A Text Based Conversational Agent for a Higher Education Institute

C. N. Malalasena.

2019

A Text Based Conversational Agent for a Higher Education Institute

A dissertation submitted for the Degree of Master of Information Technology

C. N. Malalasena

University of Colombo School of Computing

2019

Abstract

As more and more businesses turn to doing business virtually through World Wide Web, providing best experience to their website visitors has become an important factor. Even though interacting with a website is a familiar task for many, it is a comfort to have a human like conversation with a customer care agent representing the company when a virtual company is visited. This requirement has given high importance to Conversational Agents also known as ChatBots which are the software that communicate with humans, in a natural language, like a real person. This document describes the study of available technologies to develop a Conversational Agent for an Educational Institution which provides Undergraduate Degree Programs. Then develop the desired product with suitable features and Evaluate its performance.

As chatbots has become latest addition to the software business, there are many organizations providing their user friendly product to implement a chatbot for you, on your own, initially for free. Or you have the Open Source chatbot frameworks like RASA under GNU General Public License. This project implements a chatbot using RASA Framework, along with Python as the programming language and the database server MongoDB.

The chatbot is capable of greeting the visitor upon arrival to the chat window, and offer assistance by providing information regarding the available degree programs and answering other general questions that a visitor might ask in the first visit or the phone conversation. It is also capable of extracting bits of information like contact numbers, names, from the text typed in by the visitor and store them in the Database.

Keywords : Conversational Agent, Chatbot, Natural Language Processing, Natural Language Understanding

Declaration

The thesis is my original work and has not been submitted previously for a degree at this or any other university/institute.

To the best of my knowledge it does not contain any material published or written by another person, except as acknowledged in the text.

Student Name: C. N. Malalasena.

Registration Number: 2016MIT042

Index Number: 16550426

_____ _____

Signature: Date:

This is to certify that this thesis is based on the work of

Mr. C. N. Malalasena.

under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Certified by:

Supervisor Name: Dr. Ruwan Weerasinghe.

_____ _____

Signature: Date:

Acknowledgement

I would like to express my sincere gratitude to Dr Ruwan Weerasinghe for directing me to do this project, guidance provided and allocating time for me, and all the lecturers who lectured us in University of Colombo Schools of Computing during my study of Master of Information Technology, all the lectures and other staff in University of Colombo when I was an undergraduate student.

Also I thank the management of Informatics Institute of Technology for the opportunity offered to follow the postgraduate programme, and also providing information that is used for this project.

My heartfelt gratitude to my family for the encouragement and the support given in last two years allowing me to find time for studies.

Table of contents

List of figures

List of Tables

List of Abbreviations

API – Application Programming Interface

BIS – Business Information Systems

MD – Markdown

NLU – Natural Language Understanding

NLP – Natural Language Processing

YAML – YAML Ain’t Markup Language

1. Introduction

Conversational Agent also known as Chat Bot is a software that is capable of communicating with humans in a natural language like English, in human like manner. The purpose of this automated chat for a human may be from just an entertainment to pass time, or something important as scheduling an appointment.

This document presents the Project "Text Based Conversational Question & Answering Agent for private institutions" which studied and developed a Conversational Agent for a private University in Colombo, as part of completing Master of Information Technology, at University of Colombo School of Computing.

1.1. Problem Identification

Most of the companies do not respond to phone calls outside the office hours. Therefore, the only way for a potential customer to get information during odd hours is the company website. Then the visitor has to browse through web pages and find what they want. It is possible that a valuable potential customer may turn elsewhere if they fail to see what they are looking for within first two to three minutes.

Even during office hours where staff is busy serving the customers who is there right in front of them, it's possible that the customer asking for information over the phone, email or manual chat systems may not get quick responses as fast as they expect.

Therefore, the problem is Not having automated response system that can serve information like company human staff does.

1.2. Motivation

Many private organizations are interested in reducing the number of employees and the employees are also focusing on better employment opportunities rather than covering odd work shifts doing the same thing over and over again.

This creates the opportunity to automate repetitive tasks. It is a great achievement if it is possible to successfully replace the experienced customer care agent chatting with the customer without customer realizing any difference.

The demand for text based chat answering services are on the rise, as customers are busy with their duties, not having time for their own personal work. So more people are tempted to use text based chat customer care services under the office table, while giving priority to their job role at the office above the table.

More people make use of their time while travelling stuck in a vehicle, to find required details. Since they are in the middle of the crowd in the vehicle, privacy is a bit of a concern. Or in situations where they are supposed to be silent and don't want others to be disturbed by a phone conversation. These are the indications that people like to use a silent answering mechanism rather than speaking out the questions loud.

These reasons indicate that there is a high demand for human like chat systems and this demand is not going to go away until another mechanism is invested that they can communicate silently in a user friendly human like manner.

1.3. Aim and Objectives of the project.

The aim of the project is to, develop a solution that is based on an automated Conversational Agent, to satisfy information requests reaching Institutions' website in real time, as if a human is answering the questions.

Objectives of the project are,

1. Study existing requirements of conversational agent requirements of Institutes.
2. Study evaluate different approaches for designing & developing conversational question & answer agents.
3. Design & Develop a conversational question & answer agent that can answer the questions asked by the visitors of the institutions' website.

1.4. Scope of the Project

This project develops a solution that answers questions asked by the user, based on information available in a previously determined text. This project is limited to,

1. An Automated text based Conversational question & answer Agent.
2. Support one language, most probably English.
3. Questions answered are most likely to be general questions that are asked by visitors who are still not registered as students. Therefore, the answers will not be specific to an individual.
4. The selected institution for requirements gathering is a private institution in Colombo, which offer undergraduate degree courses.

1.5. Summary

This chapter documents the initially identified problem, and the motivating factors that drives this project to develop a solution for this problem. It also documents the aim, objectives and the scope.

The rest of the chapters in this document will cover the study conducted about the institutes (Chapter 2). Findings of this study which includes general expectations of an institute, types of information requirements from visitors and the phrases used by them to ask a question.

The design of the Conversational Agent and the development details are also explained (Chapter 3).

This document also includes details of current Conversational Agent technologies available as proprietary / open source solutions (Chapter 4).

The evaluation of the developed Conversational Agent is conducted and the results are documented (Chapter 5).

2. Background

The way that the institute handle information requests currently is studied, requirements of the institute are gathered, study of chatbot building tools, their features, short comings, costs are done and documented in this chapter. Existing Conversational Agents used by other organizations and their capabilities are also documented. Considering the findings, software tools to build the chatbot is selected. Available resources and the cost effectiveness were also major influential factors for the selection criteria.

The design of the user interface, and the opening dialog flow is also given.

2.1. Requirements

The expectations from a text based conversational, question and answering agent that gives information on behalf of the institution was studied and followings stake holders in Figure 2.1 Stake Holders of the Conversational Agent System were identified. Their further details are listed in Table 2.1 Stake Holders of ABC Chatbot.

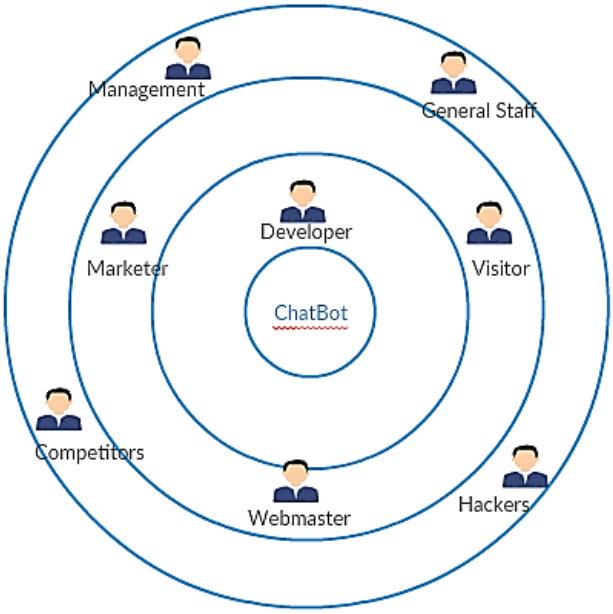


Figure 2.1 Stake Holders of the Conversational Agent System

Layer	Name	Description	Interests
1	Developer	Author will be the developer who develops the ChatBot	Successfully design and implement a working ChatBot that can be put to practical use
2	Marketer	Marketers of the institute	Finding contact details of the visitor who were interested in courses conducted
2	Visitor	People who visit the website and use chat option	Finding the information that they want to find in a conversation as if they were talking to a customer care agent.
2	Webmaster	Staff member who maintain the website	Uninterrupted smooth functioning of all elements of the website.
3	Management	Management of the institution	Reducing staff and cost, getting more revenue
3	General Staff	Staff of the institute in departments other than marketing & management	Accuracy of the details provided and reducing number of inquiries arriving at their desk
3	Competitors	Other institutions competing	Analyzing the success and effectiveness of the ChatBot in order for them to be competitive in the market.
3	Hackers	Anyone interested in finding a weakness	Finding weakness in order to help improve, disrupt services or make fun of any amusing conversations.

Table 2.1 Stake Holders of ABC Chatbot

In order to collect further requirements of Visitors, Daily information requests that reached the website of an institute, from visitors, were thoroughly studied.

Customer care officers of the Marketing staff were interviewed to find how they provide information to the visitors. These one to one interviews reviled following areas of concerns related to finding more information.

1. There is an allocated customer care officer for each course.
2. Visitor who look for details in the website can leave messages stating the course that they are interested in along with their contact details. Collection of such messages are available.
3. Style of questions & wording that the visitors / callers use to ask questions.
4. The contact number and the name of the potential customer is the most vital information that they need to proceed with the process. Once that information is found the rest of the process is very customized based on the potential customer and should not be handled by an automated process.

With the above knowledge following are the tasks carried out to gather more details,

1. Analyzing short messages left in the institute website by visitors.
2. Analyzing information provided by Marketers.
3. Studying the details available in the institute website, marketing materials and other documents.

By doing so, following retirements were recognized,

2.1.1. Functional Requirements

1. A visible link that attracts visitors to chat window, which expands when clicked.
2. Anonymously proceed with the chat to get information, and provide their details later on if they like.
3. Leave a message with contact details.
4. Ability to get general information (See table 1 for further details.)
5. Get contact details of responsible person for a specific task.
6. Identify Names, Phone numbers, email addresses, educational qualifications, provided by the visitor.
7. Identify Sinhala, Tamil language and support translators.
8. Ability to handle inappropriate words & sarcasm in a polite way.

9. User friendly mechanism to enter general details about the courses, contact details of specific people, and general information.
10. Human like behavior, having a name, avatar.
11. Suggest the next step with each reply and direct the conversation

2.1.2 Non Functional Requirements

1. Identify and prevent spam.
2. Use minimum screen space.
3. Situation awareness of the questions asked.
4. Personality.
5. Human like interacting conversation.
6. Handle wrong English, words spelled in trendy ways.
7. Reduce time to get required information compared to browsing the website.
8. User friendly interface visible to type the question at all times.

2.1.3 Categorization of Information provided

Requirement Number	Relevant Department	Information Category
1	Academic	Course Contents
2	Marketing	Facilities
3	Marketing	Location and Direction
4	Sports	Sports
5	Accounts	Payments
6	Marketing	Degree Programs
7	Marketing	Awards won

Table 2.2 Categories of Information Provided

2.2. Features required in the System

Following features are identified as the required feature in the end product.

1. A simple interface with a text field indicating the chat option is available in all pages of web page.
2. Persona that represent the characteristics of the institution
3. Ability to answer questions related to,

a. General information about courses
b. Specific information related to a course
c. Identify the information provided by the user. (Name, Contact number and the course interested)

2.3. Existing Technologies to build Conversational Agents

There are several systems that can be used to implement conversational agents. These technologies are provided by various solution providers with various features. Following is a comparison of such available technologies that are available to develop a conversational agent.

2.3.1. Dialog Flow

DialogFlow was earlier known as Api.ai from Google. DialogFlow can be used to create once own nonlinear chatbot.

Its plus points are, Ability to do NLP, having already built agents, ability to integrate other platforms, support for various languages and most importantly its free.

DialogFlow does not provide much information to customize the product in the code level. (dialogflow.com/docs)

2.3.2. Wit.ai

Wit.ai is owned by Facebook. Wit.ai allows processing human languages and extracting intents and context. Wit.ai facilitates creating intents, entities and agents.

Wit.ai is also capable of NLP, tutorial support, integrating different platforms like Facebook, Twitter, allows creating own chatbots with Node.js, Python or Ruby.

Minus points about Wit.ai is that it does not provide visual development environment, and requires user to know coding. (wit.ai/docs)

2.3.3. IBM's Watson Assistant

IBM's Watson Assistant allows even nontechnical user to build conversational solutions. It provides graphical interface, ability to set dialogs and entities, support for various languages.

Initially its free during the initial trial, but later on will require to pay per message.

2.3.4. Converse.ai

Converse.ai allows users to build chatbots without having to code. It can work with other apps like Salesforce, Stripe, Slack, Paypal. It also supports query and analytics engine that allows tracking how people interact with the chatbot.

2.3.5. Botstart

Botstart allows users to design and develop chatbots and train them online. Botstart chatbots can run in multiple platforms simultaneously and allows integrating with other software. Botstart allows the users to focus more on the conversation and language rather than coding.

Botstart is free allowing unlimited number of messages, with limited number of tags and subscribers. Once the limit is exceeded, payments have to be made based on usage.

2.3.6. Landbot.io

Landbot.io allows users to convert a website in to a chatbot. It also allows integrations with Facebook messenger, WhatsApp, live chat with human assistance features. This provides interfaces that allows to start off without coding.

2.3.7. Flow OX

Flow OX provides a graphical platform to create a chatbot for websites, Facebook, SMS, Telegram and Slack without coding. Allows integrations with many other platforms.

2.3.8. Chatfuel

Chatfuel allows user without programming skills to create chatbots. It provides user friendly interface, free up to 5000 subscribers, move users to another block based on criteria, cloning messages or entire bot, provide multimedia, NLP features, pull information from users facebook account, tools to promote itself, analytics dashboard.

2.3.8. RASA

RASA is an open source machine leaning tool developed by a community of developers. RASA NLU provides natural language understanding which allows extraction of entities and identification of intents. RASA CORE allows developers to associate actions for intents. The main advantage in RASA is that it is open source and supported by large community of users. Some of the glitches still exists which are currently handled by the open source community and are improving fast.

2.4. Summary

This chapter documents the identified stake holders and their interests. It also documents requirements of the identified stake holders. Based on these requirements, expected features were identified. These features are listed as functional requirements and non-functional requirements.

This chapter also documents the existing products and services that allow creation of conversational agents, and their features provided. It is identified that most of the vendors provide their service free but eventually will ask for payments. And also it is noted that some easy to use products lacks customization and support for code level changes. Therefore, among such products RASA is identified as a free open source conversational agent development product that is not that user friendly but allows us to customize as required.

The final outcome of the work documented in this chapter is that, Functional and Non Functional requirements, selections of RASA as the suitable free customizable conversational agent development tool.

3. Design

Designing a conversational agent is different from designing a usual software that interact with users. This chapter focuses on important areas in conversational agent designing. The user interface of a chatbot, conversational flow and the phrases that the chatbot speaks play an important role. This chapter includes the user interface suggested, the structure of the conversation flow and also the selection of the operating system are made.

3.1. User Interface

Conversational agents' user interfaces are currently matured to a level where there is a standard interface that every computer user expect to see. The usual chatbot user interfaces with text box to type user's messages, large area above that to display the conversation back and forth, usually left side displaying the text from the chatbot and the right side displaying the text typed by the visitor has become the standard. The same interface is adopted for the conversational agent.

The interface and the initial conversation of the proposed solution are designed to appear as follows.

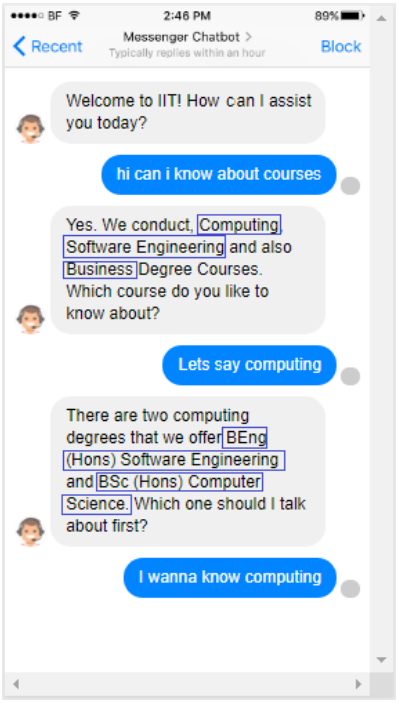


Figure 3.1 User Interface of the ChatBot

3.2. The Flow of the Conversation

The flow of the conversation is represented in the following diagram.

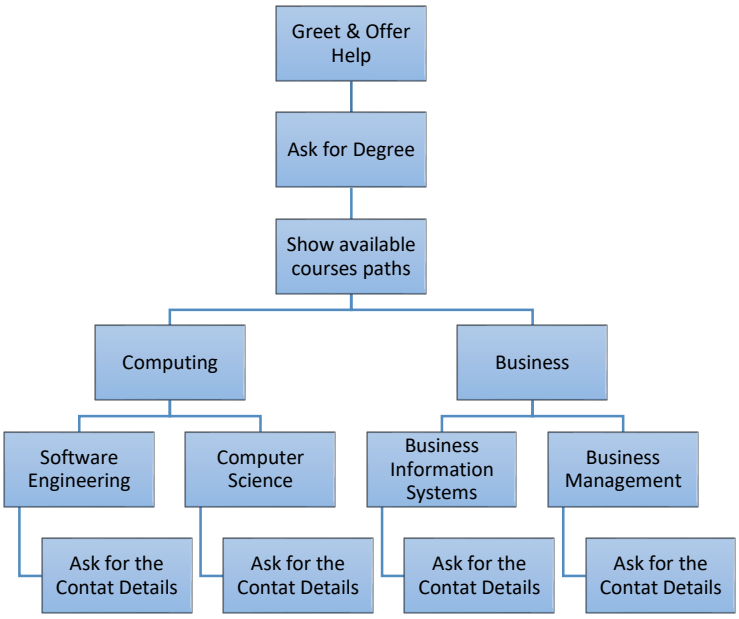


Figure 3.2 Flow of the Conversation

Chatbot will display a greeting message and direct the conversation to available course paths in the institute, forcing the user to select a course path to talk about. Then the ChatBot should display the available degree courses in the selected path. Thereafter the user can select the interested degree to get more details. Chatbot will provide a description about the selected degree program and ask for the users contact details so that further information can be provided. Chatbot should constantly be listening for the user name and the contact number to record it in a database.

Apart from above main flow of chat, there may be other short conversations that may take place independently about other information without a structured flow.

3.3. Hardware & System Software

To implement a chatbot in a website it does not require any additional hardware. Existing website can hold the front end of the chatbot.

There are various tools to implement chatbots. More and more are becoming available as the chatbot market seems to be expanding. Three main concerns when selecting the tool to develop the chatbot for this specific requirement is,

- Amount of customization that is supported
- Free long term use

- Not having to upload / share information

3.3.1. RASA Framework

Therefore, among the main candidates RASA Framework stands in the front. RASA provides free open source support for dialogue management, intents identification, identify entities from the text, complete freedom to control data, link other APIs and the RASA community support to solve technical difficulties.

3.3.2. Programming Language

Python is the primary language that is used for implementing ChatBots. The community of developers involved in chatbot development is also using Python. Therefore, Python is used as the Programming Language.

3.3.3. Code Editor

Notepad ++ is used as the code editor. Notepad++ is lightweight tool that does not take much time to execute.

3.3.4. Dependency Management

Anaconda Prompt and the Anaconda Navigator are used for executing Python files and maintaining Python Virtual Environments.

3.3.5. Database

Python works with most of the popular databases. But considering the freedom that MongoDB offer, it is used as the Database to store information.

MongoDB Compass tool is used to create the database and manipulate the data.

3.3.6. Operating System

As above selected software tools are having versions supporting Windows OS, and sufficient amount of tutorial and online community support is also available, it is decided to use existing Windows 10 running laptop available for the developer.

3.4. Existing ChatBots and their features

There aren't any institutes that use Chatbots in Sri Lanka at the moment. Most of the institutions are not using the chat facility. Few are using manual chat system where a customer care officer reply from the other end which is not available during off office hours.

Sampath Bank has ChatBot in their website which is capable of directing to the information asked for. It does not provide any actions apart from providing the

information and links to information in their website. Other banks maintain manual chat or no such facility at all.

One of the best attempts to run automated ChatBot to represent a company is Coddy. The chatbot from CogCom.ai. Which answers general questions about the company and provide their details in remarkably human like conversation. Even though its not capable of providing answers for all the questions asked, most of the relevant questions are well answered.

3.4.1. Summary of Chatbot Features

The common features found in the Chatbots found are,

1. Appears on the right lower corner as a small banner which expands if clicked. Minimizes at the second click on the banner.
2. Attracts the attention of the visitor to the link.
3. Uses simple short answers.
4. Provides links to webpages rather than trying to extract the information and display in the chat window.
5. Most of the ChatBots provide text output. Some provides audio output as well.
6. Whenever the accuracy of the understanding of the question is doubtful chatbot refrain from answering the question and ask the user to rephrase and ask again, rather than taking the risk.

4. Methodology

4.1. Introduction

This chapter discuss the implementation details of the Text Based Conversational Question & Answering Agent. The core of the conversational agent that is implemented and the associated conversations that are used to train, are documented here.

4.2. Structure of the System

Following technologies are identified and selected as appropriate tools.

- RASA Framework for the Natural Language Understanding (NLU), creating response text, execution of actions as required.
    - RASA NLU is used to understand the users input and to extract the Intents and the Entities.
    - RASA CORE is used to do the execution of actions in functions, as a response for users input.
- Json files were used to store sample data used for training. But later on as technology changed, Markdown file is used to structure the training data of the chatbot. Example conversations are stored in .md files, which will be used by RASA NLU.
- Anaconda Navigator and Anaconda Prompt are used to install and manage Python libraries and dependencies.
- Node JS and npm package manager is used to install rasa-nlu-trainer which is used to enter and mark intents, entities in training conversation at the beginning when Json files were used to store the sample conversations.
- Python 3.5 as the programming language.
- NotePad++ is used for Python coding.
- MomgoDB is used as the Database.

4.3. Installation and Setup

Following software were used for the development of the chatbot.

- Windows 10 operating system (But Linux seems to be a better option where things will run much better without hassle. This will be considered in the future.)
- Anaconda Prompt (To install and manage Python versions, Virtual Environments and Dependencies)

- Visual C++ Compiler (As some older versions of RASA required this.)
- Python dependencies recommended by RASA(Listed in requirements.txt in Appendix A)
- Download and install spacy language model

Following Figure 4.1 Rich Picture of How RASA works shows how the components interact with each other,

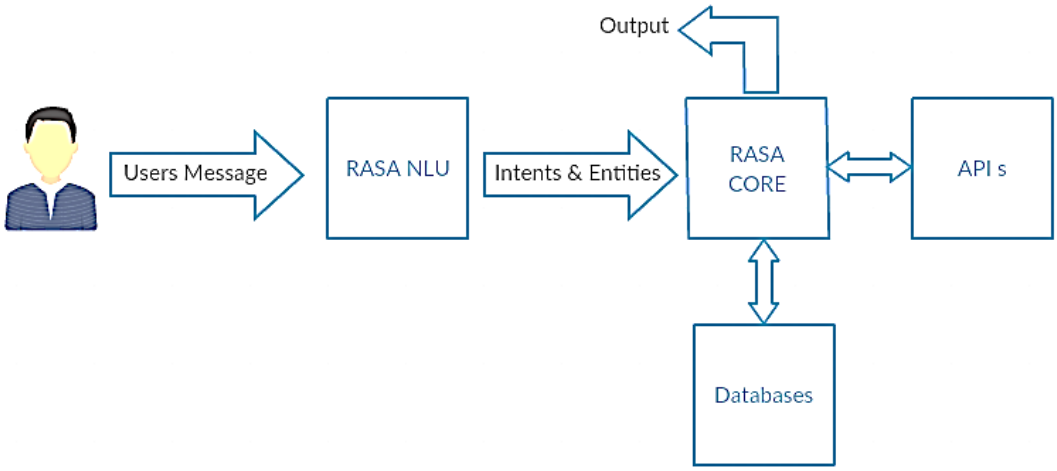


Figure 4.1 Rich Picture of How RASA works

RASA NLU can extract the meaning out of the message received from the user. These meaning will be extracted as Intents and the Entities. Intent describes user expectations and the Entities are the various values that are in the user message, for example users contact number.

Extracted Intents and the Entities can be used to decide how should the chatbot respond. RASA CORE will perform actions necessary to communicate with databases or other APIs (if necessary) to get the necessary information to create the response.

4.4. Design of Database

It is considered to have the up to date information required for the chatbot, in a database. MongoDB can be used for this purpose as it provides flexibility in structure.

Following Collections (Tables) are suggested to keep data required for the chatbot to give a proper reply.

- Courses

  This collection holds the details of the courses that RASA Core will use to formulate the reply when visitor ask for course details. It contains an auto

generated id, coursed code, course name, description, entry requirement, target jobs and duration.

- Visitors

  This collection will store the visitors details along with the course that they were interested in, the date & time. This collection contains auto generated id, name, contact number, course, date and time.

4.5. Code

Following components get together in creating the implemented chatbot.

- Python 3.5 virtual environment.
- Python dependencies installed
- Code in files
  - Markdown file that keep sample training data. (nlu_data.md)
  - YML file containing configuration file that keeps configuration parameters (nlu_config.yml) This is where it specifies what pipelines.
  - YAML file that specify the domain for the chat bot (domain.yml)
  - Python file where actions to perform are written (actions.py)
  - Markdown file that maps the intents and appropriate response (stories.md)

Following describes the coding that is developed to implement the prototype.

4.6. Sample Training Data

Chatbot need to know what users may say, what they may be trying to do (intent), and what information that should be kept in mind (entities). nlu_data.md file contains samples of text that user may use to express what they want, labeled with intents and entities.

Contents of nlu_data.md file is listed in Appendix B.

4.7. Configuration Parameters

Configurations file nlu_config.yml will tell the chatbot what RASA NLU pipeline to use. Spacy_sklearn has pre trained word vectors. A word vector tries to do the mapping between words by calculating the probability, how likely each term is related to another, which gives a mathematical meaning for each word with respect to others among which they appear.

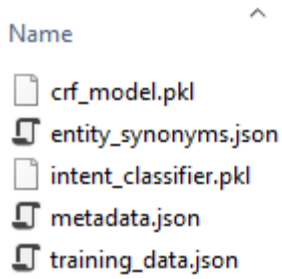
nlu_config.yml is listed in Appendix C.

4.8. Creating the Trained Model

The RASA Framework of code takes the above mentioned nlu_config.yml, nlu_data.md and the path in which the model has to be created as parameters.

```
python -m rasa_nlu.train -c nlu_config.yml --data data/nlu_data.md
-o models --fixed_model_name nlu --project current --verbose
```

The model created by the Python Trainer will create the model in the given destination. This model will consist following files.



4.9. Domain of the Chatbot

The domain.yml file consist of Entities, their data types, list of Intents, templates where responses for each intent is listed and actions that need to be executed as the response for certain intents.

domain.yml is listed in Appendix D.

4.10. Actions to perform by chatbot

Some entities need further actions rather than just a text respond, like retrieving data from database or API call to outside system. These are programmed in separate classes in actions.py file.

The class ActionProvideDegreeDetails is used to get the details of the degree that the visitor asked for. Available details of the degree will be retrieved from the database and display as a description.

The available courses are taken from the database and displayed to the visitor when asked for What courses or degree programs are available. This is done by the ActionProvideDegreeNames class.

Code in above classes are listed in Appendix E.

4.11. Stories

The stories.md file connects intents with correct set of responses which are known as utterances and actions. Then one of the replies from the correct utterance in the domain.yml will be sent to the user as the reply by the RASA Agent. Or if it is an action that is listed in the stories.md for the given intent, the correct action will be performed.

4.12. Sample outputs

4.12.1. Intermediate NLU Output

Following output show how the chatbot understood the given user inputs. These outputs are generated by the code in nlu_model.py which create the NLU model and select the best Intent and extract entities according to the model created.

Input: hello

```
{'text': 'hello', 'intent': {'confidence': 0.7423977000652854, 'name': 'greet'},
'intent_ranking': [{'confidence': 0.7423977000652854, 'name': 'greet'}, {'confide
nce': 0.13670082207421097, 'name': 'thank_you'}, {'confidence': 0.067519049980920
15, 'name': 'tell_interested_degree'}, {'confidence': 0.04065381046899304, 'name'
: 'exit_greet'}, {'confidence': 0.012728617410590275, 'name': 'ask_degrees'}], 'e
ntities': []}
```

NLU model created identify the user is trying to greet with probability of 74%.

Input:bye

```
{'entities': [], 'intent': {'name': 'exit_greet', 'confidence': 0.789383432730614
3}, 'text': 'bye', 'intent_ranking': [{'name': 'exit_greet', 'confidence': 0.7893
834327306143}, {'name': 'greet', 'confidence': 0.1099273751698558}, {'name': 'tha
nk_you', 'confidence': 0.07421192763424986}, {'name': 'tell_interested_degree', '
confidence': 0.020274165458451465}, {'name': 'ask_degrees', 'confidence': 0.00620
30990068289975}]}
```

Model identify with 78% probability, that the user is trying to finish the conversation with a greeting.

Input:thank you so much

```
{'entities': [], 'intent_ranking': [{'name': 'thank_you', 'confidence': 0.4379703
1667268554}, {'name': 'exit_greet', 'confidence': 0.252768255328359}, {'name': 'g
reet', 'confidence': 0.14992874448226765}, {'name': 'tell_interested_degree', 'co
nfidence': 0.08129454939629145}, {'name': 'ask_degrees', 'confidence': 0.07803813
412039674}], 'text': 'Thank you so much', 'intent': {'name': 'thank_you', 'confid
ence': 0.43797031667268554}}
```

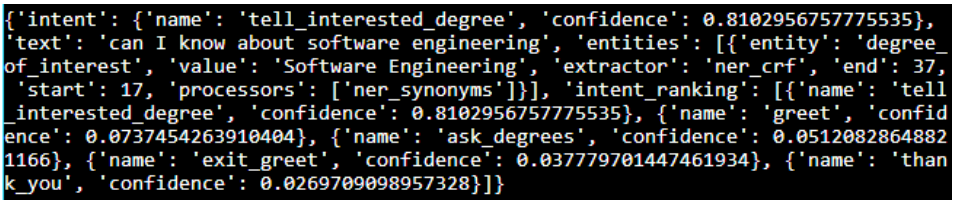
Model identify with 43% probability, that the user is trying to thank the chatbot. Here the probability of the right intent is very low. This should be improved with more training data.

Input:what are the available curses?

```
{'intent': {'name': 'ask_degrees', 'confidence': 0.3460282989887127}, 'text': 'wh
at are the available courses?', 'intent_ranking': [{'name': 'ask_degrees', 'confi
dence': 0.3460282989887127}, {'name': 'greet', 'confidence': 0.33741731979913975}
, {'name': 'exit_greet', 'confidence': 0.18446596600061496}, {'name': 'tell_inter
ested_degree', 'confidence': 0.09867346708340578}, {'name': 'thank_you', 'confide
nce': 0.033414948128126905}], 'entities': []}
```

Model identify with 34% probability, that the user is asking for course details. Here the probability of the right intent is very low. This should be improved with more training data.

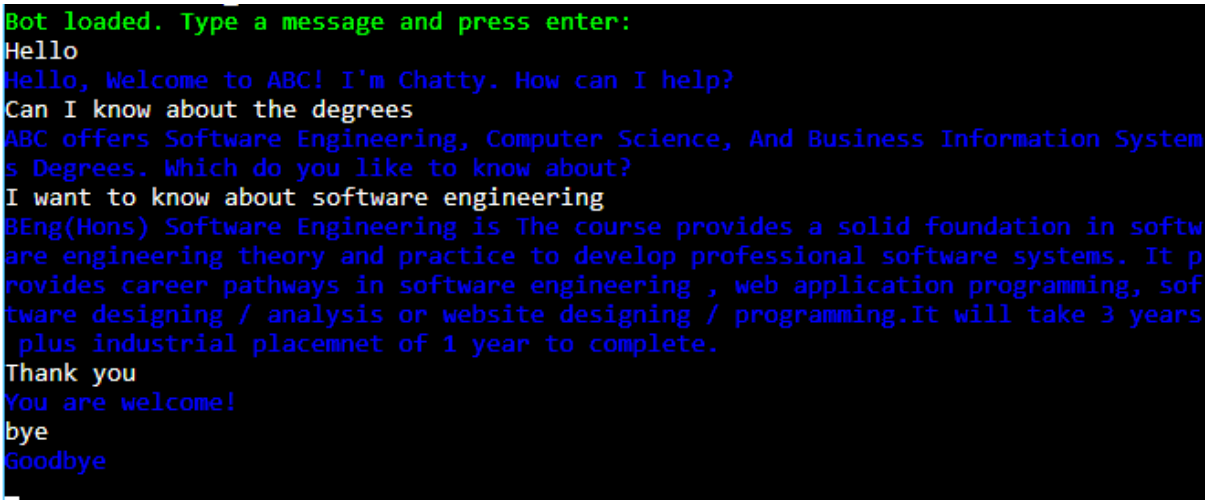
Input:can I know about software engineering



Model correctly identify that the user is trying to express interested degree that the user wants to know about with the probability of 81%. It also identities the entity "Software Engineering"

4.12.2. Little conversation with the chatbot early version

Following is a small conversation with the chatbot via Anaconda Prompt. It correctly replies to greeting and offer Help. Correctly identify the intent that the user wants to know about the degree programs. Show available degrees. Correctly responds to the question by identifying the intention and the entity "Software Engineering". It also finishes the conversation naturally.



4.13. User Interface

User Interface for the chatbot is implemented using the framework provided by scalableminds, a community of programmers. The open source product called chatroom

developed by them are available for RASA framework. It is customized and used to develop the User Interface.

Followings are few screen captures of conversations.

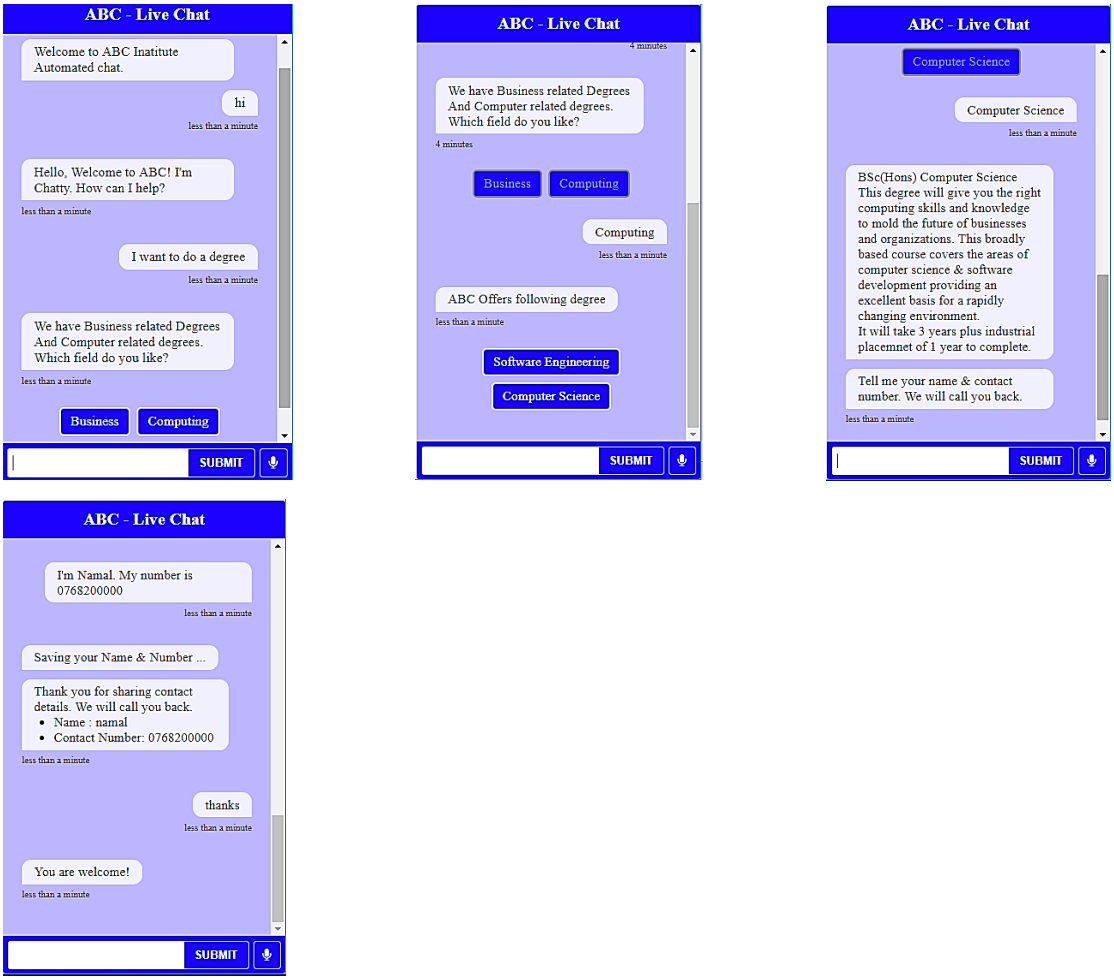


Figure 4.2 User Interface

Conversation Start, Select Path, Select the degree, Provide the name & contact number

Outputs from other short conversations are listed below.

Figure 4.3 UI - Short Conversation

5. Testing and Evaluation

5.1. Introduction

The chatbot is tested to assure the accuracy of the responses that it provides for specific anticipated questions that the visitor may ask. It is also tested to evaluate its capability to identify the important contact information of the visitor, that is the name and the contact number.

5.2. Testing Objectives

Following areas are identified for testing, based on the requirements identified in the requirement gathering stage.

- Short Replies & Human like sensible answers.
  Chatbot will be tested to see how chatbot reply to messages that expect short greetings like answers. E.g. Good morning, Bye. Whether the chatbot reply with meaningful logical answers.
- Correct Intent identification
  Intent is the reason for visitor to ask the question. Or the objective that the visitor expects to fulfill by asking the question. Chatbot should correctly identify the intents.
- Correct Entity identification
  Entities are the small facts that are extracted from the text that is coming from the visitor. For example, chatbot should be able to identify a contact number in an answer given by the visitor.
- Fallback replies.
  Fallback replies are the answers that the chatbot should give for messages that is not programmed to answer. Basically these are the intents that are not handled or text that which fails to map in to existing intent.
- Navigation
  Ability to direct the conversation to a successful path in order to keep the correct flow of the conversation rather than ending up in a dead end.
- Store contact details in the Database.
  It is required to store the name and the contact number of the visitor in a database if they provide it in the conversation.

** Following areas of requirements are only planed but not tested as such features are not implemented yet.

- Sinhala Tamil conversations
- User friendly interface to update database
- Display contact details of specific people in staff

5.3. Testing Plan

Black Box tests are planned to see whether the expected output is given by the chatbot. Following Test Cases are designed to achieve above objectives.

Index Number	Objective	Test Cases Number	Test Cases
1	Short Replies & Human like sensible answers.	1	1.1 Welcome greeting message
			1.2 Farewell message
			1.3 Reply for Thanking
			1.4 Ask for location
			1.5 Ask about payments
			1.6 Ask about Available facilities
2	Correct Intent Identification	2	2.1 Ask for Degree courses
			2.2 Tell the degree interested in
3	Correct Entity Identification	3	This is tested in 6. Storing Contact Details
4	Fallback Replies	4	4.1 Fallback Replies
5	Navigation	5	5.1 Navigation path for Software Engineering degree
			5.2 Ask about the Computer Science Degree
			5.3 Ask about the Business Information Systems Degree
			5.4 Ask about the Business Management Degree
6	Store contact details	6	6.1 Storing Contact Details

Table 5.1 Test Plan

Above test cases will be completed with following input values and the expected result is compared with the actual output. If it is sufficiently matched, Test result is considered success.

5.3.1 Testing Welcome greeting message

Test Case Number	Feature	Input	Expected Output	Result
1.1	Welcome greeting message	1.1.1. Hi	"Hello, Welcome to ABC! I'm Chatty. How can I help?"	1.1.1. Success
		1.1.2. Hello		1.1.2. Success
		1.1.3. how are things		1.1.3. Success
		1.1.4. how is it going		1.1.4. Success
		1.1.5. whats up		1.1.5. Success
		1.1.6. hi nice to meet you		1.1.6. Success

Table 5.2 Test Welcome greeting message

According to the above results chatbot can identify visitors greeting messages and the intent greet correctly.

5.3.2. Testing Farewell Message

Test Case Number	Feature	Input	Expected Output	Result
1.2	Farewell message	1.2.1. Bye	"Goodbye"	1.2.1 Success
		1.2.2. Bye bye		1.2.2 Success
		1.2.3. See you		1.2.3 Success
		1.2.4. Catch you later		1.2.4 Success
		1.2.5. Talk to you later		1.2.5 Success
		1.2.6. See you later		1.2.6 Success

		1.2.7. Have a good day		1.2.7 Success
		1.2.8. Take care		1.2.8 Success

Table 5.3 Testing Farewell Message

Chatbot correctly identifies intent exit_greet and the visitor's messages that ends the conversation.

5.3.3. Reply for Thanking

Test Case Number	Feature	Input	Expected Output	Result
1.3	Reply for thanking	1.3.1 Thank you	"You are welcome!"	1.3.1 Success
		1.3.2 Thanks		1.3.2 Success
		1.3.3 Thanks a lot		1.3.3 Success

Table 5.4 Test Reply for Thanking

Chatbot identifies messages related to visitor thanking the chatbot with intent thank_you.

5.3.4. Ask for Location

Test Case Number	Feature	Input	Expected Output	Result
1.4	Ask for Location	1.4.1 Where are the classes	We are at, No, 57, Ramakrishna Road, Colombo 06.	1.4.1 Success
		1.4.2 Where is the location		1.4.2 Success
		1.4.3 What is the address		1.4.3 Success

		1.4.4 What is the location	Tele 0112 360 212	1.4.4 Success
		1.4.5 Tell me the directions		1.4.5 Success
		1.4.6 Where are the classes		1.4.6 Success

Table 5.6 Test Ask Location

Chatbot can answer questions related to whereabouts of the institute. Also identifies intent location successfully.

5.3.5. Ask about payments

Test Case Number	Feature	Input	Expected Output	Result
1.5	Ask about payments	1.5.1 How much will it cost	Payments can be done in installments. Tell me your contact details. We'll call you back	1.5.1 Success
		1.5.2 What are the payment plans		1.5.2 Success
		1.5.3 Can I pay in installments		1.5.3 Success
		1.5.4 How much do I have to pay		1.5.4 Success
		1.5.5 Course fee		1.5.5 Success
		1.5.6 How much is the course fee		1.5.6 Success

Table 5.7 Test Ask About Payments

Questions Related to course fee or other payments can be handled. Also identifies intent payments successfully.

5.3.6. Ask about Available facilities

Test Case Number	Feature	Input	Expected Output	Result
1.5	Ask about available facilities	1.5.1 What are the other facilities available	We have Air Conditioned Lecture rooms & Computer Labs, Library, Study areas, Canteen, Swimming pool, Medical room, Play area.	1.5.1 Success
		1.5.2 What are the available facilities		1.5.2 Success
		1.5.3 What facilities are available		1.5.3 Success
		1.5.4 What other resources are there		1.5.4 Success
		1.5.5 Is there a library		1.5.5 Success
		1.5.6 What are the extra facilities available there		1.5.6 Success
		1.5.7 What facilities are available there		1.5.7 Success

Table 5.8 Test Ask about Available facilities

Correctly identifies questions related to Facilities. Also identifies intent Facilities successfully.

5.3.7. Correct Intent Identification

Tests conducted when testing “Short Replies & Human like sensible answers” in the above test 1.1 to 1.6 confirms the Intents, Greet, Exit_greet, Thank_you, Ask_payment, facilities and Location are properly identified. Apart from these Intents, following intents are also separately tested, and results are listed below.

5.3.8. Ask for Degree Courses

Test Case Number	Feature	Input	Expected Output	Result
2.1	Ask for Degree Courses	2.1.1 Can I know about the degree	We have Business related Degrees And Computer	2.1.1 Success
		2.1.2 I want to know about the degree		2.1.2 Success

		2.1.3 Can you tell me about the degree	related degrees. Which field do you like? <<Computing>> <<Business>>	2.1.3 Success
		2.1.4 Tell me about the degree		2.1.4 Success
		2.1.5 What are the available degrees there		2.1.5 Success
		2.1.6 I'm looking for a degree		2.1.6 Success
		2.1.7 I want to do a degree		2.1.7 Success
		2.1.8 Do you have computer degrees		2.1.8 Success
		2.1.9 Are there degree courses		2.1.9 Success
		2.1.10 Give me degree details		2.1.10 Success
		2.1.11 I want to join the degree		2.1.11 Success

Table 5.8 Ask for Degree Course

Above results show that the chatbot can correctly understand the users intent to know about the available degree programs.

5.3.9. Tell the Degree Interested in

Test Case Number	Feature	Input	Expected Output	Result
2.2	Tell The Degree Interested in	2.2.1 I want to know about software engineering	Show the description of the	2.2.1 Success

	(Here visitor express what degree program that he/she wants to know more about)	2.2.2 I like SE	degree formulated using the data extracted from the database.	2.2.2 Success
		2.2.3 Tell me more about SE		2.2.3 Success
		2.2.4 I want to know about Business Information Systems		2.2.4 Success
		2.2.5 Tell me more about BIS		2.2.5 Success

Table 5.9 Test Interested Degree

Results in the above table show that the chatbot is capable of identifying the intent of the visitor to know about a specific degree program. Also chatbot is capable of correctly identifying the entity degree_of_interest which represents exactly which degree that the visitor is interested in finding out.

5.3.10. Correct Entity Identification

Entities are the facts that may include in what visitor say. For example, visitor may express what his name, contact number, which are entities that the chatbot is supposed to identify and keep track of during the conversation and store in the database.

Following Entities are planned to be tested,

- Degree_of_Interest

  This entity will keep track of the Degree that the user need to know about. When testing “Tell the Degree Interested in” in above test case 2.2, the chatbot successfully identify this intent which held the values “Software Engineering”, “SE”, “Business Infromation Systems” and “BIS”. Therefor it is not tested again.
- User_name

  During the conversation user may express his/her name, that the chatbot is supposed to identify. This will be tested along with the test case “6 . Store Contact details”
- User_contact _number

  The chatbot is supposed to identify user contact number in case it is

mentioned. This too will be tested along with the test case “6 . Store Contact details”

5.3.11. Fallback Replies

Fallback replies are the response that the chatbots should give when a matching correct intent with sufficient probability is not identified. In the policies.yml file nlu_threshold and the core_thresold are set to 0.4 which states that if a visitor message cannot be mapped to an existing Intent with at least 40% probability Fallback Intent should be mapped. Then the reply should be a reply that gently acknowledge the visitor that his question was not understood.

Test case “4 Fallback Replies” tests whether the Intent Fallback is mapped for irrelevant questions.

Test Case Number	Feature	Input	Expected Output	Result
4.1	Ask about some irrelevant or lengthy question	1.4.1 When was this started?	One of the fallback reply get displayed E.g. "Sorry I did not understand what you said."	1.4.1 Success
		1.4.2 Who owns this company?		1.4.2 Success
		1.4.3 How old are you?		1.4.3 Success
		1.4.4 I would like to do any program that is available there as long as it falls only on Sundays and poya days.		1.4.4 Success
		1.4.5 How far is it from Galle?		1.4.5 Success
		1.4.6 Can me and my sister get it in to the same batch?		1.4.6 Success

According to the test results, questions that is not trained to answer are replied with default message all the time.

5.3.12. Navigation

The flow of the conversation is tested to confirm it takes the visitor in the path as in the design. When a visitor shows the interest of following the degree program, conversation get in to a set navigation path as given below.

Chatbot Ask the field of interest for studies >> Show the available fields >> Visitor select / or type interested field >> chatbot asks the degree that is interested >> Show available degrees in selected field >> Visitor select the degree he wants >> Details of the selected degree appears.

This path is available in Figure 3 Flow of the conversation.

Test Case 5.1 Navigation path Software Engineering degree

Test Case Number	Feature	Inputs	Expected Output	Result
5.1	Navigation path	Can I know about the degree	We have Business related Degrees And Computer related degrees. Which field do you like? <<Business>> <<Computing>>	Success
		Visitor select <<Computing>>	ABC Offers following degree <<Software Engineering>> <<Computer Science>>	Success
		Visitor select <<Software Engineering >>	Details of Software Engineering appears	Success

Test Case 5.2 Navigation path Computer Science degree

Test Case Number	Feature	Inputs	Expected Output	Result
5.2	Navigation path	I'm looking for a degree	We have Business related Degrees And Computer related degrees. Which field do you like? <<Business>> <<Computing>>	Success
		Visitor select <<Computing>>	ABC Offers following degree <<Software Engineering>> <<Computer Science>>	Success
		Visitor select <<Computer Science>>	Details of Computer Science appears	Success

Test Case 5.3 Navigation path Business Information Systems degree

Test Case Number	Feature	Inputs	Expected Output	Result
5.3	Navigation path	What are the available degrees there	We have Business related Degrees And Computer related degrees. Which field do you like? <<Business>>	Success

			<<Computing>>	
		Visitor select <<Business>>	ABC Offers following degree <<Business Information Systems>> <<Business Management>>	Success
		Visitor select <<Business Information Systems>>	Details of Business Information System appears	Success

Test Case 5.4 Navigation path Business Management degree

Test Case Number	Feature	Inputs	Expected Output	Result
5.4	Navigation path	I want to join the degree	We have Business related Degrees And Computer related degrees. Which field do you like? <<Business>> <<Computing>>	Success
		Visitor select <<Business>>	ABC Offers following degree <<Business Information Systems>> <<Business Management>>	Success
		Visitor select <<Business Management>>	Details of Business Management appears	Success

According to the above results all four navigation paths are working properly.

5.3.13. Storing contact details

Test Case Number	Feature	Input	Expected Output	Result
6.1	Retrieve available courses	I am Namal I like Software Engineering My contact number is 0776563000	Upon completion of all three inputs in any order record get written to the database along with Date and time	Success

These test results listed above indicates that the extracting Entities related to contact details (name & contact number) works well.

5.4. Conclusion of Test Results

This chapter identifies major areas that need to be tested in the chatbot. Then the test cases are planned, executed and results are compared with expected results. According to the results it is confirmed that the required features are working.

6. Conclusion

The implementation of the project “A Text Based Conversational Agent for a Higher Education Institute” is completed with a study of available technologies to implement conversational agents. It also gathered information requirements of visitors who arrive at a website set up for a private institute in Colombo which offers Undergraduate degree programs. These findings were documented in the Background chapter.

The requirements revealed by the background study are filtered and information requirement categories were identified. Questions asked by the visitors in each of these categories and the usual set of phrases that they use were also identified. Based on these questions conversations agent is trained to identify the correct intent of the question. The intent decides the response that should be given to the visitor. These responses, which are called utterances are also designed based on the information found from the institution under study.

The background study done about the available technologies revealed that RASA is an open source chatbot implementation platform that allows us to create our own chatbots customized for our own needs. The best feature that RASA provides over other vendors is that its free and allows us to run our own chatbot server, without having to share our data.

Design of the user interface, hierarchical structure of the conversation and the answers that the chatbot should give, in each situation are documented in design chapter. Based on the information found, flow of the conversation and replies that the chatbot should give was decided. Set of questions that the visitor may ask and set of appropriate answers are prepared for training model.

Implementation chapter covers how the conversational agent is implemented with RASA natural language processing and RASA core modules. The core requirement identified from the institute is implemented allowing customer care officers to get visitors contact details and the course that they are interested about. So that they can call back once they have time. This makes the implemented system a very practical and useful system for the institute.

Since appropriate open source technologies were identified based on the ability to customize and long term economic benefits during the background study, that is RASA

Framework, Python programming language, MongoDB Database, institute will benefit economically in the long run.

Developed chatbot is tested and evaluated to confirm whether it's core requirements are functioning. Test results show that all the test cases give good results. There are situations that the chatbot fail when questions that are not previously trained are asked.

6.1. Deficiencies & Future enhancements

As there are no perfect chatbot, this conversational agent also has deficiencies that can be observed after chatting with it for a while in a natural free conversation. These can be listed as follows,

- **Limited data set**
  The sample questions and the replies that are used to train the chatbot is currently limited to initial set of information found at the initial study of facts. The questions that the visitor s ask will have to be recorded and periodically updated.
- **Limited number of intents**
  At the moment the chatbot is limited to 13. As more data becomes available, more intents will have to be added.
- **Too many Fallbacks**
  Currently if the questions are asked in too lengthy way, chatbot fails to identify intent with higher probability. This results default reply to be given. The number of times that the default reply is given seems to be a considerable amount. Which need to be reduced.
- **Testing in real sever**
  The testing was carried out in the development Laptop, in which the chatbot and the client both exists. Therefore, it is unknown how it will respond if it is tested over the internet with several users at the same time.

Above deficiencies should be fixed in future. Data become available for the chatbot to learn from after exposing it to the real time visitors.

This conversational agent is a practically usable economical conversational agent that can be trained further every day as it get exposed to more and more users. Therefore, this project can be considered as success as it has achieved project aim.

References

[1] Carlos Segura, Jordi Luque, Rafael E. Banchs and Marta Ruiz Costa-jussa. *Chatbol, a chatbot for the Spanish "La Liga"*, International Workshop on Spoken Dialog System Technology 2018 (IWSDS), At Singapore.

[2] Tom Bocklisch, Joey Faulkner, Nick Pawlowski and Alan Nichol. *Rasa: Open Source Language Understanding and Dialogue Management*, Presented at NIPS Workshop on Conversational AI. Dec 2017

[3] *The MongoDB 4.2 Manual — MongoDB Manual.* (2019) Accessed on: May 2019 [online]. Available: https://docs.mongodb.com/manual/

[4] *Build contextual chatbots and AI assistants with Rasa* (2019) Accessed on: May 2019 [online] Available: https://rasa.com/docs/

[5] *React-based Chatroom Component for Rasa Stac* (2019) Accessed on: May 2019 [online] Available: https://github.com/scalableminds/chatroom

[6] Daniel Braun, Adrian Hernandez Mendez, Florian Matthes and Manfred *Langen. Evaluating Natural Language Understanding Services for Conversational Question Answering Systems*, SIGDIAL 2017 Conference, Saarbrucken, Germany.

Appendix A

Requirements.txt

Requirements.txt file holds the list of python dependencies that need to be installed for RASA to run.

```
## Setup py requirements
jsonpickle==0.9.6
six==1.11.0
redis==2.10.6
fakeredis==0.10.3
future==0.16.0
numpy==1.14.5
typing==3.6.4
ruamel.yaml==0.15.37
requests~=2.20
keras==2.1.6
tensorflow==1.10.0
h5py==2.7.1
apscheduler==3.5.1
tqdm==4.23.3
ConfigArgParse==0.13.0
networkx==2.1
pykwalify<=1.6.0
coloredlogs==10.0
flask==1.0.2
flask_cors==3.0.4
scikit-learn==0.19.1
rasa_nlu~=0.13.0
colorhash==1.0.2
pika==0.11.2
jsonschema==2.6.0
packaging==17.1
gevent==1.2.2
pyyaml==3.12
pytz==2018.4
rasa-core-sdk~=0.12.1
rasa_core~=0.12.0
pymongo==3.5.1
python-dateutil==2.7.3
spacy==2.0.12
sklearn_crfsuite==0.3.6
msgpack==0.5.6
```

Appendix B

nlu_data.md file contain the things that the user may say to the chatbot.

```
## intent:exit_greet
- Bye
- Goodbye
- See you later
- Bye bot
- Goodbye friend
- bye
- bye for now
- catch you later
- gotta go
- See you
- goodnight
- have a nice day
- i'm off
- see you later alligator
- we'll speak soon

## intent:greet
- Hi
- Hey
- Hi bot
- Hey bot
- Hello
- Good morning
- hi again
- hi folks
- hi Mister
- hi pal!
- hi there
- greetings
- hello everybody
- hello is anybody there
- hello robot

## intent:thank_you
- Thanks
- Thank you
- Thank you so much
- Thanks bot
- Thanks for that
- cheers
- cheers bro
- ok thanks!
- perfect thank you
- thanks a bunch for everything
- thanks for the help
- thanks a lot
- amazing, thanks
- cool, thanks
- cool thank you
```

intent:name
- My name is [Juste](user_name) <!--- Square brackets contain the value of entity while the text in parentheses is a a label of the entity -->
- I am [Josh](user_name)
- I'm [Lucy](user_name)
- I'm [Namal](user_name)
- I'm [Jagath](user_name)
- Am [Josh](user_name)
- Im [Lucy](user_name)
- Im [Namal](user_name)
- Im [Jagath](user_name)
- People call me [Greg](user_name)
- It's [David](user_name)
- Usually people call me [Amy](user_name)
- My name is [John](user_name)
- You can call me [Sam](user_name)
- Please call me [Linda](user_name)
- Name is [Tom](user_name)
- I am [Richard](user_name)
- I'm [Tracy](user_name)
- Call me [Sally](user_name)
- I am [Philipp](user_name)
- I am [Charlie](user_name)
- My name is [Namal](user_name)
- I'm [Jagath](user_name) and my contact number is [0776895641](user_contact_number)
- I'm [Jagath](user_name) and my number is [0776895641](user_contact_number)
- I'm [Jagath](user_name) and my number [0776895641](user_contact_number)
- I'm [Jagath](user_name) and contact number is [0776895641](user_contact_number)
- I'm [Jagath](user_name) and number is [0776895641](user_contact_number)
- I'm [Jagath](user_name) and contact number [0776895641](user_contact_number)
- Im [Jagath](user_name). Call me on [0332451285](user_contact_number)
- I'm [Sarath](user_name) you can call me on [0112562389](user_contact_number)
- [Kasun](user_name) [0332562312](user_contact_number)
- [Amal](user_name) [0112784512](user_contact_number)
- [Saman](user_name) [0778524163](user_contact_number)
- Name [Nimal](user_name) number [0772561287](user_contact_number)
- Number [0252879865](user_contact_number) name [Samantha](user_name)

intent:ask_payments
- How much will it cost
- How much is the cost
- How much is the fee
- How much will be the fee
- How much will be the term fee
- How much is the term fee
- How much is the course fee
- What are the payment plans
- What is the payment plan
- What is the payment
- What is the cost
- What will be the term fee
- Can I pay in installments
- How much do I have to pay
- How much money will it cost

- How much do I have to pay
- What is the payment
- Course fee
- How much is the course fee
- How much will be the course fee
- What will be the cost
- What is the course fee

intent:location
- Where are the classes
- Where is the location
- Location
- What is the location
- Tell me the directions
- Where can I find your location
- How to get there
- Tell me how to get there
- Tell me the route to come there
- Tell me the directions
- Tell me the way to get there
- How to get there
- Tell me the directions to get there
- Tell me the road to get there
- What is the address
- Tell me the address
- How can I get there
- Tell me the way to get there

intent:facilities
- What are the other facilities available
- What are the avaialble facilities
- What facilities are available
- What other resources are there
- Is there a library
- What are the extra facilities available there
- What facilities are available there
- What are the other facilities available to us

intent:ask_degrees
- Can I know about the degree
- Can I know about the degree programme
- I want to know about the degree
- I want to know about the degree programme
- Can you tell me about the degree
- Can you tell me about the degree programme
- Tell me about the degree
- Tell me about the degree programme
- Tell me details about the degree
- Tell me details about the degree programme
- Tell me what degrees are available there
- What are the available degrees there
- What are the degrees that you have
- I'm looking for a degree

- I want to do a degree
- I want to start a degree
- Do you have degrees
- Do you have computer degrees
- Do you have Business degrees
- Do you conduct degree courses
- Are there degree courses
- Hi, Can I know about the degree
- Hi, Can I know about the degree programme
- Hi, I want to know about the degree
- Hi, I want to know about the degree programme
- Hi, Can you tell me about the degree
- Hi, Can you tell me about the degree programme
- Hi, Tell me about the degree
- Hi, Tell me about the degree programme
- Hi, Tell me details about the degree
- Hi, Tell me details about the degree programme
- Hi, Tell me what degrees are available there
- Hi, What are the available degrees there
- Hi, What are the degrees that you have
- Hi, I'm looking for a degree
- Hi, I want to do a degree
- Hi, I want to start a degree
- Hi, Do you have degrees
- Hi, Do you have computer degrees
- Hi, Do you have Business degrees
- Hi, Do you conduct degree courses
- Hi, Are there degree courses
- I want degree details
- Give me degree details
- I want to join the degree

intent:tell_field
- [Computer](field)
- I like [computer](field) field
- I want [computer](filed) degree
- [Computer](field) Degree
- It is [computer](field) field
- It's [computer](field) that I like
- I'm looking for [computer](field)
- [Computing](field)
- I like [computing](field) field
- I want [computing](filed) degree
- [Computing](field) Degree
- It is [computing](field) field
- It's [computing](field) that I like
- I'm looking for [computing](field)
- [Business](field)
- I like [business](field) field
- I want [business](filed) degree
- [Business](field) Degree
- It is [business](field) field
- It's [business](field) that I like
- I'm looking for [business](field)

intent:provide_contact_number
- My phone number is [0112454545](user_contact_number)
- My number [0775454545](user_contact_number)
- Call me [0765555555](user_contact_number)
- [0332255654](user_contact_number)
- [0332287542](user_contact_number)
- [0112454545](user_contact_number)
- Call me on [0348989856](user_contact_number)
- My contact number is [0776563214](user_contact_number)
- My contact number [0727441125](user_contact_number)
- Contact number [0112562356](user_contact_number)
- My mobile number [0252362578](user_contact_number)
- Mobile number [0325878787](user_contact_number)
- Mobile [0112748596](user_contact_number)

intent:tell_interested_degree
- [SE](degree_of_interest)
- [Software Engineering](degree_of_interest)
- I want to know about [software engineering](degree_of_interest)
- I like [SE](degree_of_interest)
- I like [software engineering](degree_of_interest)
- Tell me more about [SE](degree_of_interest)
- Tell me more about [Software Engineering](degree_of_interest) Degree
- [Business Information Systems](degree_of_interest)
- I want to know about [Business Information Systems](degree_of_interest)
- I like [BIS](degree_of_interest)
- I like [Business Information Systems](degree_of_interest)
- Tell me more about [BIS](degree_of_interest)
- Tell me more about [Business Information Systems](degree_of_interest)
- [BM](degree_of_interest)
- [Business Management](degree_of_interest)
- I want to know about [business management](degree_of_interest)
- I like [BM](degree_of_interest)
- I like [business management](degree_of_interest)
- Tell me more about [BM](degree_of_interest)
- Tell me more about [Business Management](degree_of_interest) Degree
- [CS](degree_of_interest)
- [Computer Science](degree_of_interest)
- I want to know about [computer science](degree_of_interest)
- I like [CS](degree_of_interest)
- I like [computer science](degree_of_interest)
- Tell me more about [CS](degree_of_interest)
- Tell me more about [Computer Science](degree_of_interest) Degree

intent:ask_prerequisites
- What are the prerequisites for the degree
- What qualifications should I have to follow the degree
- What qualifications are needed to follow the degree
- What should I have to follow degree
- Qualifications to start the degree
- What is the entry qualification for degree
- Do I need ALs to start degree
- Do I need ALs to registre for the degree

- I dont have ALs can I do the degree
- I have not completed ALs
- Do I have to complete A level
- What if I don't have ALs
- What are the qualifications I need
- What qualifications do I need

intent:ask_foundation
- What is [foundation](degree_of_interest)
- Tell me about [foundation](degree_of_interest)
- Can I do [foundation](degree_of_interest)
- Can you tell me about [foundation](degree_of_interest)
- I want to do [foundation](degree_of_interest)
- I like to do [foundation](degree_of_interest)
- Tell me more about the [foundation](degree_of_interest)
- Give me [foundation](degree_of_interest) details
- Can I know more about the [foundation](degree_of_interest)
- [Foundation](degree_of_interest) details
- I need [foundation](degree_of_interest) details

intent:resgister
- How to get register
- How can I get registered
- How to join the degree
- What is the registration process
- Can I apply
- How to get applications
- I want to join the degree

Appendix C

Contents of nlu_config.yml

language: "en"

pipeline: spacy_sklearn

Appendix D

domain.yml contain all the list of intents, entities, data types of entities, actions, templates which match the intents and the utterances or the actions.

```
intents:
 - greet
 - exit_greet
 - thank_you
 - tell_field
 - ask_degrees
 - ask_payments
 - location
 - prerequisite
 - facilities
 - resgister
 - ask_foundation
 - ask_prerequisites
 - tell_interested_degree
 - name:
    use_entities: false


entities:
 - degree_of_interest
 - user_name
 - user_contact_number
 - field

slots:
 degree_of_interest:
    type: text
 user_name:
    type: unfeaturized
 requested_slot:
    type: unfeaturized
 user_contact_number:
    type: unfeaturized
 field:
    type: unfeaturized


actions:
 - utter_greet
 - utter_exit_greet
 - utter_thank_you
 - utter_ask_degrees
 - utter_fields
 - utter_tell_interested_degree
 - action_provide_degree_details
 - action_provide_degree_names
 - name_form
```

```
 - utter_slots_values
 - utter_ask_payments
 - utter_location
 - utter_facilities
 - utter_prerequisite
 - utter_ask_foundation
 - utter_resgister

templates:
 utter_greet:
 - "Hello, Welcome to ABC! I'm Chatty. How can I help?"
 utter_exit_greet:
 - "Goodbye"
 - "Have nice day"
 - "Bye bye"
 utter_thank_you:
 - "You are welcome!"
 utter_ask_payments:
 - "Payments can be done in installments. Tell me your contact details. We'll call you back"
 utter_location:
 - "We are at, \nNo, 57, Ramakrishna Road, \nColombo 06. \nTele 0112 360 212"
 utter_facilities:
 - We have Air Conditioned Lecture rooms & Computer Labs, Library, Study areas, Canteen,
Swimming pool, Medical room, Play area.
 utter_fields:
 - text: "We have Business related Degrees And Computer related degrees. Which field do
you like?"
   buttons:
   - title: "Business"
     payload: /tell_field{"field":"business"}
   - title: "Computing"
     payload: /tell_field{"field":"computing"}

 utter_ask_degrees:
 - "ABC offers Software Engineering, Computer Science, And Business Information Systems
Degrees. Which do you like to know about?"
 utter_tell_interested_degree:
 - "Thank you for sharing your interested degree."
 utter_default:
 - "Sorry I did not understand what you said."
 - "I really did not get it. Can you rephrase it and ask again."
 - "I did not understand. Please ask in a simple way."
 utter_slots_values:
 - text: "Thank you for sharing contact details. We will call you back.\n
          - Name : {user_name}\n
          - Contact Number: {user_contact_number}"
 utter_submit:
 - "Saving your Name & Number ..."

 utter_prerequisite:
 - "You must have passes GCE A/L or equivalent to follow any of the degree. \n\nIf not you
can start from Foundation Course."

 utter_ask_foundation:
 - text: "If you haven't completed A/Ls or any other equivalent qualification, then you can
```

start from Foundation Course,"
  buttons:
  - title: "Foundation"
    payload: /tell_interested_degree{"degree_of_interest":"foundation"}

 utter_resgister:
 - "You will have to get in touch with our course counsellors. Tell me your name and contact number. We will contact you."

 forms:
  - name_form

Appendix E

```
# -*- coding: utf-8 -*-
from __future__ import absolute_import
from __future__ import division
from __future__ import print_function
from __future__ import unicode_literals

import logging
import requests
import json
from rasa_core_sdk import Action
from rasa_core_sdk.forms import FormAction

from rasa_core.events import SlotSet
import pymongo
import datetime
import winsound


logger = logging.getLogger(__name__)


def getUserContact(tracker):

        '''This function get the users name contact number and the
        course that the user want to follow. Store it in database
        along with date and time'''

        visitor_degree=tracker.get_slot('degree_of_interest')
        visitor_name=tracker.get_slot('user_name')
        visitor_contact_number=tracker.get_slot('user_contact_number')
        print(visitor_degree+" "+visitor_name+" "+visitor_contact_number)

        myClient=pymongo.MongoClient("mongodb://localhost:27017/")
        myDb=myClient["ABCInstitute"]
        myCollection=myDb["visitors"]

        visitor = { "name": visitor_name, "contact_number":
visitor_contact_number,"date_time":
        datetime.datetime.now(),"course":visitor_degree }
        x = myCollection.insert_one(visitor)


class ActionProvideDegreeDetails(Action):

        '''This is the class that define the action that should run for the
        intent tell_interested_degree.'''

        def name(self):
                return "action_provide_degree_details"
                winsound.Beep(500,1000)

        def run(self,dispatcher,tracker,domain):
```

```
                degree=tracker.get_slot('degree_of_interest')
                titleCasesDegree=degree.title()

                myClient=pymongo.MongoClient("mongodb://localhost:27017/")
                myDb=myClient["ABCInstitute"]
                myCollection=myDb["courses"]

                number=myCollection.find({"course_name":titleCasesDegree}).count()
                if number>0:
                        record=myCollection.find({"course_name":titleCasesDegree})
                        exact_name=record[0]['exact_name']
                        description=record[0]['description']
                        duration=record[0]['duration']
                        text=exact_name+"\n"+description+"\n It will take "+duration+" to
                        complete.\n\n Tell me your name & contact number. We will call you
                        back."
                else:
                        text=degree+" course is not Found in Database. \nPlease check
                        Spellings in "+titleCasesDegree

                dispatcher.utter_message(text)

                return []

class ActionProvideDegreeNames(Action):

        '"This is the class that define the action that should run for the
        intent tell_field.'"

        def name(self):

                return "action_provide_degree_names"

        def run(self,dispatcher,tracker,domain):

                buttons=[]

                myClient=pymongo.MongoClient("mongodb://localhost:27017/")
                myDb=myClient["ABCInstitute"]
                myCollection=myDb["courses"]

                record=myCollection.find({},{"_id":0,"course_name":1,"field":1})
                if tracker.get_slot('field')=="computer" or
                tracker.get_slot('field')=="Computer" or
                tracker.get_slot('field')=="Computing" or
                tracker.get_slot('field')=="computing":
                        field="computing"
                else:
                        field="business"

                for x in record:
                        if x['field']==field:

        buttons.append({"title":x['course_name'],"payload":"/tell_interested_degree{\"degree
```

```
            _of_interest\":\"" +x['course_name']+ "\"}"})

                    dispatcher.utter_button_message("ABC Offers following degree",buttons)
                    return []

class NameForm(FormAction):
  '"This class constantly listen for user name & contact number.
  This is triggered by name intent'"

  def name(self):
    # type: () -> Text
    """Unique identifier of the form"""
    return "name_form"

  @staticmethod
  def required_slots(tracker):
    # type: () -> List[Text]
    """A list of required slots that the form has to fill"""
    return ["user_name","user_contact_number"]

  def submit(self, dispatcher, tracker, domain):
    # type: (CollectingDispatcher, Tracker, Dict[Text, Any]) -> List[Dict]
    """Define what the form has to do
       after all required slots are filled"""
    getUserContact(tracker)
    dispatcher.utter_template('utter_submit', tracker)
    return []
```