EASYREQ

INTELLIGENT SOFTWARE AGENTS PROJECT

GROUP MEMBERS	*STUDENT ID*
Aniket Mohan Arasanipalai	A0198481N
Siddhant Naveria	A0198485H
Vishaag Suriyanarayanan	A0198490N
Akshay Sachdeva	A0198532W

Contents

EXECUTIVE SUMMARY

With many students and working employees coming in to Singapore, either for studies or work related environments, housing has become the first priority for all of them. While either moving in to a house for the first time, or moving houses after a period of time, there exists a lot of forms that need to be manually filled which may become exhausting for the customer.

Based on robotic process and intelligent process automation concepts, our group of four members, brainstormed and decided to build a system that can easily automate this process of form filling when searching/relocation of houses, based on the customer details. The entire system would be a chatbot based system where a customer can interact with an intelligent chatbot, which would assist him/her to fill up various forms that are required, and moreover, organize key appointments that maybe required in certain cases.

We began by getting a good understanding of the overall key processes of the proposed solution. Next, we determined the areas which needed targeting that we agreed should be implemented into business processes. We also made sure we analysed the processes with our improvements and see how these improvements were handled with test cases to validate our assumptions.

The outcome of the system, is that the system would share screenshots of the filled forms to the user for his consent, post which, based on the approval the respective forms can be ‘submitted’.

1. BUSINESS PROBLEM BACKGROUND

One of the more interesting areas over recent time has been in the area of robotic process automation. With the advent of more intelligent systems, the process of automation is becoming highly demanded.

Add to this, the growing rise in people looking to relocate houses on a daily basis is huge. The time consumed on the aspects of filling forms might become tedious. Therefore, it becomes necessary to have a system which can help in filling out these necessary forms based on the user details, so that the user can focus on the other aspects while moving to a new house.

This system can help those students who need to change houses frequently as well as the people who would generally relocate after a certain amount of time.

2. OBJECTIVES AND SUCCESS MEASUREMENTS

2.1 OBJECTIVES

The objective of this project is to create an intelligent framework to assist a user in completion of the various online forms that are required as part of relocation or moving to a new house.

With the vast majority of population, looking to move houses on a daily basis, and with so many factors and categories to keep track of, it becomes easy if they are able to interact with a system which takes in user details and automates this process, in order to minimize the time spent on the same.

The systems aims to provide a solution to two challenges currently faced:

1. **Scalability** - where personalized interactions are challenging to support at scale due to the nuances of human patterns in communication which varies depending on individual and cultural differences
2. **Speed** - where in this current world, users expect instantaneous responses and services to their queries. Such solutions will free up expensive resources for more complex and high value-adding tasks

2.2 SUCCESS MEASUREMENTS

It is vital to understand the success of the system based on a few measures.

There are a few key measures of our system:

1. Whether the system was able to understand the user
2. Whether the system was able to respond to the specific query being asked
3. Whether the system was able to present the related information
4. Whether the system was able to provide alternatives in case information related to user's query was not present.

3. SOLUTION

3.1 TARGET AUDIENCE

The major target audience would be the students/people living on a rental basis who move houses on a regular basis and want to get the process finished in a timely manner.

3.2 DATA QUALITY AND KNOWLEDGE BASE

All information used to build the 'EasyReq' system is obtained from the user itself.

3.3 PROJECT SCOPE

In this project, an intelligent form filling assistant is created, using the front end – back end framework that will assist the user in automating the process of application filling based on house relocation, required by various websites.

The system will respond to user preferences and provide the answer on a chatbot.

4. IMPLEMENTATION

4.1 SYSTEM DEVELOPMENT

System Development is the process of defining, designing, testing and implementing a new software application or a program. It can include the internal development of customized systems, the creation of database systems or the acquisition of third party developed software.

4.2 SYSTEM ARCHITECTURE

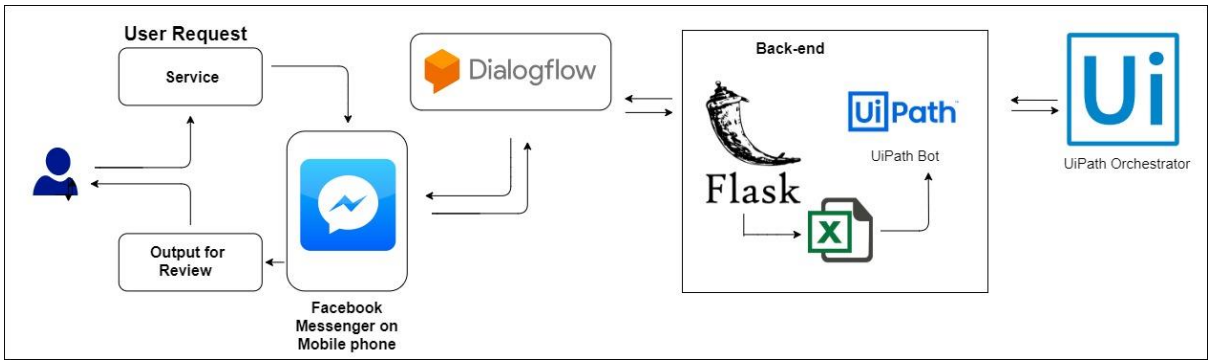


Fig 4.2 System Architecture of EasyReq

Figure 4.2 shows the system architecture of 'EasyReq'. It illustrates how the different components interact with each other. After the user keys in input requesting for the service(Broadband/Utilities application process), he wants to opt for as well as providing his personal details, using Facebook Messenger, Dialogflow will receive the input. Dialogflow will extract the intents and parameters from the sentence and feed into the flask server.

The flask server then communicates with the UiPath Bot which automates the required service, the user has opted for. The orchestrator is used for controlling the UiPath Bot. Post this, the results are returned via the same route, and the user would get the response for his request on the chat window as well as obtain the screenshots on this email.

4.2.1 DIALOGFLOW SETUP

As Dialogflow provides the understanding of natural language it makes it easy to design and integrate the conversational user interfaces. ‘EasyReq’ has been built using the Dialogflow. The training of the chatbot has been completed using the intents which categorizes the end-user’s intention for end user’s intention. For the project, multiple intents have been created to well recognize user’s questions.

Dialogflow provides accessibility to features such as Intents, Entities, Fulfilment and Integrations which have been used in the project.

To provide the accessibility of a backend server, the URL has been provided in the URL section of Webhook in Fulfilment. As each intent parameter has the type called entity which dictates how exactly end user expressions would be extracted. As for each of the question asked to the chatbot, the Chatbot will match the entity of the intents based on the training in the Dialogflow, then the parameters will be passed through the webhook call to the backend of the project.

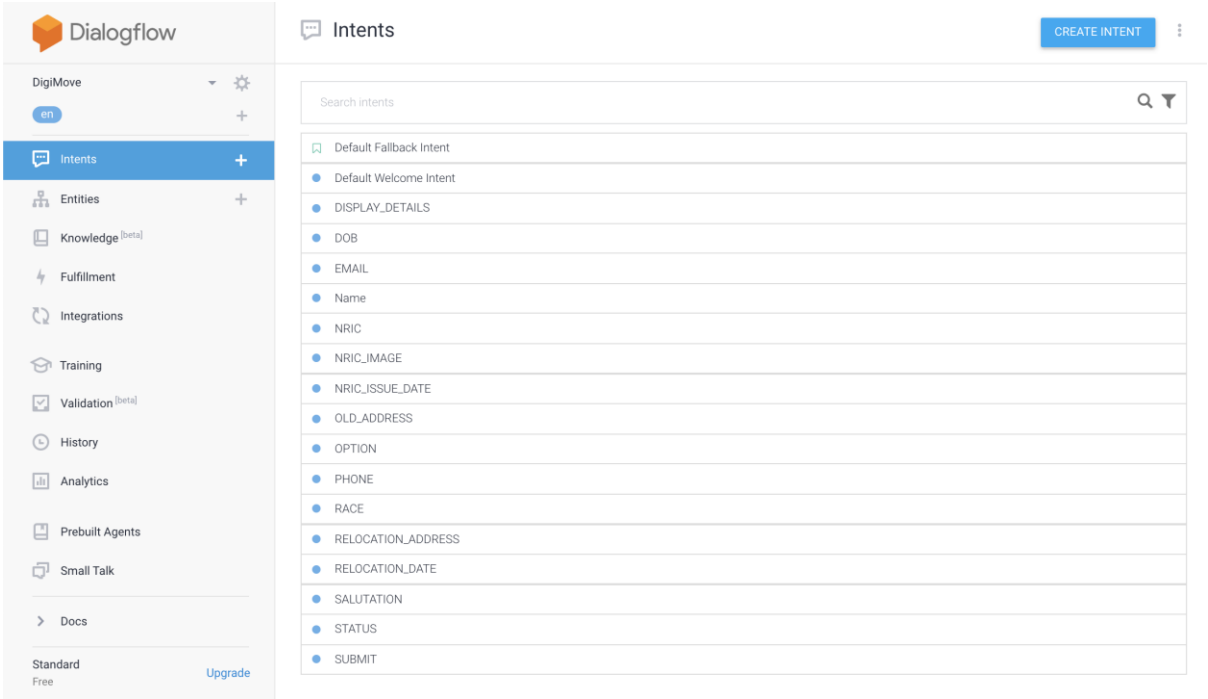


Fig 4.2.1 Dialogflow Intent Page

4.2.2 UI TEST

To get a good user experience, we have used the Facebook Messenger platform provided in Dialogflow.

4.2.3 FACEBOOK MESSENGER

Facebook Messenger is the virtual assistant that we have used, which provides the flexibility and ease to be integrated with Dialogflow. It provides the interface to use the Chatbot which can be deployed in very few steps. Facebook Messenger is already accessible on most of the Android smartphones and can be downloaded for iOS too.

It provides speech as well as text response along with Basic Cards, Chips, Button, Carousel and many other UI features.

The ease of use, flexibility, scaling are some of the features provided by Facebook Messenger after the application is deployed.

4.2.4 FLASK SERVER

Flask is a micro web framework written in Python. It is classified as a micro framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions.

However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

4.2.5 UiPATH

The UiPath platform is automation software that is powerful, fast and flexible. It consists of three basic components:

UiPath Studio – a tool that enables you to design automation processes in a visual manner, through diagrams.

UiPath Robot - executes the processes built in Studio, as a human would.

UiPath Orchestrator - a web application that enables you to deploy, schedule, monitor and manage Robots and processes.

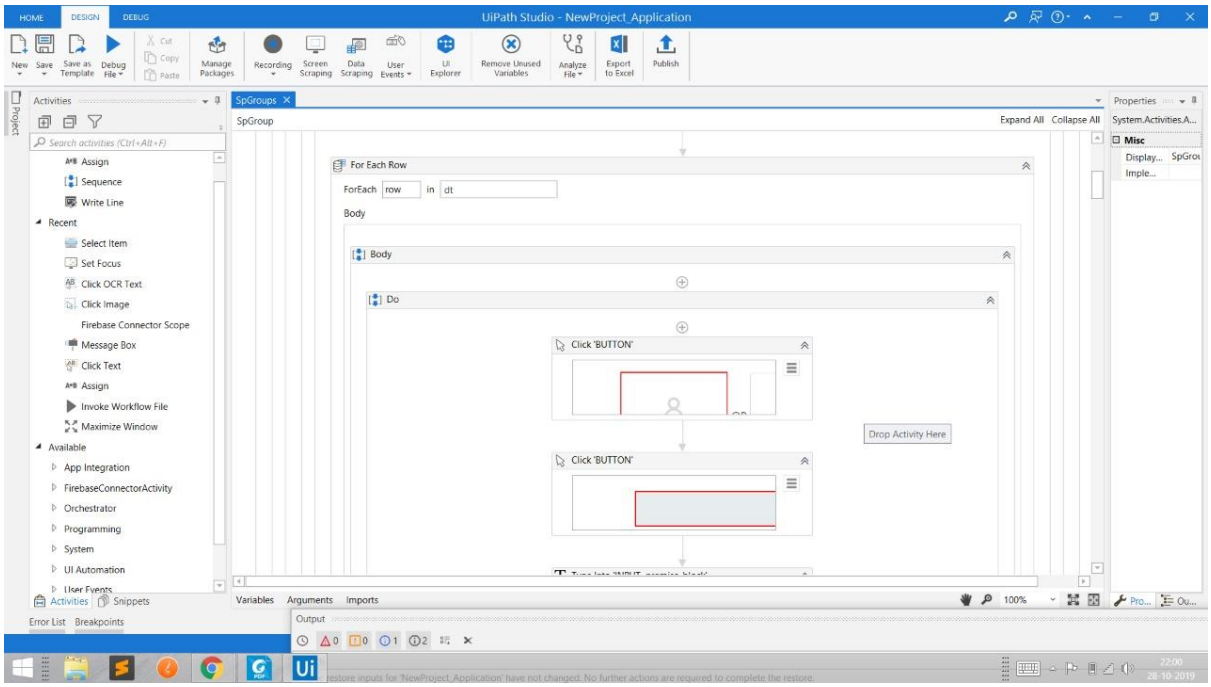


Fig 4.2.5 UiPath Process Flow Environment

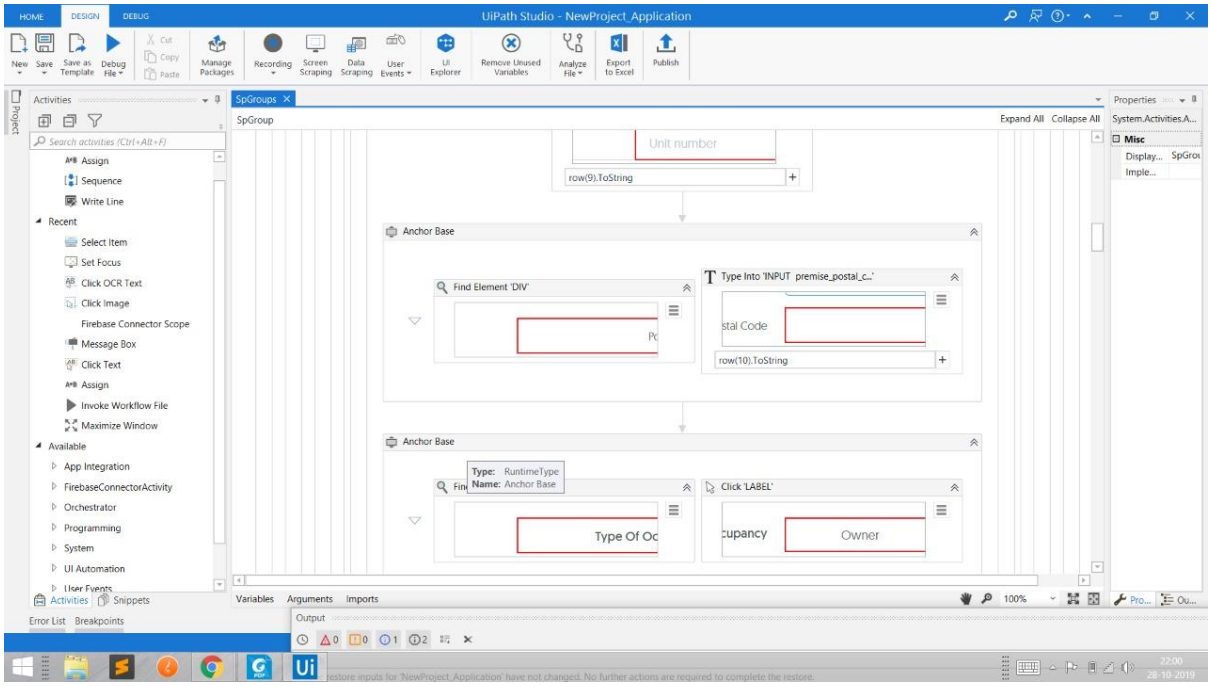


Fig 4.2.5 UiPath Process Flow Environment

5. PERFORMANCE AND VALIDATION

We perform validation on a scenario to ensure ‘EasyReq’ provides the correct expected output.

The Scenario is a person who wants to use ‘EasyReq’ to fill the Broadband and Utilities application process, that would run one after the other. Below images, will show how the user could interact with ‘EasyReq’.

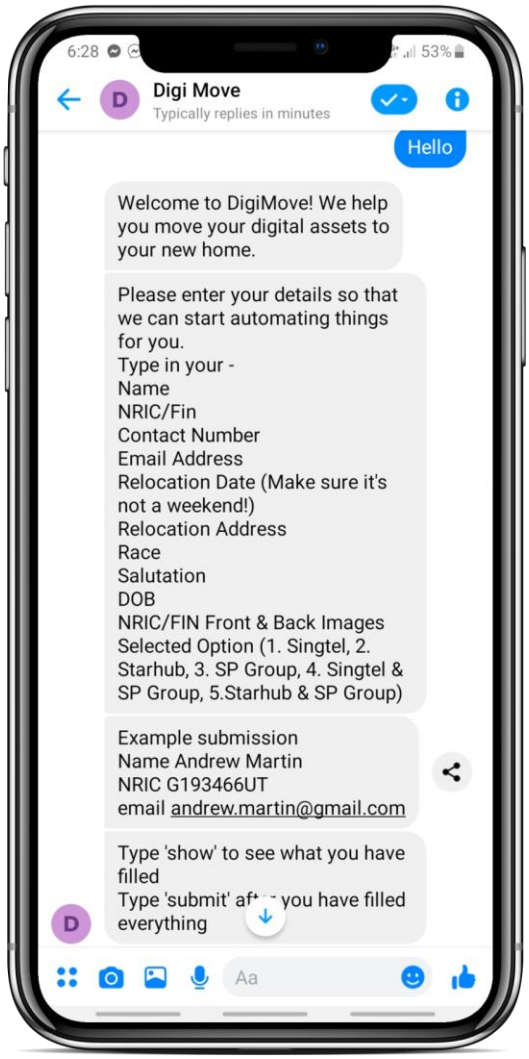


Fig 5 User Interacts with Chatbot and provides details

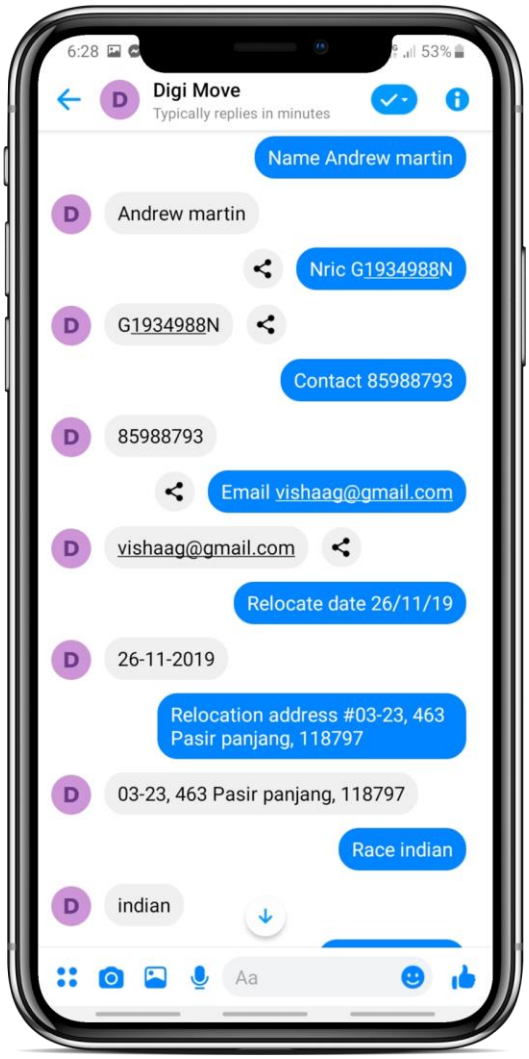
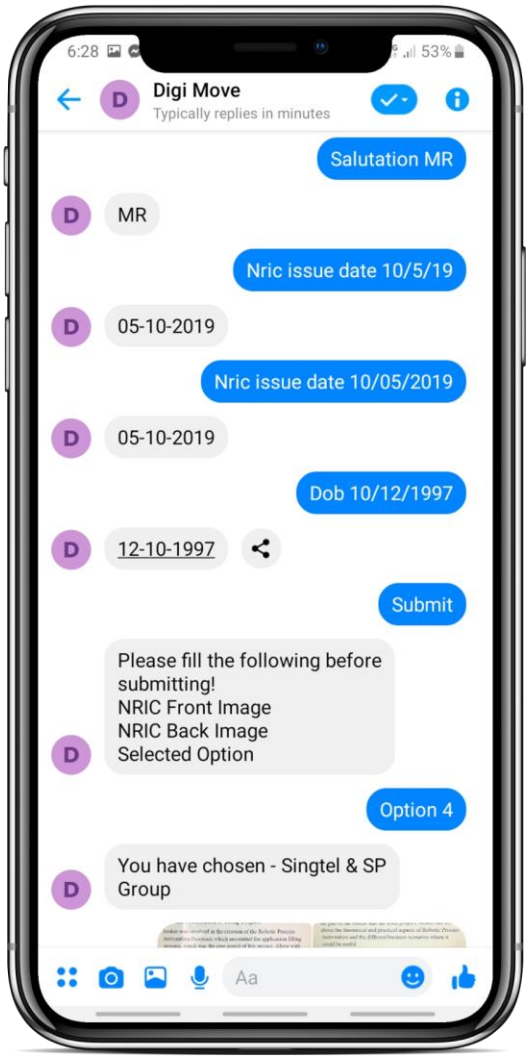
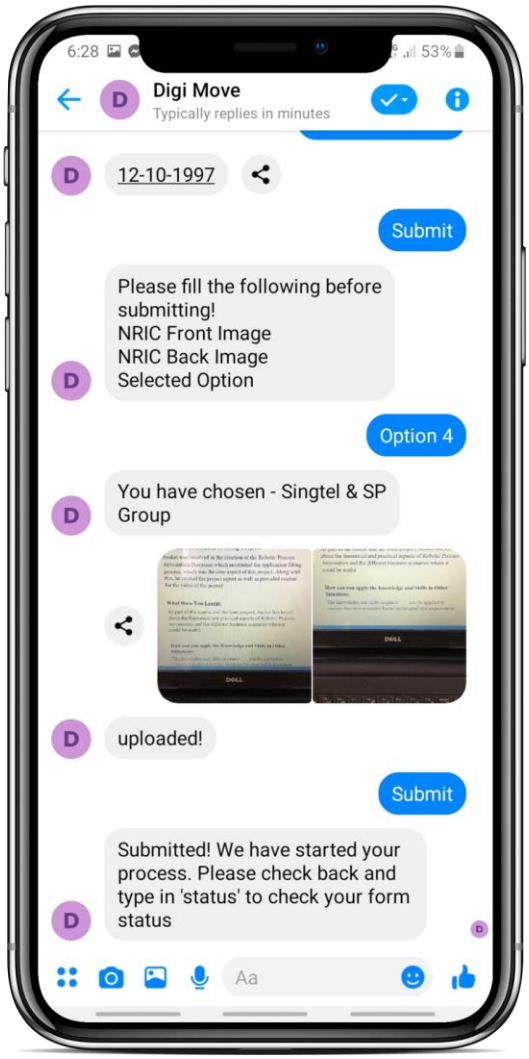


Fig 5 User provides details


6:28
53%
Digi Move
Typically replies in minutes
Salutation MR
MR
Nric issue date 10/5/19
05-10-2019
Nric issue date 10/05/2019
05-10-2019
Dob 10/12/1997
12-10-1997
Submit
Please fill the following before submitting!
NRIC Front Image
NRIC Back Image
Selected Option
Option 4
You have chosen - Singtel & SP Group
Aa


6:28
53%
Digi Move
Typically replies in minutes
12-10-1997
Submit
Please fill the following before submitting!
NRIC Front Image
NRIC Back Image
Selected Option
Option 4
You have chosen - Singtel & SP Group
uploaded!
Submit
Submitted! We have started your process. Please check back and type in 'status' to check your form status
Aa

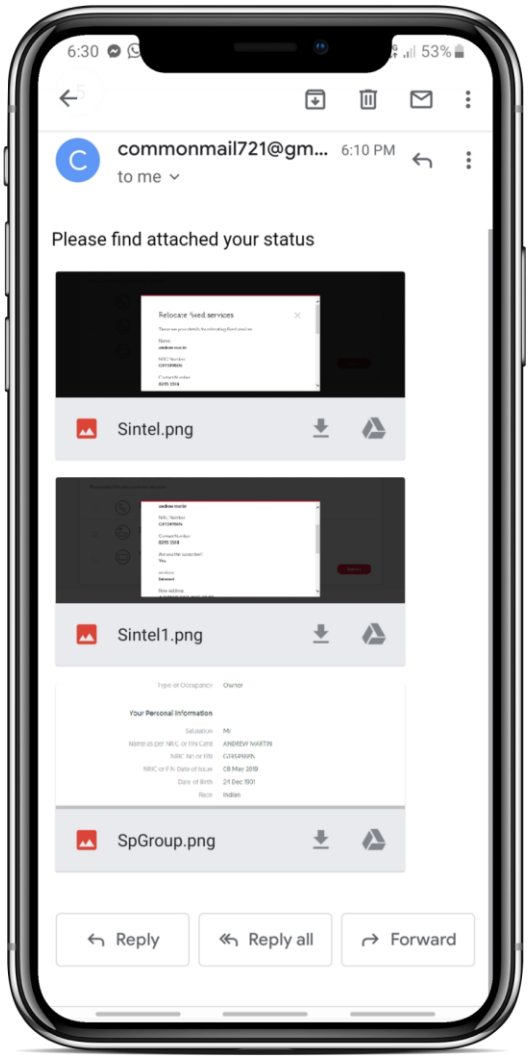


Fig 5 User obtains screenshots of filled application on his mail

6. CONCLUSION AND FUTURE STEPS

In this project, an Intelligent Process Automation system, 'EasyReq', is developed using UiPath and Google DialogFlow, in order to add customers in different application processes while relocating or moving to new locations.

Firstly, the chatbot is trained with more than 100 possible utterances that are annotated with intents and tagged with entities. Then, when a text input is received, DialogFlow will detect its respective intent and entity. The request is

passed through a flask server and the automation phase of the process is done in UiPath.

With such a system, the responses are standardised, less erroneous and are capable of performing with high accuracy. It works very well for such specific closed domain, and thus customer satisfaction and attention can be improved.

Therefore, this system can serve as an effective customer service tool and the valuable
human resources can be reallocated to handle more complex tasks.

Currently, 'EasyReq' operates based on 3 websites who's applications are generally required while relocation. Future enhancement of the project could involve, including link to other websites, scaling the system to include other domains such as Opening of bank account, Income Tax Filing, as well as syncing the application with user's calendar in order to obtain his schedule beforehand and providing suggestions.

7. REFERENCES

"Knowledge Connectors | Dialogflow Documentation | Google Cloud"
https://cloud.google.com/dialogflow-enterprise/docs/knowledge-connectors

"UiPath"

https://docs.uipath.com/

"Singtel Relocation Form"

https://www.singtel.com/personal/my-account/manage-accounts/relocate-services-web

"Starhub Relocation Form"

https://www.starhub.com/personal/forms/relocation.html

"SP Group Utilities"

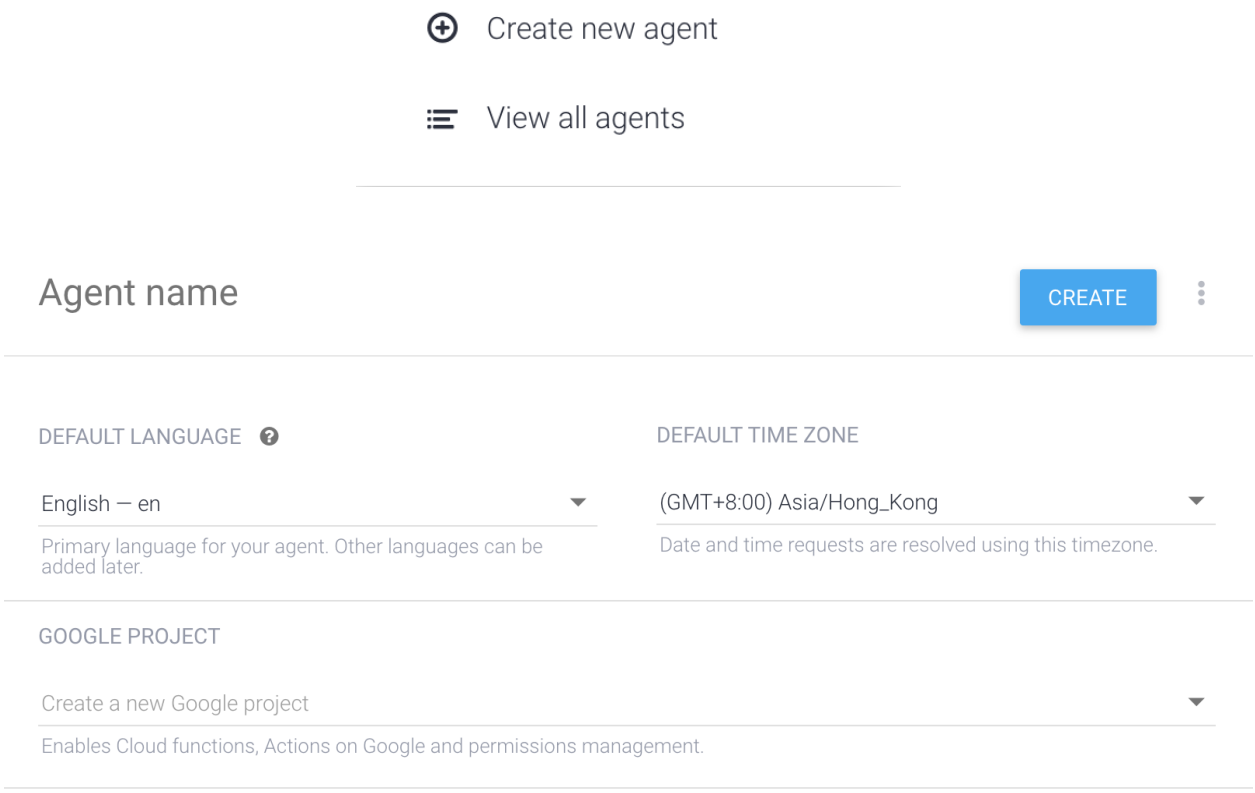
https://services.spservices.sg/#/terminateAccount

APPENDICES

APPENDIX A: USER GUIDE

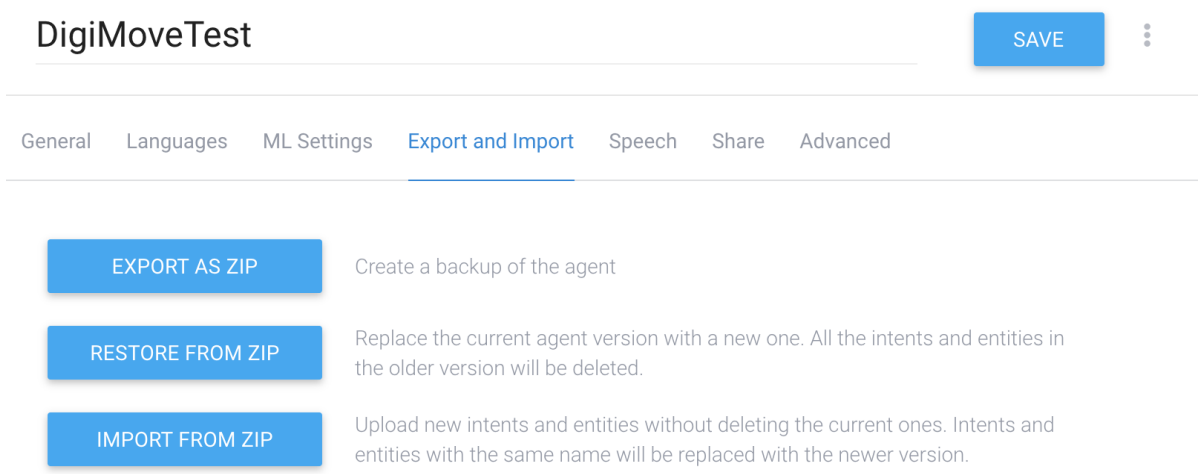
Part 1 - Dialogflow

1. Create an account in Google Dialogflow
2. Create a new agent from the Dialogflow dashboard.

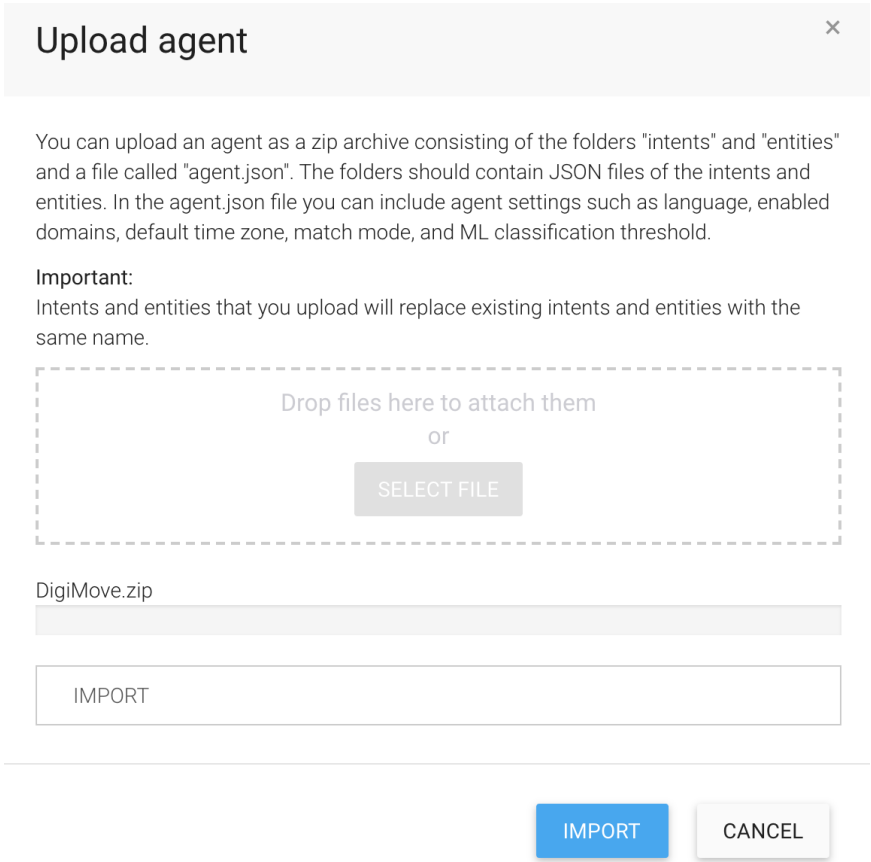


3. After creating a new agent, go to the agent **settings** -> **Export and Import Tab**

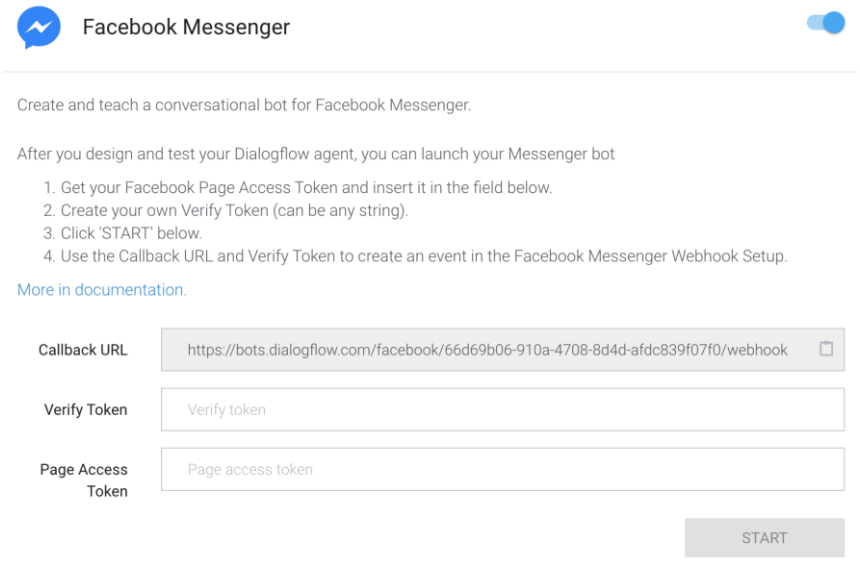
And click on Import From Zip.



4. Upload the .zip file inside the Dialogflow folder in the project repository.



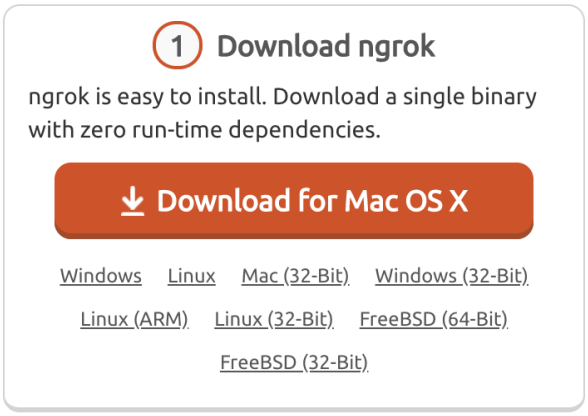
5. From the Dialogflow dashboard, click on Integrations from the left panel and enable Facebook Messenger Integration.
6. Now, you will see a popup asking you to fill in the Verify Token and the Page access Token.



7. Click on **More in documentation** follow the guide to create a Facebook App from the developers console and a corresponding facebook page for it. Then, paste in the Verify Token and the Page access token in Dialogflow facebook integration.
8. Now, we need to set up the backend for the system. For this, go the the folder named Backend in the project repository and open the command prompt there.
9. Run python app.py in the directory. If you get the following text, your backend is running perfectly.

```
(base) ➜  digimove_backend python app.py
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 248-267-968
```

10. Now that the server is running locally, we need to give access to Dialogflow to send requests to this local server. In order to do that, we would need a tool called ngrok.
11. Go to ngrok.com and make an account.

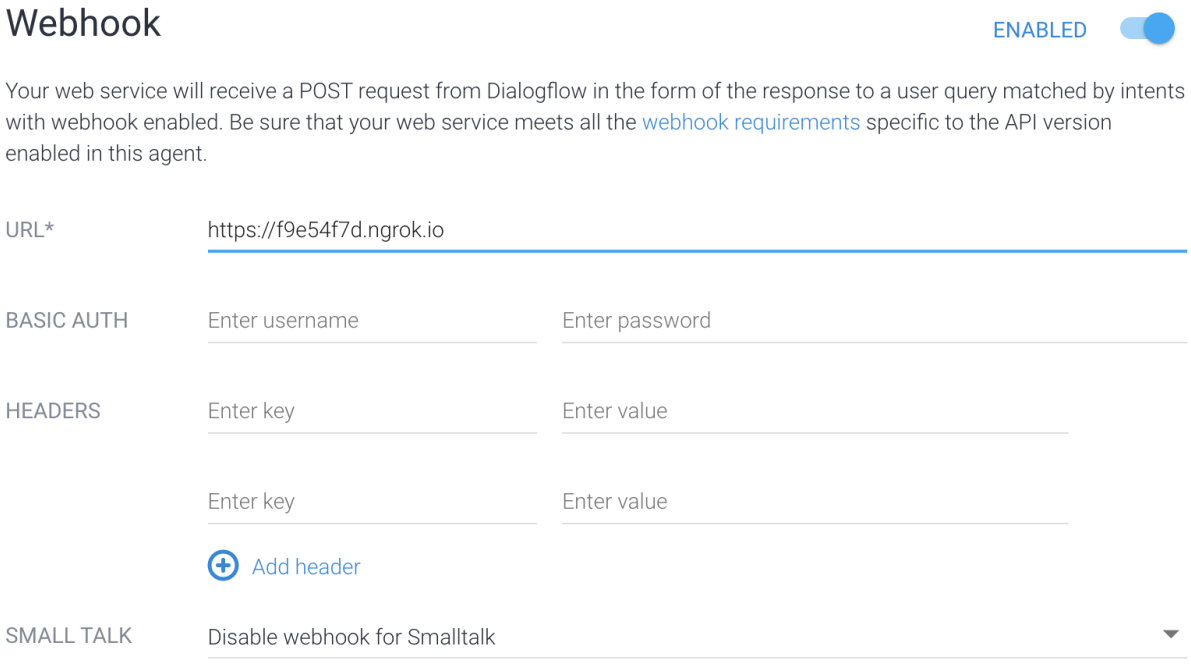


12. Download ngrok after logging in.
13. Go the the directory where ngrok has been downloaded and open the terminal in the directory.
14. In the terminal, type in ./ngrok http 5000

```
(base) ➜ projects ./ngrok http 5000
```

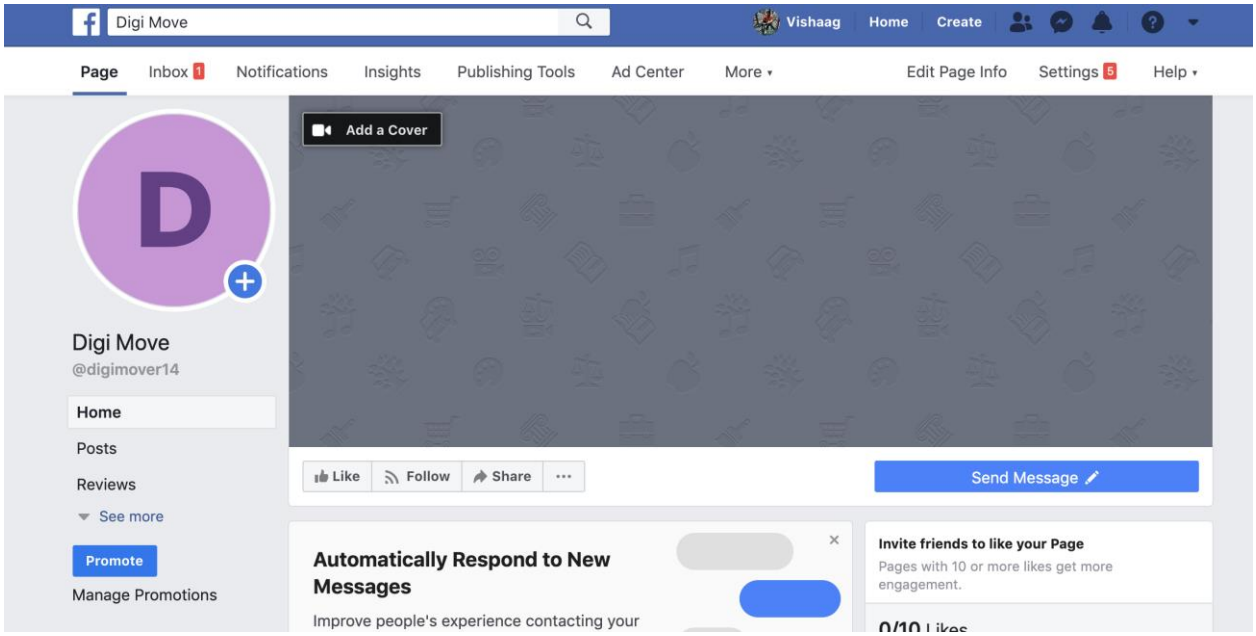
```
ngrok by @inconshreveable

Session Status                online
Account                       Vishaag Suriyanarayanan (Plan: Free)
Version                       2.3.35
Region                        United States (us)
Web Interface                 http://127.0.0.1:4040
Forwarding                    http://ebc0c352.ngrok.io -> http://localhost:5000
Forwarding                    https://ebc0c352.ngrok.io -> http://localhost:5000

Connections                   ttl     opn     rt1     rt5     p50     p90
                              147     0       0.00    0.00    0.02    0.23
```

15. You will see the output as shown above. Copy the text next to **Forwarding** as highlighted in the above image.
16. Now, go to Dialogflow again and click on **Fulfilment** from the left menu.
17. Enable the webhook and paste the copied link in the URL area.

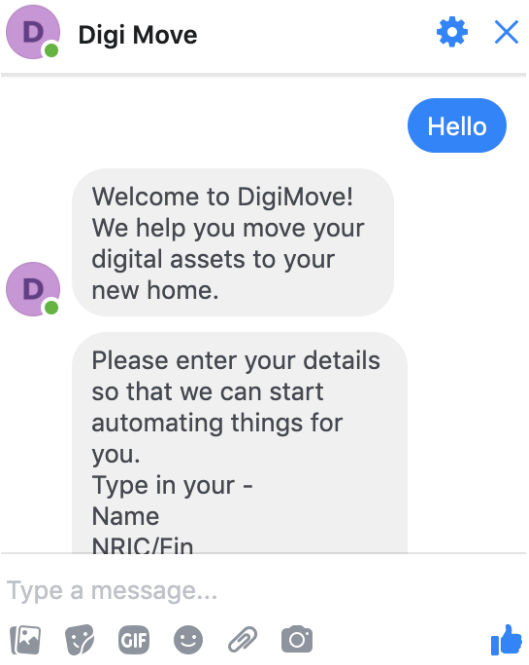


18. Now DIalogflow will be able to communicate with the local backend server through the link given in the fulfilment.

19. Now, to test, go to facebook and open the page you had created earlier.



20. Click on the Send Message and click on Test Button and you will open the chat window. Type in 'Hi' or 'Hello' and you should see the chatbot responding.



Prerequisite for RPA backend

Uipath should be installed in the system to run the RPA backend. For installation, kindly go the below and download community edition **UiPathStudioSetup.exe** file.

https://www.uipath.com/developers/community-edition

Kindly refer the below link for guidance in installation process.

https://docs.uipath.com/studio/docs/about-installation#section--exe-installer-user-mode-installation

1. After the installation is complete, chrome extension needs to be added in the uipath studio and enable the extension on chrome as well. This step is important as bot is configured to run on chrome and internet explorer browsers. Please refer to the below link for guidance.

   https://docs.uipath.com/studio/docs/installing-the-chrome-extension

2. Next step in the prerequisite is to create a robot in the orchestrator and configuring the bot created in orchestrator to the bot installed in the system.
   Link to orchestrator- https://platform.uipath.com/portal_/cloudrpa

   For standard robot creation in orchestrator refer to the below link.

   https://docs.uipath.com/orchestrator/docs/managing-robots#section-creating-a-standard-robot

   For creating environment, refer to the below link.

   https://docs.uipath.com/orchestrator/docs/about-environments

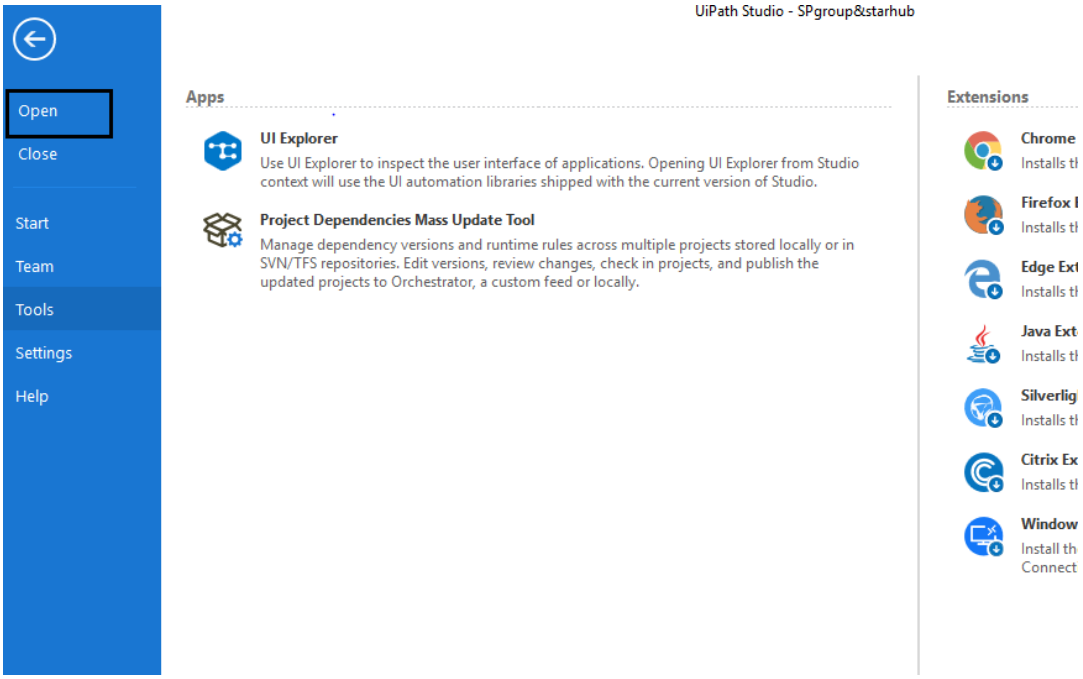
   For configuring orchestrator with the system, refer to below link.

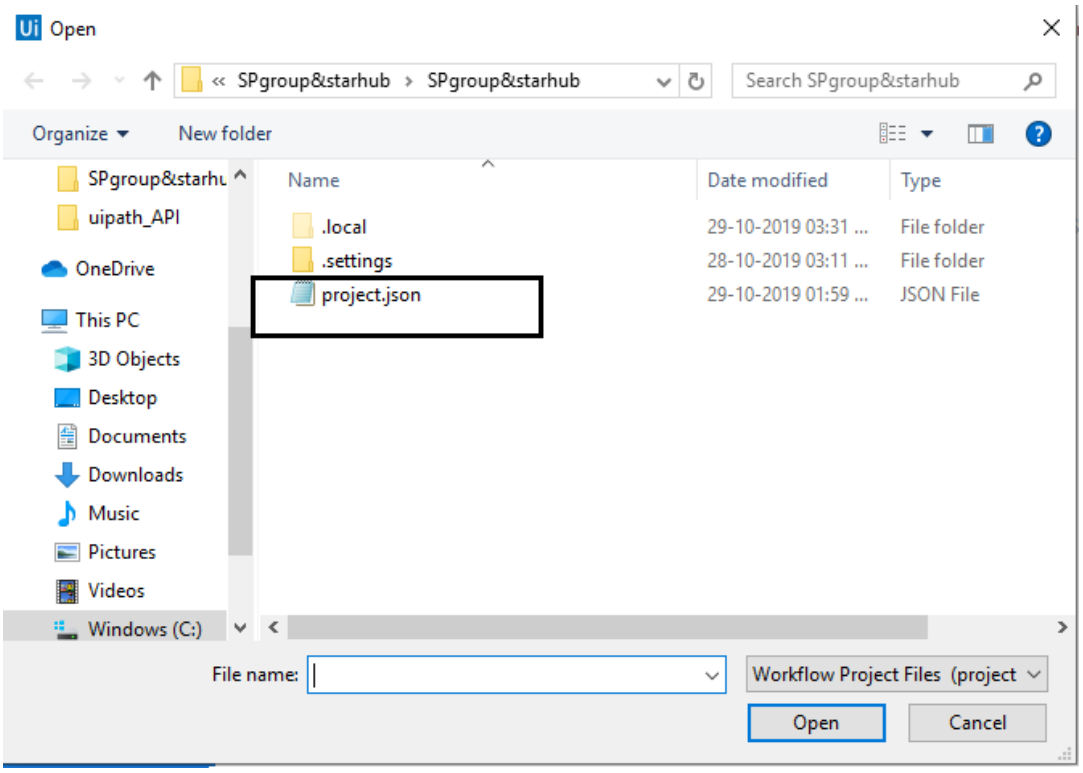
   https://docs.uipath.com/orchestrator/docs/connecting-robots-to-orchestrator

Now the system is configured to handle the RPA processes. Next step for creating the RPA backend is to push the code and configure the API_bot.py file according to the system.

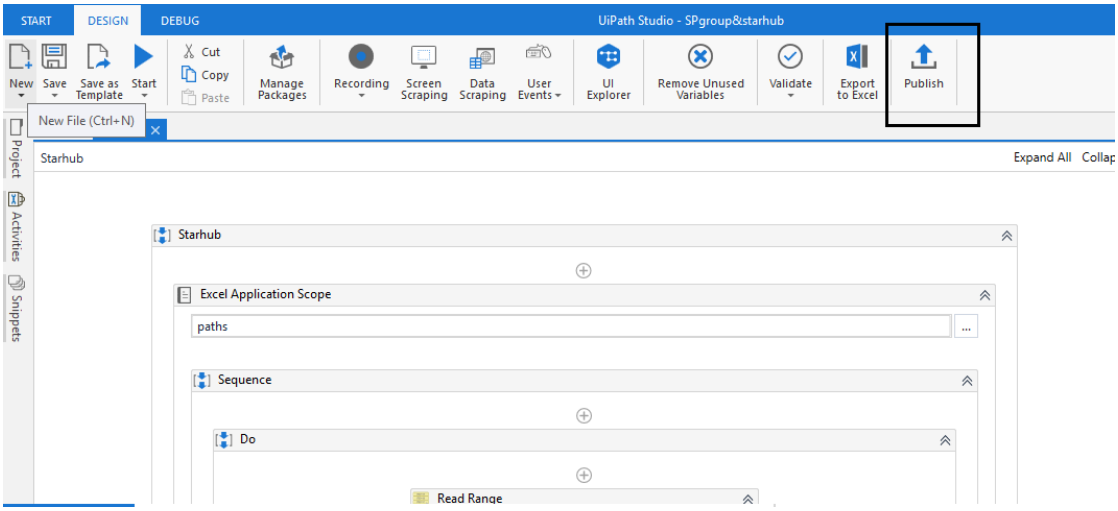
Adding the code to the orchestrator

1. Download the code from the github repository, using below link "github link"
2. Open the Uipath studios and open the project. Refer to below screenshots.





3. After loading the code to the studios, click on publish button to push the code to the orchestrator.



4. Perform the same steps to push both the projects to the orchestrator.
5. Now next step is to login to the orchestrator and create process for both the projects. Kindly refer the below link for guidance. https://docs.uipath.com/orchestrator/docs/managing-processes

Configuring API_bot.py file

This file is used for making API calls to the orchestrator and for making those calls access token is needed. Below is the code snippet, these constants need to be configured according to the orchestrator.

```
refresh_token="[redacted]
account_logical_name="sidhaiorqunx"
service_instance_logical_name="sidhantDefa1a3t171503"
process_name_sintel="'newtest_testEnv'"
process_name_spgroup_starhub="'SPgroup_starhub_testEnv'"
OrganizationUnitId="1327"
API_jobdetails_spg_starhub="https://platform.uipath.com/"+account_logical_name+"/"+service_instance_logical_name+"/odata/Releases?$filter=%20Name%20eq%2
API_jobdetails_sintel="https://platform.uipath.com/"+account_logical_name+"/"+service_instance_logical_name+"/odata/Releases?$filter=%20Name%20eq%20"+ p
API_jobrun="https://platform.uipath.com/"+account_logical_name+"/"+service_instance_logical_name+"/odata/Jobs/UiPath.Server.Configuration.OData.StartJob
```

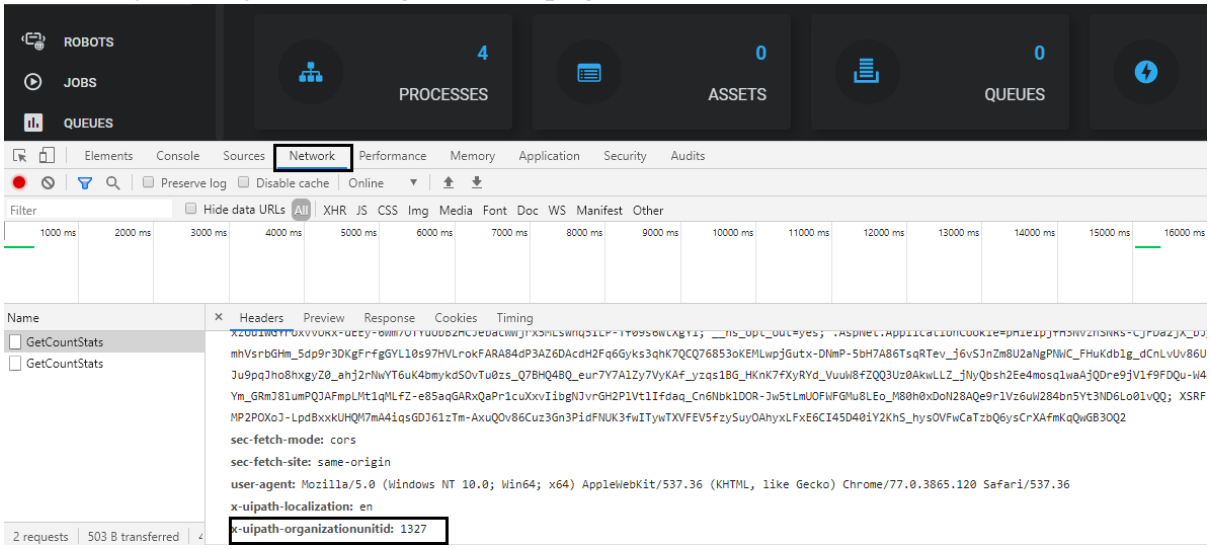
Kindly follow the steps in below link to generating the “refresh_token” and to fetch the values of “account_logical_name” and “service_instance_logical_name” (need to use postman to get the details).

https://docs.uipath.com/orchestrator/v2019/reference#consuming-cloud-api

Process name is the name of your process with environment name(processName_environmentName)

Easiest way to fetch organization unit id is using orchestrator.

1. Open the developer tools (F12) when you're logged in to Orchestrator
2. Make any call by accessing another page in Orchestrator



3. Click on the name and scroll down on the headers tab.

APPENDIX B: INDIVIDUAL REPORT

Name: Aniket Mohan Arasanipalai

Personal Contribution to Group Project:

Aniket was involved in the project idea, brainstorming, and in the creation of the Robotic Process Automation Processes which automated the application filling process, which was the core aspect of this project. Along with this, he created the project report as well as provided content for the video of the project

What Have You Learnt:

As part of the course and the team project, Aniket has learnt about the theoretical and practical aspects of Robotic Process Automation and the different business scenarios where it could be useful

How can you apply the Knowledge and Skills in Other Situations:

The knowledge and skills acquired can be applied to various business scenarios based on the need and requirement.

Name: Akshay Sachdeva

Personal Contribution to Group Project:

Akshay was involved in the project idea, brainstorming and creation of the Robotic Process Automation Processes which automated the application filling process, which was the core aspect of this project. Along with this, he created the project report as well made the video presentation of the project.

What Have You Learnt:

As part of the course and the team project, Akshay has learnt about the theoretical and practical aspects of Robotic Process Automation and the different business scenarios where it could be useful

How can you apply the Knowledge and Skills in Other Situations:

The knowledge and skills acquired can be applied to various business scenarios based on the needs and requirements of the industry.

Name: Sidhant Naveria

Personal Contribution to Group Project:

Sidhant was involved in the project idea, brainstorming and providing guidance in creation of the Robotic Process Automation Processes which automated the application filling process. He is involved in deployment process of code to the orchestrator and creation of script to make API calls to the orchestrator to execute the process.

What Have You Learnt:

As part of the course and the team project, Sidhant has learnt about API integration of applications with the Uipath orchestrator and has developed an understanding about the Orchestrator as a tool for controlling the automation process

How can you apply the Knowledge and Skills in Other Situations:

The knowledge and skills acquired can be applied to various business scenarios based on the needs and requirements.

Name: Vishaag Suriyanarayanan

Personal Contribution to Group Project:

Chatbot - Dialogflow NLP Agent Creation, Dialogflow Fulfilment Backend (python - flask), Facebook Messenger Integration

What Have You Learnt:

Vishaag has learnt to setup a complete chatbot using Google Dialogflow and integrate with external services such as facebook messenger. Vishaag has also learnt to set up a backend server using flask, a python framework.

Vishaag has got an understanding of how to set up an RPA project in UiPath and trigger it remotely using UiPath Orchestrator.

How can you apply the Knowledge and Skills in Other Situations:

Chatbots provide an easy way to interact through a smartphone and useful in customer support situations. Vishaag can use what he's learnt in this course to build RPA systems that many customers can interact with using a chatbot. Vishaag can build services to fill forms, scrape websites for information etc. to rid of repetitive processes.

In his previous work, Vishaag could have built such a system to start application build processes (jenkins etc.) through the chatbot and return with the response state.