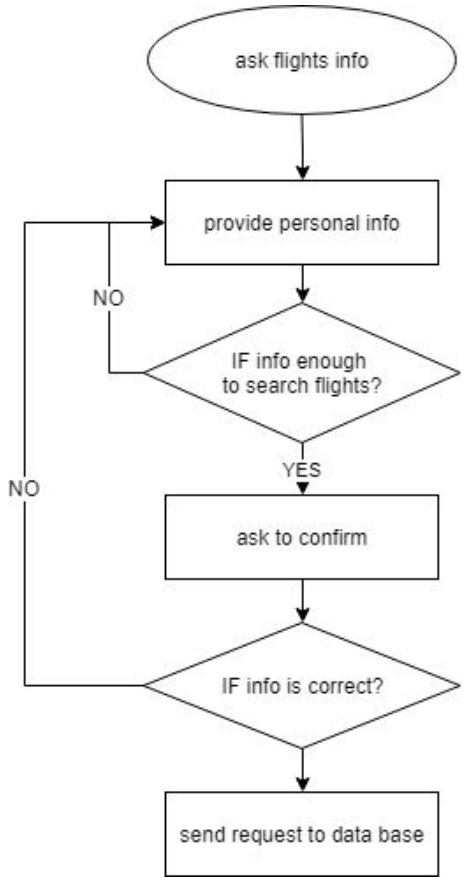




2.1. Chatbot

2.1.1. WeChat Session

Our WeChat chatbot's workflow is shown in the figure below. First of all, users need to provide the information that is required to search for flights. In addition, we also need to know how many days the user wants our system to monitor the flights as the ticket price may change day by day. When the mandatory information required to perform search is gathered, our chatbot will ask the user to verify and confirm all the information we gathered are accurate. If any information is incorrect, they can modify it in a fixed format. Finally, our chatbot will send an excel file containing our flight recommendation and its details at a scheduled time each day, till the end of monitoring period.



The table below shows the mandatory information needed for flight searching, for example departure city, destinations, departure dates, cabin class, number of adults, number of children and their age. When asking for the request, user needs to mention flights in their request text. Our system will try to recognize all the information we need.

Information	Question	Description	Example
City	Input Departure city in order	Departure cities	Singapore, beijing
			singapore
Date	Departure dates	Departure dates	July 1st, July 4th
			01/11/2019, 04/11/2019
Cabin class	cabin class	Type of cabin class	economy
Adult number	How many adults?	Number of adults	3
Children age	If you have children, input their ages	Children's age	2,3
			No
Monitor days	Monitor days?	Number of days the user wants to keep receiving the flight information	3

Here are some sample texts users can use for searching flights:

One way:
I would like to book a flight from singapore to beijing on November 1st for 2 adults.

Return:
I want to book a flight from Dalian to Shanghai on January 3rd and returning on January 5th for 1 adult and 2 children age 2 and 3

Multiple city:
1. Book a flight from Edinburgh to Dalian on December 10th, Dalian to Tokyo on December 15th, Tokyo to Osaka on December 20th
2. Book flights from Singapore to Edinburgh and then to Beijing, on November 1st, November 2nd and November 3rd

After all the information has been collected, our chatbot will ask the user to check the information is all correct or not. If there is something wrong, users can modify the information in fixed format. Below is the fixed format:
{ Type of information the user wants to change } : {new value} .

For example:
1. dates: 01/11/2019,04/11/2019
2. city: beijing,singapore.
3. adults number: 2
4. children age: 2,3

2.1.2. Speech-to-Text

In order to meet higher demands, we created speech-to-text recognition function. First of all, we used itchat to get the user's audio message and downloaded it into local folder. WeChat's audio messages are all in mp3 format and which have variable bitrate (VBR), thus the system need to pre-process the audio file before performing speech recognition.

To achieve that, the system will transfer the variable bitrate mp3 file into fixed bitrate (at 24kbps) mp3 file.
Subsequently, it uses 'ffmpeg' to convert the audio file format from mp3 to wav.

Finally, the system use the function in speech_recognition library, called 'recognize_google' to extract text from audio file.

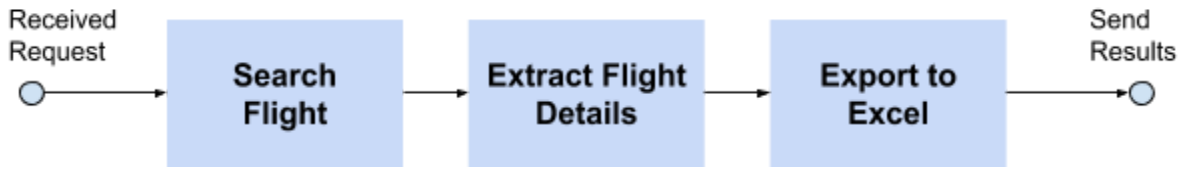
2.1.3. Text Recognition

Our system can retrieve the text from messages received by our chatbot, either in the form of a text message or an audio message. In order to recognize the information related to flights booking, we created a simple natural language processing function. We use spaCy to extract entities from the text, as all the cities have the label called 'GPE' and dates have the label called 'DATE'. Word 'Age' is classified as DATE entity by space, thus the function will exclude it in the entity detection. Instead, the function will use the words 'adult' and 'child' as anchors, to detect and capture the number of adults and the children's age if they have.

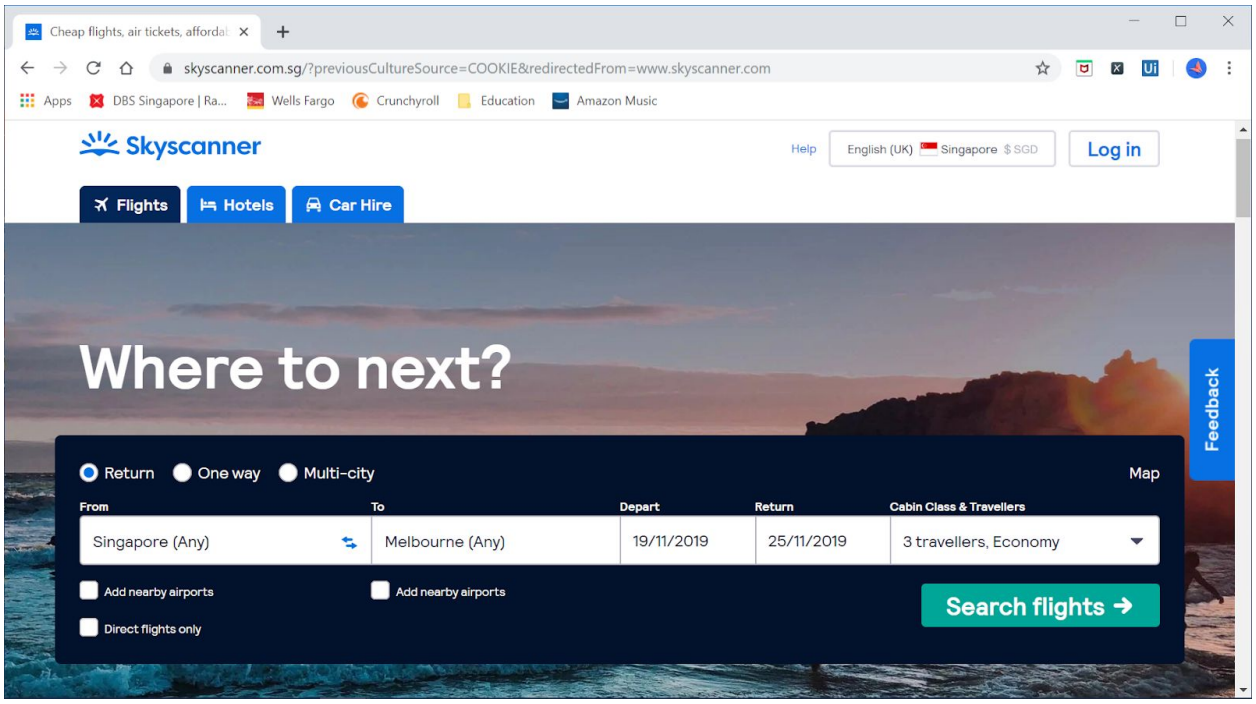
2.1.4. Send Flight Deals

After searching every day's flights requests, our system will export the data to Excel file and placed them in the folder corresponding today's date. Next, it will extract the user's nickname from the file name and send the file to the corresponding user.

2.2. RPA to Search for Flight Deals



2.2.1. Search Flight Deals

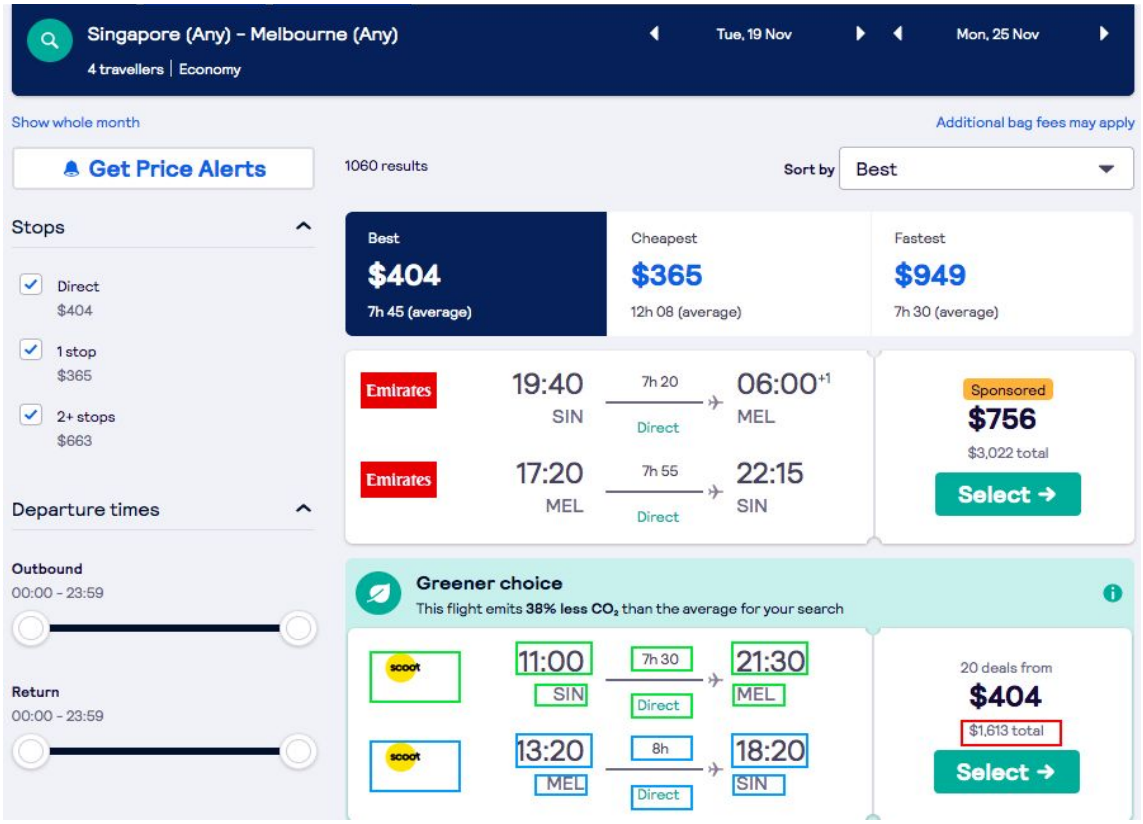


Based on the flight request (passed from Chatbot), the system will start searching for deals via Skyscanner platform. Depending on the type of flight, for example single trip, return trip or multi-city trip, the input search form will be different and the web element name and/or structure may also change too.

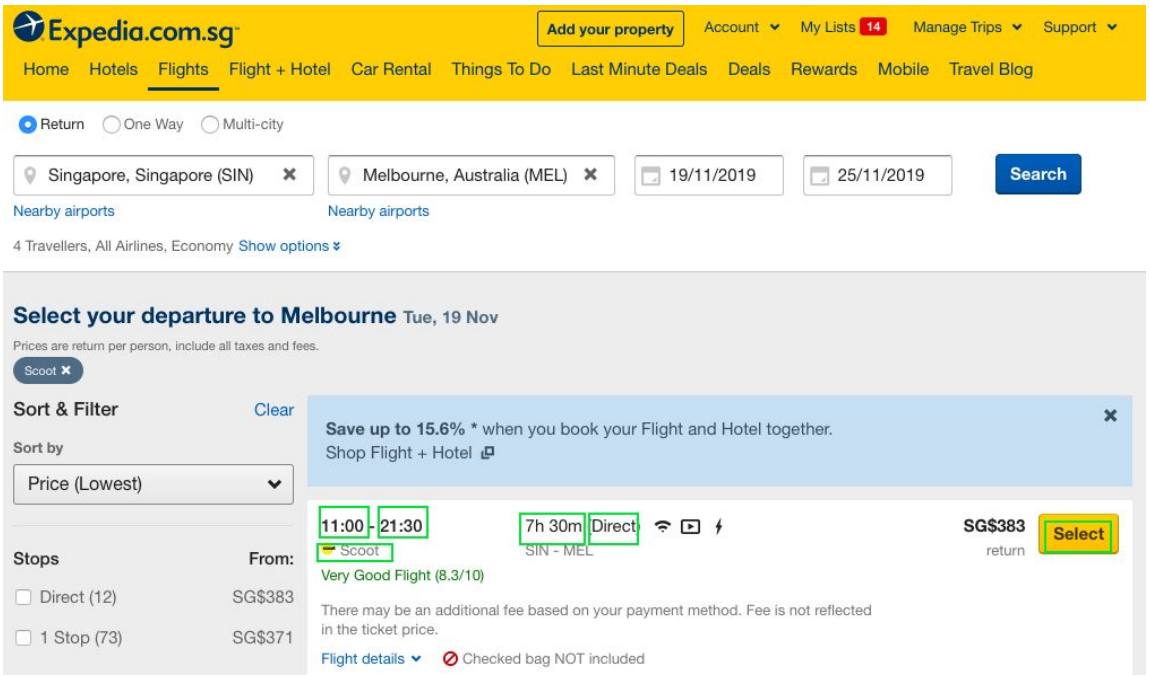
We also created a function to perform the same search using Expedia platform. This is to enable us to extract and compare flight deals and present users with the best deals for their request.

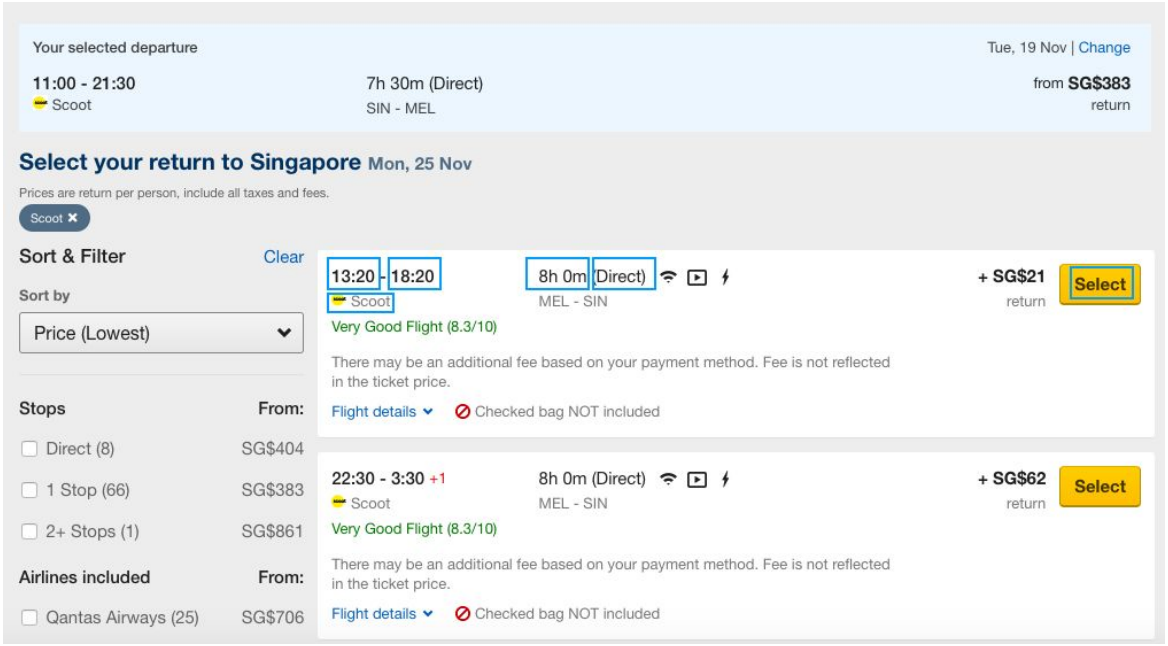
2.2.2. Extract Flight Deals

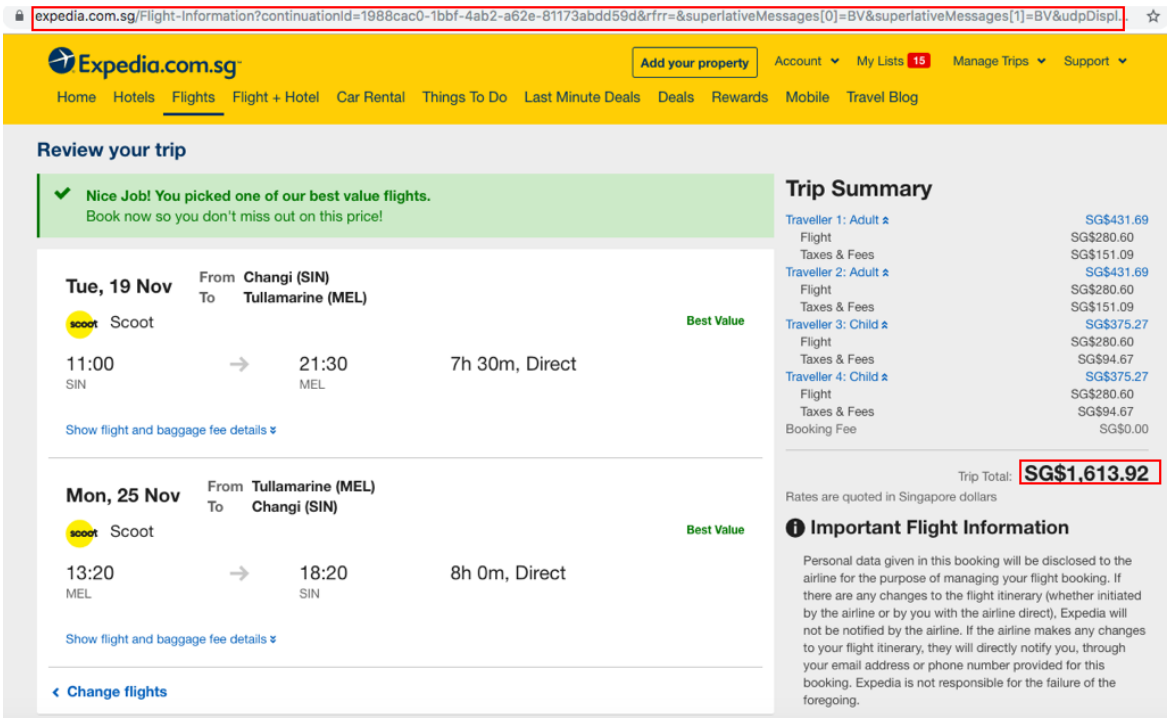
Firstly, We will choose the best two flights instead of the sponsored advertisement in Skyscanner and extract the necessary flight information including departure time, arrival time, flight duration, transfer type and transfer places, airline, and flight deal information including price and hyperlink to book. An example of return trip has been shown in the following capture:



After finding the best two flights, we will compare the prices of those chosen flights on Expedia. We will identify the flight with flight details including airline, departure time and flight duration. After comparing the price, we will give the cheapest price with the corresponding booking link of that website to the customers.







This function can cover three types of flights including single, return and multiple trip.

2.2.3. Export Results

After finding the best deals, the data are saved to the database and exported to Excel (XLS) and send to the user's WeChat app.

Following is a sample of the file content.

	A	B	C	D	E	F	G	H	I	J
1	**Deal**	**Price**	**Hyperlink**							
2	1	SG$1706.(	https://www.skyscanner.com.sg/transport/d/sins/2019-11-01/bjsa/bjsa/2019-11-05/tyoa?adults=2&children=2&adultsv2=2							
3		**Flight Leg**	**Bound**	**Departure Time**	**Arrival Time**	**Duration**	**Transfer**	**Transfer Place**	**Airline**	
4		1	SIN - PEK	01/11/2019 09:30 AM	01/11/2019 03:30 PM	6h	Direct		Air China	
5		2	PEK - HND	05/11/2019 05:20 PM	05/11/2019 09:30 PM	3h 10	Direct		Air China	
6	2	SG$1706.(	https://www.skyscanner.com.sg/transport/d/sins/2019-11-01/bjsa/bjsa/2019-11-05/tyoa?adults=2&children=2&adultsv2=2							
7		**Flight Leg**	**Bound**	**Departure Time**	**Arrival Time**	**Duration**	**Transfer**	**Transfer Place**	**Airline**	
8		1	SIN - PEK	01/11/2019 09:30 AM	01/11/2019 03:30 PM	6h	Direct		Air China	
9		2	PEK - HND	05/11/2019 03:35 PM	05/11/2019 07:45 PM	3h 10	Direct		Air China	
10										

3. Challenges

We have encountered a number of challenges when trying to fill up the search form as well as extracting data from the websites. In this section, we are going to cover the major challenges we encountered during the development of our RPA and how we overcome these challenges .

3.1. Audio message in WeChat has variable bitrate

In order to save the storage space, WeChat set their audio message in variable bitrate. However, when we transfer the format of WeChat audio message from mp3 to wav, it may has the warning said that because the transfer is based on bitrate, it may leading inaccurate and as a result, we can not recognize the text from the audio file.

To solve this problem, we set the bitrate to 24k before transfer the mp3 file into wav and can get a better text recognition result.

3.2. Chatbot and searching need to run at the same time

As our project aims to provide daily personalized flights information to each user, we need to do flights searching everyday at a particular time, which may cost a little time. However, some users may submit a new request while searching time.We need to make sure the two process can run at the same time.

As a result, we create two threads to run the two process. One is for WeChat chatbot to collect users' information and store it in the database. Another is for flights searching and sending excel files to users.

3.3. Users' texts are natural languages

It is hard and not user-friendly to ask the user to provide information in a fixed format, so we need to handle the different formats of the users' request. With the help of spaCy library, we can do some simple natural language processing, such as extract departure cities, destinations, departure dates, adult number and children's age.

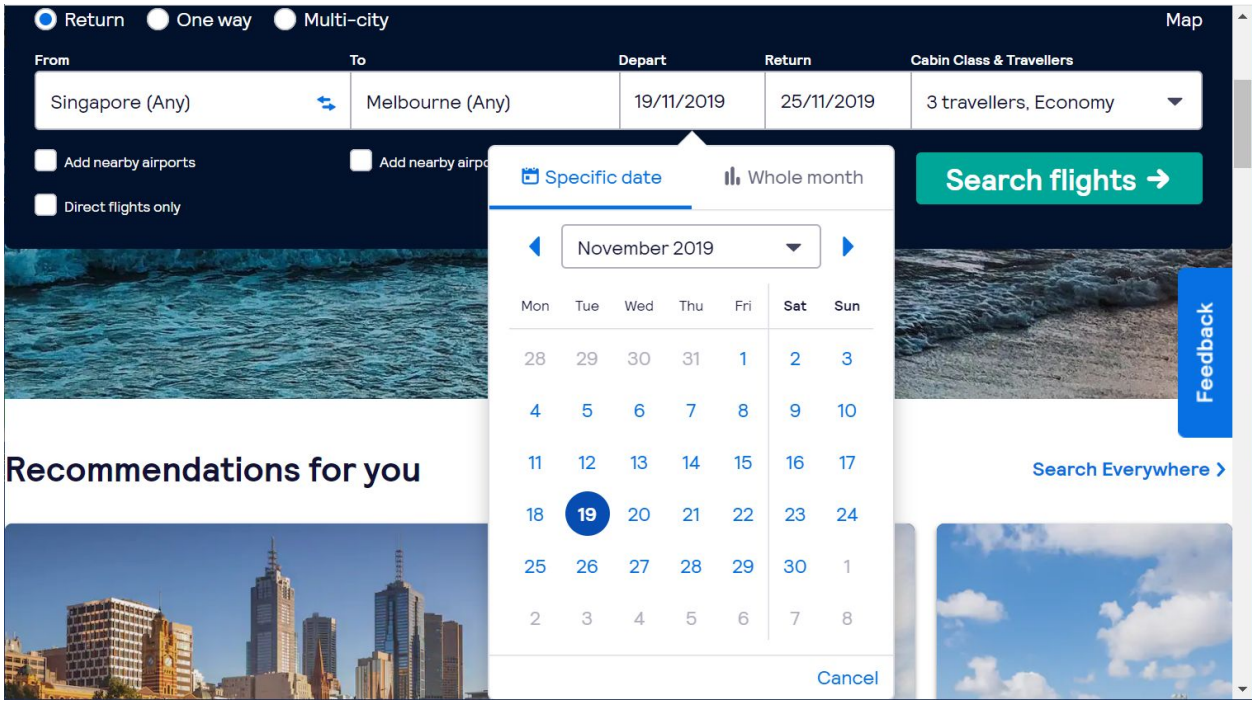
3.4. Several users may ask for a request at the same time

When different users ask for their requests at the same time, it may lead to overlapping or conflicts if their information can not be dealt with independently.

In our project, we create a global list to store different user's related information. All the operation(update information, send messages, etc.) for one user will use their nickname as the key to distinguish him. Once the request is finished and uploaded to the database, the user will be deleted from the list.

3.5. Date input is not editable

For Expedia, the date input fields are editable. Thus, we can simply use TagUI type() function to fill in the dates. However, for skyscanner, its date input fields are not editable. Thus, additional steps to look for the correct month and select the correct date are needed and that added additional time to the search process.



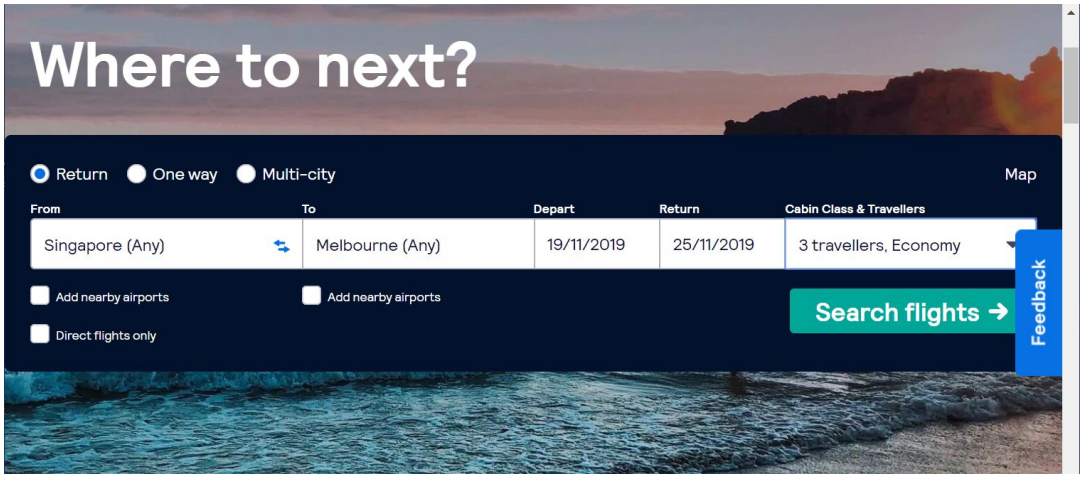
Even when the date input field is type-able, the system needs to first clear the value in the input field before it can start typing. However, clearing the value does not always work. For example in Expedia, we can clear the data in Depart date field but the same process does not work in Return date.

Thus, as a workaround, we set the return date in the search result page and search the deals again.

3.6. Content “overlayed” form input field, resulted in error

When the Chrome window is too small, we noticed “Feedback” overlay button is covering a corner of the “Search flight” button. This impacted the submission of the search. We are unable to determine if the issue is related to TagUI click() function of a submit button.

Thus, to ensure this does not happen, we set TagUI config file and set window size to a minimum of 1280 x 600.



3.7. Website form “remembers” last search criterias

For both Expedia and Skyscanner, once you did a successful search, it “remembers” the search criteria and populates the form. This is a problem especially for setting the number of travellers, where it is using a plus or minus button to increase and decrease the number of travellers.

As a result, if the RPA fill up the form without first checking the current value in the input field, it will not fill up the form correctly.

3.8. Handling sponsored advertisement on Skyscanner

Occasionally, there will be sponsored deals or advertisement at the top of the search results in Skyscanner. In order to return the best deals, our system will skip these sponsored deals and return deals purely based on its price and flight duration.

3.9. Different flight types, different structures

For Expedia and Skyscanner, the time and duration formats are different. So when using the result from Skyscanner to search the same flight on Expedia, data formatting is must needed.

3.10. The information on two websites had different format

In Expedia, the return type has a different structure with single and multiple trips. For the return trip, outbound and inbound trip has to be chosen separately. But for single and multiple trips, combined deals will be generated automatically. So two types of structures should be considered.

3.11. Search result opened in a new window tab

In Expedia, after selecting the specific flight leg, the final result containing the final price of the deal is open in a new window tab. This causes an issue with TagUI as the session is still tied to the original window tab.

To resolve, we used TagUI Utility function - popup() to find the tab where URL containing the specific keywords “Flight-Information?” to look.

4. Conclusion

Our RPA heavily depends on the design of the website we used to automate the search. If there are any changes to the page layout or element name, the RPA will failed. We can build periodical test of our RPA and alert when error encountered. That is the reason we selected more established platform like Skyscanner and Expedia, where changes to web page design does not change often and drastically. In addition, they are also the top search platform for flight deals.

In addition to website layout, for some of the element, we are using TagUI Utility functions to identify their presence in the page. As a result, we need to ensure no other window is on top of the web browser when running the service.

By default, our system is using the price as the criteria to determine which is the best deals. However, different users will have different criteria on what is deemed the best deals. With enough data collected from the different requests we received and deals that users selected, we can use it to create a model to profile the users' preference and that will be factored in when the system search for deals. This can potentially improve the search results that our system generated.