Chhatrapati Shahu Ji Maharaj University

Project Report

on

[ChatBot]

Submitted in Partial Fulfillment of the Requirements for the Degree of

Bachelors of Computer Application

By:

[Samarth Swaroop Saxena]

[0202348]

Under the Supervision of

[FC.Pawan Sir]

PSIT College of Higher Education

Kanpur-Agra-Delhi National Highway – 2, Bhauti, Kanpur

(2020-2021)

FORMAT OF THE STUDENT PROJECT REPORT ON COMPLETION OF THE PROJECT

1. Cover Page as per format
2. Certificate
3. Declaration
4. Acknowledgement
5. Abstract
6. Index/List of Contents consists following Chapters:

- Chapter-I:
  - Introduction
  - Existing System
  - Problems in Existing System
  - Proposed System
  - Objective of the Project

- Chapter-II:
  - Modules & Description of the Modules
  - System Requirements
  - Technology used in project

- Chapter-III:
  - Feasibility Study
  - Software Development Life Cycle
  - Data Flow Diagram

- Chapter-IV:
  - Data Base Schema Design
  - Screen/Snap-Shots of the project

- Conclusion

- References

Declaration

I hereby declare that the project entitled "**ChatBot**" submitted for the Bachelor of Computer Application degree is my original work and the project has not formed the basis for the award of any other degree of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Signature of the student
Samarth Swaroop Saxena

(Samarth Swaroop Saxena)
(0202348)
BCA
PSIT College of Higher Education, Kanpur

CERTIFICATE

This is to certify that project entitled "**ChatBot** " submitted for partial fulfillment of the degree of BCA under the Department of Bachelor of Computer Application to through PSIT College of Higher Education, Kanpur, done by Mr./Ms **Samarth Swaroop Saxena** Roll No. **0202348** is an authentic work carried out by me under the guidance of **Fc.Pawan Sir**. The matter embodied in this project work has not been submitted earlier for award of any degree or diploma to the best of my knowledge and belief.

Internal Examiner/Guide External Examiner

Head of Department

ACKNOWLEDGEMENT

Presentation inspiration and motivation have always played a key role in the success of any venture.

I express my sincere thanks to **Dr. Shivani Kapoor, Director, PSIT College of Higher Education, Kanpur.**

I pay my deep sense of gratitude to **Asst.Prof.Santosh Kumar Sharma(HOD)** of BCA Department, **PSIT College of Higher Education** to encourage me to the highest peak and to provide me the opportunity to prepare the project. I am immensely obliged to **my friends** for their elevating inspiration, encouraging guidance and kind supervision in the completion of my project.

I feel to acknowledge my indebtedness and deep sense of gratitude to my guide **Mr. /Ms. Pawan Sir** whose valuable guidance and kind supervision given to me throughout the course which shaped the present work as its show.

Last, but not the least, **my parents** are also an important inspiration for me. So with due regards, I express my gratitude to them.

Abstract

People interact with systems more and more through voice assistants and chatbots. The days of solely engaging with a service through a keyboard are over. These new modes of user interaction are aided in part by This research will investigate how aadvancements in Artificial Intelligence and Machine Learning technology. are being used to improve many services. In particular it will look at the development of chatbot as a channel for information distribution. This project aimed to implement a web-based chatbot to assist with online banking, using tools that expose artificial intelligence methods such as natural language understanding. Allowing users to interact with the chatbot using natural language input and to train the chatbot using appropriate methods so it will be able to generate a response. The chatbot will allow users to view all their personal banking information all from within the chatbot. The produced prototype was found to be a very The TrueLayer API will be used to retrieve account information which has just recently been made accessible to the public. This is part of the Open Bank Standard which allows developer's access banking data through the API Rest client. This came about through the bank as a platform standard. useful tool to justify the need of a modern method of interaction to be integrated within many services offered by banks and financial businesses. In an industry with low user satisfaction rates and limited technology to increase accessibility. It is clear the chatbot overcomes the challenges banks face to increase the use of their services and gain a competitive edge over leading competitors. With many people adopting Smart Assistant Devices such as Google Home or Amazon's Alexa. The chatbot was tested across a range of devices such as Google Home and Assistant on android devices to outline the key differences between the two modes of interaction , spoken and text dialog. These test were carried out to identify the value in integrating such technology surrounding the recent interest in chatbots and conversational interfaces. Proving chatbots can be applied to a specific domain to enhance accessibility, reaffirming that they are more than just a passing fad and have a viable use.

1. Cover Page as per format
2. Certificate
3. Declaration
4. Acknowledgement
5. Abstract
6. Index/List of Contents consists following Chapters:

- Chapter-I:
    - Introduction
    - Existing System
    - Problems in Existing System
    - Proposed System
    - Objective of the Project

- Chapter-II:
    - Modules & Description of the Modules
    - System Requirements
    - Technology used in project

- Chapter-III:
    - Feasibility Study
    - Software Development Life Cycle
    - Data Flow Diagram

- Chapter-IV:
    - Data Base Schema Design
    - Screen/Snap-Shots of the project

- Conclusion

- References

CHAPTER-1

Introduction

“Digitalisation, the surge of mobile and internet connected devices has revolutionised the way people interact with one another and communicate with businesses” (Eeuwen, M.V. (2017)). Millennials are accepting and supporting new technology into the routine of their everyday life, this is becoming more is becoming more prevalent as technology companies are streamlining Artificial IntelliIntelligence (AI) into the products they offer, such as; Google Assistant, Google Home and Amazon Alexa. The new and upcoming generation are expected to be critical and game changing customers for businesses. “They demand effortless experiences, answers within seconds, not minutes and more intelligent self-service options” (Teller Vision,, . (2017)). The banking and the financial service industry was one of the first industries to adopt technology. This integration has grown massively, helping banks reach a wider customer base enabling them to perform their banking conveniently (Baptista and , G. and Oliveira, , T. (2015)). Banks are becoming ever more competitive with each other to adopt the newest advancements in technology to provide an improved delivery service to satisfy customers. Ulster Bank, Deloitte, AIB and PTSB are wanting to focus on integrating new technology to improve the speed at which transactions are acknowledged (Global Banking News, . (, 2017)). With this in mind the relationship with the customer is always evolving due to the growth of technology. Banks are now enabling the use of technology so customers can perform more tasks online, such as; cheque image clearing to allow the payment of cheques remotely and intelligent chatbots to increase customer service and assist employees. A chatbot is a “simple software program that can respond to customer prompts i.e. what’s my bank balance?” (Entrepreneur, . (, 2016)). Mastercard has launched Kai an artificial intelligent chatbot and other bots for financial services. They can handle customer queries such as: ‘what is APR?’, requests, look at spending habits and solve problems. This in turn enables financial institutions to provide a new, engaging experience and strengthen their relationship with the customer, with the aid of natural language used by bots to establish a more personal and contextual conversation (Wire, . (, 2016)). The focus of this project is to implement these new technologies to create an intelligent chatbot to enable banks to appeal to millennials and potentially gain a lifelong customer.

Existing System

The people are working timely don’t have time to do their basic works like playing music view mails while driving basically they are not multitasker. We have to on their device and type and scroll things and give time to the devices.

Problems in Existing System

It is evident from the research carried out in the literature review that modern basic services are constantly seeking to expand their technologies, both to improve customer service and increase delivery of services through the advancements in technology. This is to gain a competitive edge over other banks for financial benefits and to expand its customer base. A domain specific chatbot will be implemented to assist users with their banking. In order to overcome the user satisfaction issues associated with online banking services. The chatbot will provide personal and efficient communication between the user and their bank in order to manage their finances and get assistance when needed, such as; answering any queries and booking appointments. The chatbot will allow users to feel confident and comfortable when using this service regardless of the user's computer literacy due to the natural language used in messages. It also provides a very accessible and efficient service as all interactions will take place within the one chat conversation negating the need for the user to navigate through a site.

Proposed System

The proposed solution is to create a chatbot to simulate a human conversation to assist users with their banking needs and to provide a more personal experience. Advancements in artificial Intelligence, machine learning techniques, improved aptitude for decision making, larger availability of domains and corpus, have increased the practicality of integrating a chat bot into applications (Dole et al., 2015). Users will be able to ask any banking related queries in natural language that they are comfortable using such as; view account information, transactions and check balance. The chatbot will identify and understand what the user is asking and generate an appropriate response based on the conversational context. Immediate responses will be provided by the chatbot to redeem the need for the user to have to call or visit their local banks branch and wait in queue in order to get through to an advisor for assistance. In order to make the application more secure Googles 2 Factor Authentication will be integrated to increase security ensuring only registered users can gain access to their account preventing the risk of fraud.

Objective of the Project

This work aims to provide a fast and convenient way to manage your basic work. The online voice customer support chatbot will help facilitate the user with queries and assist with personal work. The application will allow users to c: Check their routien, View date and time View your mails. There will be integration of SMS, two factor authentication and the ability to interact with the service through a chatbot. • SMS confirmations and email mini statement schedule. The Starling API will be used to access the users banking information requested through the chatbot. • tele conversions • Ask queries through chatbot and get appropriate and immediate responses • Improve customer service through conversation using the chatbot The chatbot will be implemented using the Laravel Framework and Dialogflow. The Dialogflow will be utilised as the NLU to perform artificial intelligence methods such as; Natural Language Processing (NLP), POS tagging, and entity recognition to analyse text and carry out the appropriate actions. Dialogflow is a Natural Language Understanding Engine (NLU) used for the extraction of entities and intent from a user's message. Google Two-Factor Authentication will be implemented as an extra measurement of security for the customer. This ensures that no one else can access their account and view their personal banking information. A unique code will be generated and sent to their phone from scanning a QR code displayed after a successful login. Users will download the Google Authenticator app onto their mobile device to receive the unique code generated which will grant access to the chatbot.

CHAPTER-II

Modules & Description of the Modules

As shown in the figure, there are six components in our project. The interfaces are the front end chat box for user to talk to the bot, which can be the Bot Portal, Skype, Facebook, etc. The connectorworks as a common gateway for all the interfaces. The outbound side calls different APIs to different front end, but the inbound APIs kept the same for our bot to connect. Fortunately, this connector has already been implemented by the bot framework SDK, we only need to rightly configure them. The botpart contains the main flow control of our project. It is responsible for redirect the input to different models, parse the return values, and determines what to do next. It is also connected to the database to retrieve and update values.

```
from urllib.request import ProxyDigestAuthHandler
import pyttsx3
import speech_recognition as sr

import wikipedia
import webbrowser
import os
import datetime
import smtplib
from pyttsx3 import engine
engine = pyttsx3.init('sapi5')
voices = engine.getProperty('voices')
engine.setProperty('voice', voices[1].id)
```

System Requirements

Hardware Requirements:

Processor : intel 3(minimum)

Hard Disk : 256MB (minimum)

RAM : 4GB(minimum)

Software Requirements:

Operating System : Windows

Technology : PYTHON

IDE: Visual Studio Code

Technology used in project

ABOUT PYTHON:
Dating from 1991, Python is a relatively new programming language. From the start, Python was considered a gap-filler, a way to write scripts that "automate the boring stuff" (as one popular book on learning Python put it) or to rapidly prototype applications that will be implemented in one or more other languages. However, over the past few years, Python has emerged as a first-class citizen in modern software development, infrastructure management, and data analysis. It is no longer a backroom utility language, but a major force in web application development and systems management and a key driver behind the explosion in big data analytics and machine Perfect for IT, Python simplifies many kinds of work, from system automation to working in cutting-edge fields like machine learning. Python is easy to learn. The number of features in the language itself is modest, requiring relatively little investment of time or effort to produce one's first programs. Python syntax is designed to be readable and straightforward. This simplicity makes Python an ideal teaching language, and allows newcomers to pick it up quickly. Developers spend more time thinking about the problem they're trying to solve, and less time thinking about language complexities or deciphering code left by others. Python is broadly used and supported. Python is both popular and widely used, as the high rankings in surveys like the Tiobe Index and the large number of GitHub projects using Python attest. Python runs on every major operating system and platform, and most minor ones too. Many major libraries and API-powered services have Python bindings or wrappers, allowing Python to interface freely with those services or make direct use of those libraries. Python may not be the fastest language, but what it lacks in speed, it makes up for in versatility. Python is not a "toy" language. Even though scripting and automation cover a large chunk of Python's use cases (more on that below), Python is also used to build robust, professionalquality software, both as standalone applications and as web services.

. What is Python used for?

The most basic use case for Python is as a scripting and automation language. Python isn't just a replacement for shell scripts or batch files, but is also used to automate interactions with web browsers or application GUIs or system provisioning and configuration in tools such as Ansible and Salt. But scripting and automation represent only the tip of the iceberg with Python.

- Python is used for general application programming. Both CLI and cross-platform GUI applications can be created with Python and deployed as self-contained executables. Python doesn't have the native ability to generate a standalone binary from a script, but third-party packages like cx_Freeze or PyInstaller can be used to accomplish that.
- Python is used for data science and machine learning. Sophisticated data analysis has become one of fastest moving areas of IT and one of Python's star use cases. The vast majority of the libraries used for data science or machine learning have Python interfaces, making the language the most popular high-level command interface to for machine learning libraries and other numerical algorithms.
- Python is used for web services and RESTful APIs. Python's native libraries and third-party web frameworks provide fast and convenient ways to create everything from simple REST APIs in a few lines of code, to full-blown, data-driven sites. Python's latest versions have powerful support for asynchronous operations, allowing sites to handle up to tens of thousands of requests per second with the right libraries.
- Python is used for metaprogramming. In Python, everything in the language is an object, including Python modules and libraries themselves. This allows Python to work as a highly efficient code generator, making it possible to write applications that manipulate their own functions and have the kind of extensibility that would be difficult or impossible to pull off in other languages.
- Python is used for glue code. Python is often described as a "glue language," meaning it can allow disparate code (typically libraries with C language interfaces) to interoperate. Its use in data science and machine learning is in this vein, but that's just one incarnation of the general idea. Also worth noting are the sorts of tasks Python is not well-suited for. Python is a high-level language, so it's not suitable for system-level programming—device drivers or OS kernels are straight out. It's also not ideal for situations that call for cross-platform standalone binaries. You could build a standalone Python app for Windows, Mac, and Linux, but not elegantly or simply. Finally, Python is not the best choice when speed is an absolute priority in every aspect of the application. For that you're better off with C/C++ or another language of that caliber.

The Python language's pros and cons

Python syntax is meant to be readable and clean, with little pretense. A standard "hello world" in Python 3.x is nothing more than:

- print("Hello world!")
- Python provides many syntactical elements that make it possible to concisely express many common program flows. Consider a sample program for reading lines from a text file into a list object, stripping each line of its terminating newline character along the way:
- with open('myfile.txt') as my_file:
- file_lines = [x.strip('\n') for x in my_file]
- The with/as construction is a "context manager," which provides an efficient way to instantiate a given object for a block of code and then dispose of it outside of that block. In this case, the object in question is my_file, instantiated with the open() function. This takes the place of several lines of boilerplate to open the file, read individual lines from it, then close it up.
- The [x … for x in my_file] construction is another Python idiosyncrasy, the "list comprehension." It allows a given item that contains other items (here, my_file and the lines it contains) to be iterated through, and to allow each iterated element (that is, each x) to be processed and automatically appended into a list.
- You could write such a thing as a formal for… loop in Python, much as you would in another language. The point is that Python has a way to economically express things like loops that iterate over multiple objects and perform some simple operation on each element in the loop, or work with things that require explicit instantiation and disposal. Constructions like this allow Python developers to balance terseness and readability.
- Python's other language features are meant to complement common use cases. Most modern object types—Unicode strings, for instance—are built directly into the language. Data structures—like lists, dictionaries (i.e., hashmaps), tuples (for storing immutable collections of objects), and sets (for storing collections of unique objects)—are available as standard-issue items.
- Like C#, Java, and Go, Python has garbage-collected memory management, meaning the programmer doesn't have to implement code to track and release objects. Normally garbage collection happens automatically in the background, but if that poses a performance problem, it can be triggered manually or disabled entirely.
- An important aspect of Python is its dynamism. Everything in the language, including functions and modules themselves, are handled as objects. This comes at the expense of speed (more on that below), but makes it far easier to write high-level code. Developers can perform complex object manipulations with only a few instructions, and even treat parts of an application as abstractions that can be altered if needed.
- Python's use of significant whitespace has been cited as both one of Python's best and worst attributes. The indentation on the second line shown above isn't just for readability; it is part of Python's syntax. Python interpreters will reject programs that don't use proper indentation to indicate control flow
- Syntactical white space might cause noses to wrinkle, and some people do reject Python out of hand for this reason. But strict indentation rules are far less obtrusive in practice than they might seem in theory, even with the most minimal of code editors, and the end result is code that is cleaner and more readable. Python 2 versus Python 3

- Python is available in two versions, which are different enough to trip up many new users. Python 2.x, the older “legacy” branch, will continue to be supported (i.e. receive official updates) through 2020, and it might even persist unofficially after that. Python 3.x, the current and future incarnation of the language, has many useful and important features not found in 2.x, such as better concurrency controls and a more efficient interpreter.
- Python 3 adoption was slowed for the longest time by the relative lack of third-party library support. Many Python libraries supported only Python 2, making it difficult to switch. But over the last couple of years, the number of libraries supporting only Python 2 has dwindled; most are now compatible with both versions. Today, there are few reasons against using Python 3. 29 |

Chapter-III

Software Development Life Cycle

Deciding upon an appropriate methodology is vital for the overall development of any software application to ensure a realistic timeframe is established for each stage of the project and requirements are clearly outlined. Various development methodologies will be discussed and considered for the development and design of this software. This section will highlight the development methodology that is best suited to this project.

Waterfall Methodology

This is a very traditionally methodology, which is usually introduced when you initially learn about software development. The waterfall model is a very predictive approach to software development that consists of 5 stages to include; requirements gathering, analysis, design, implementation and testing. Each stage is completed subsequently of one another. A major drawback of the waterfall model is that it is very inflexible, as the project is broken up into phases. Each phase is given a deadline in order for a deliverable to be produced at the end of each phase to adhere to the overall project schedule. The success and progression of the project is measured from the project deliverables, design documents and test plans. As each phase of the project is outlined at the beginning of the project lifecycle and targets have been set it's difficult to integrate new requirements or a change in requirements that may be identified at a later stage as it would adversely affect the overall project schedule. The waterfall model moves a lot of the more high risk and difficult components towards the end of the project life cycle.

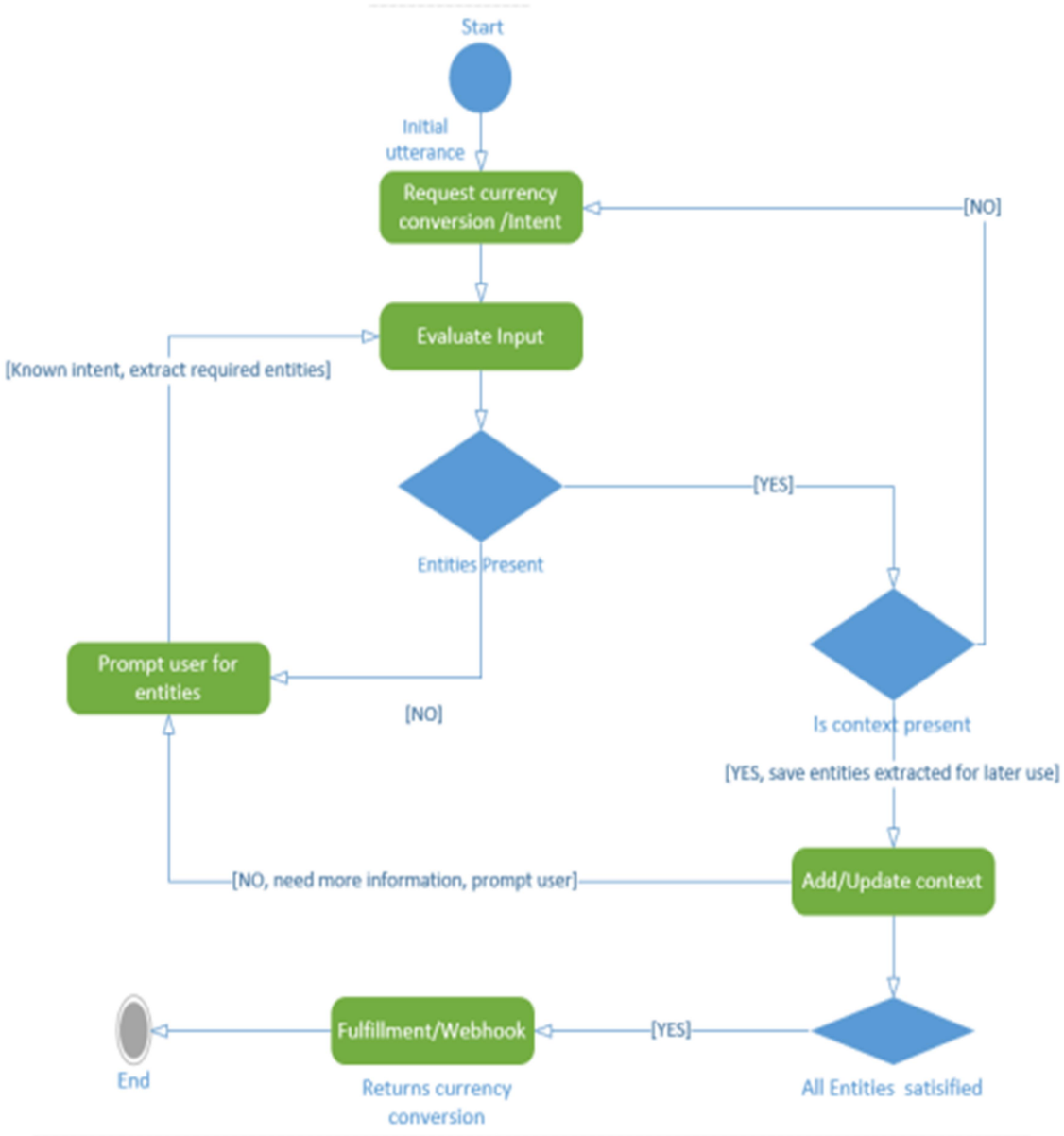
Incremental Model

This software methodology evolved from the waterfall model. The application is designed, developed and tested using iterative incremental build stages. At the end of each build a subsystem or feature will be created. The project will progress in complexity as new requirements are likely to be discovered and implemented in each incremental build, developing on top of the functionality from the last build leading to the overall development of the application. It is very common for software to be released in stages, it is critical that component versions utilised within the software are managed throughout the entire lifecycle using version control tools such as GitHub. Each build will only last a few weeks to produce a baseline version of the application. Feedback can be given on any requirement errors or faults found in the application. Distributing the development of the project over various build cycles can lower the risks associated with development to a more manageable level as requirements are broken down into smaller functionality to be implemented at the end of each build.

Chosen Methodology

The incremental model is the most suitable development methodology to implement for this project. The flexibility of the incremental model makes it ideal for this project as it is likely new requirements will be identified during the later stages of development and each iterative build makes it easy to implement new requirements throughout the the development process.

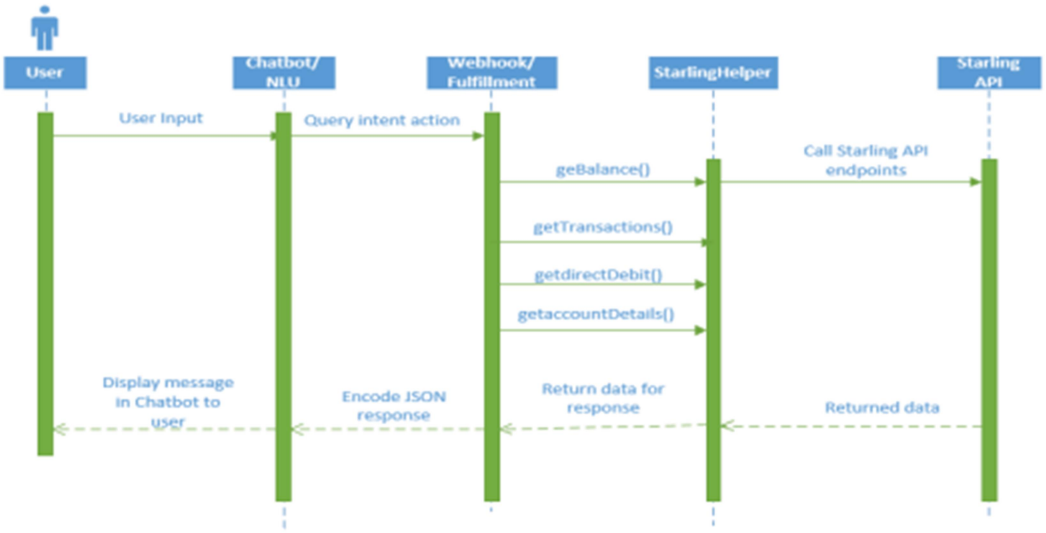
Data Flow Diagram



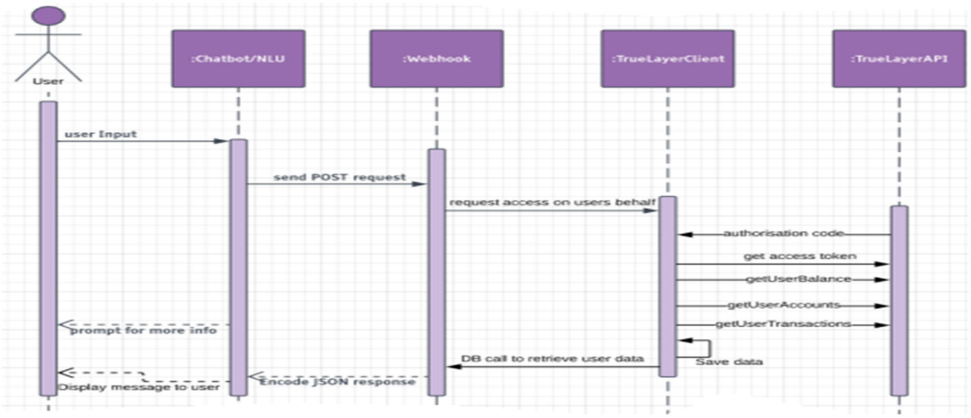
Activity diagrams are drawn to specify the informational flow of the data and outline the processes that are involved in the chatbot. They represent the users' behaviour when interacting with the chatbot and how it responds to these actions through processes. These will breakdown the distinct functionality of the chatbot. Above Figure describes the chatbot logic when a user requests currency conversion. Once the NLU evaluates the user input and determines a context, the user utterance is passed to the Fulfilment webhook. The webhook utilises POST endpoints and accepts JSON returned from the chatbot NLU. The webhook then determines the associated action of an intent the user supplied utterance is mapped to; the action of an intent was defined in the Dialogflow console during training. If the action is present it will extract the entities from the given utterance and call the Fixer.io API and return the appropriate response encoded in JSON format. If the NLU cannot recognise the intent of the user supplied utterance it will prompt the user for more information to try and recognise the intent and context to provide fulfilment for the user.

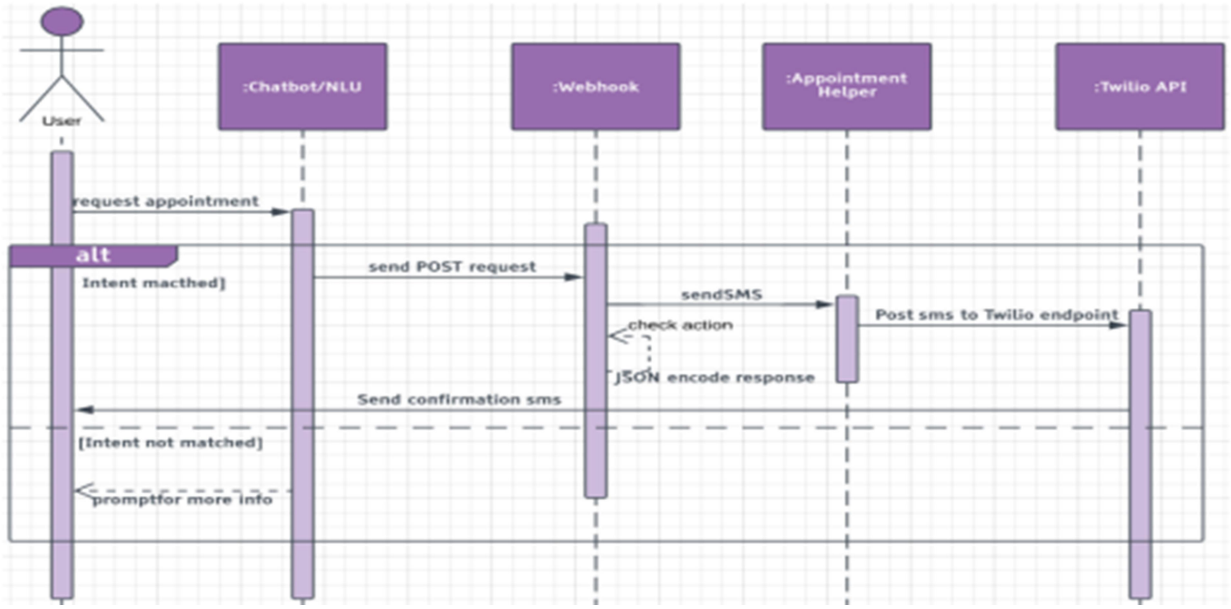
Sequence Diagrams

The sequence diagram shown in Figure 13 3.6.1 highlights the operation of the user requesting their banking information through the chatbot, outlining the fundamental process involved. The webhook receives the user input from the Chatbot NLU engine (Dialogflow) in the form of JSON. The webhook determines the action of the defined intent posted from the NLU by parsing the JSON it receives. If the action matches a defined banking operation, the webhook will make the appropriate method method call to the TrueLayerClientStarlingHelper Class, which calls the TrueLayer Starling API and returns the data to the webhook to generate a for



the response. Once the webhook receives the data returned by the API it encodes the data to JSON which is as the response sent back to the NLU to be displayed to the user through the chatbot view.



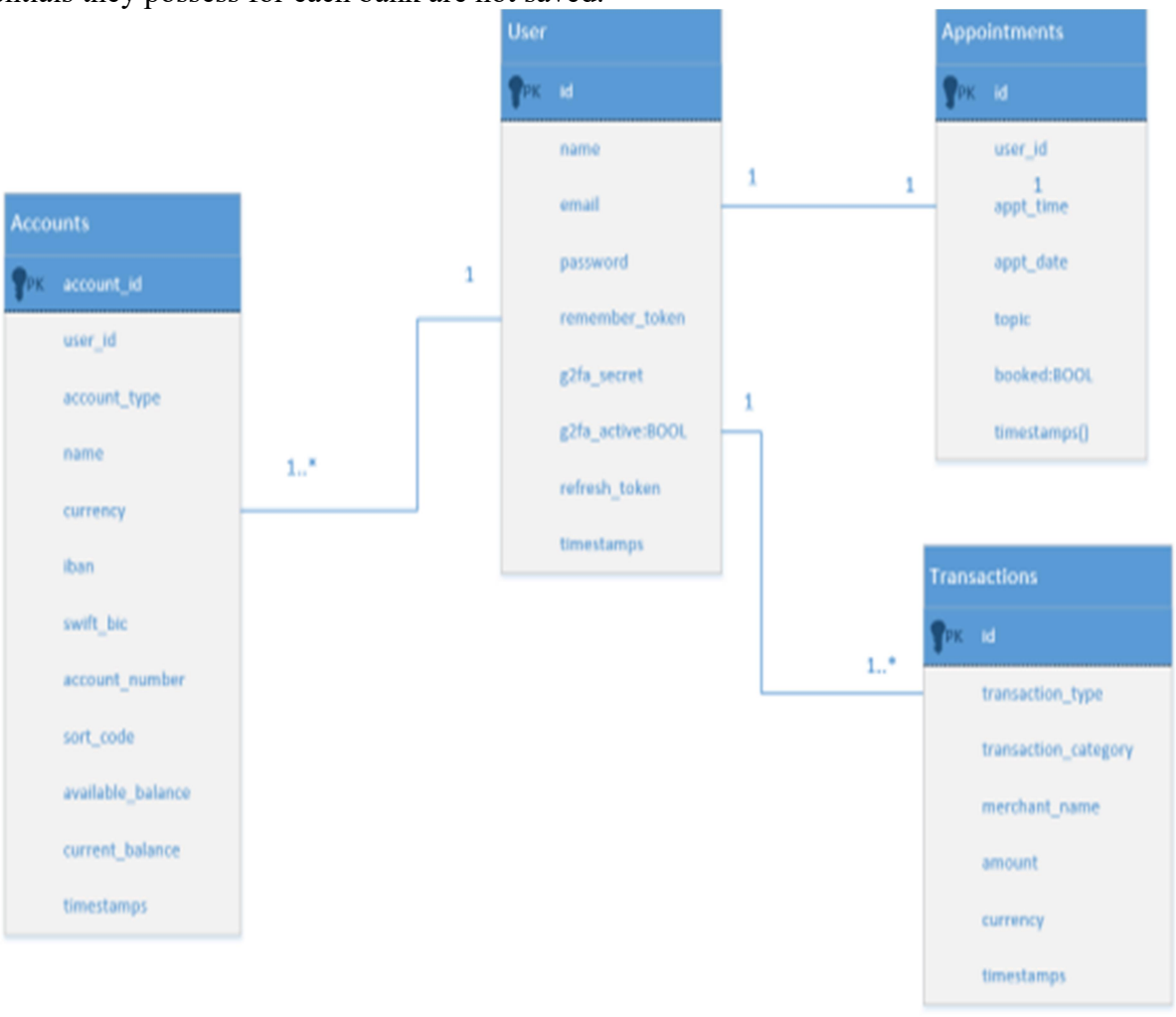


In order to implement a successful chatbot the user group that will be interacting with it is carefully thought through. The target audience for the chatbot consists of different age groups with varying technical ability, as mobile banking is relevant across a broad range of age groups. To appeal to all age groups the chatbot will utilise simplistic and straightforward responses. Google assistant allows developers to determine a voice for their bot that matches the personality and utilise rich responses (Actions on Google, 2018). However the majority of users of an application like this already carry out the most part of their online interaction through some messaging platform, so the use of such an application should come with ease (Interactions.acm.org, 2017).

Chapter-IV

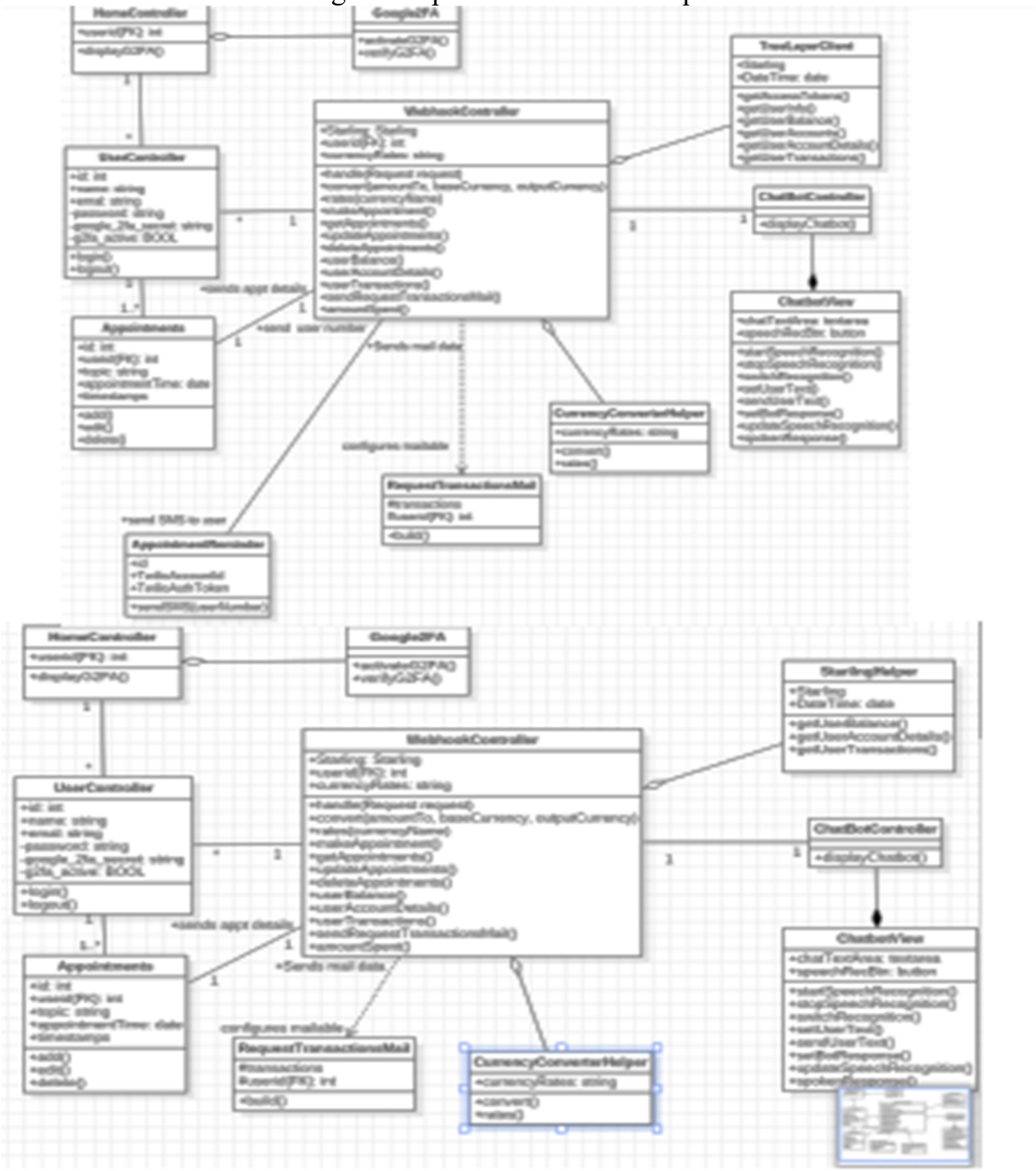
Data Base Schema Design

Laravel also comes with Eloquent ORM, which "provides an ActiveRecord implementation for working with database objects" (Laravel, 2018). The data is stored locally using a MySQL Workbench database, it is small-scale to conserve memory. Every table within the database is represented using a 'model' which allows logic and relationships between database objects to be defined and manipulated. The database model for the proposed chatbot is outlined in Figure 83.2.1 with defined multiplicity. The users table contains data about the users of the application and whether or not they are verified through google 2 Factor Authentication. The appointments table holds data about the user's appointments such as 'topic'. This holds information specified by the user about what the appointment is for. The 'booked' attribute is set to a Boolean value to determine whether or not the user already has an appointment. The accounts and transactions table holds data regarding the usersuser's bank accounts, for instance they may have multiple account types with the one bank such as current and savings, however the users online banking credentials they possess for each bank are not saved.



Class Diagram

The class diagram shown in Figure 9 3.3.1 illustrates the classes required to implement the chatbot. This will act as a guide for implementing the chatbot however other classes may be identified or modified during development to meet the requirements.



Speech Recognition

Users are able to interact with the chatbot using voice commandsspeak to the chatbot through the built inbuilt-in microphone on the chatbot device. The HTML5 speech Recognition API is implemented within the web app for speech recognition and synthesis.

Dialogflow will then take the deciphered speech and convert it into a structured JSON object it can analyse, this is used as the text input. The JSON response is given an entity matching score also known as confidence score. This score expresses how well the NLU engine matched the user input to an intent defined on the console (DialogFlow, 2015). Scores range between 0-1, 1 being an exact match. Below is a snippet of the JSON object response. User says: “Who are you?


```
{       "fulfilment": {
            "speech": "I am your smart banking bot, use me to get your account balance quickly, view transactions, debits, account
        details and receive currency conversions.",
            "messages": [
              {
                "type": 0,
                "speech": "I am your smart banking bot, use me to get your account balance quickly, view transactions, debits,
        account details and receive currency conversions."
} ]
},  "score": 1 - Exact Match
```


.

Conclusion

Chatbots are the new Apps! As we have discussed in the above deliverables, this project brings the power of chatbots to Yioop and enriches its usability. Chatbots in Yioop can give a human like touch to some aspects and make it an enjoying conversation. And they are focused entirely on providing information and completing tasks for the humans they interact with. The above mentioned functionality in all the deliverables is implemented and pushed in to Yioop code. By implementing the above mentioned deliverables I was able to add a basic chatbot functionality in to the Yioop. I.e., configuring and creating accounts for bot users with bot settings which is mentioned in deliverable 2, activating a bot whenever a user asks for it via post in a thread which is discussed in deliverable 3 and as I discussed in deliverable 4, I have implemented a simple weather chatbot that gives weather information whenever a user ask and Fig. 3 tells that I was also able to converse with the bot in Yioop. I intend to enhance the system developed so far in CS298. Next step towards building chatbots involve helping people to facilitate their work and interact with computers using natural language or using set of rules. Future Yioop chatbots, backed by machine-learning technology, will be able to remember past conversations and learn from them to answer new ones. The challenge would be conversing with multiple bot users and multiple user.

Reference

1. Bayan Abu Shawar and Eric Atwell, 2007 “Chatbots: Are they Really Useful?”
2. LDV Forum - GLDV Journal for Computational Linguistics and Language Technology.
3. http://www.ldv-forum.org/2007_Heft1/Bayan_Abu-Shawar_and_Eric_Atwell.pdf
4. Bringing chatbots into education: Towards natural language negotiation of open learner models. Know.-Based Syst. 20, 2 (Mar. 2007), 177-185.
5. Intelligent Tutoring Systems: Prospects for Guided Practice and Efficient Learning. Whitepaper for the Army's Science of Learning Workshop, Hampton, VA. Aug 1-3, 2006.
6. http://en.wikipedia.org/wiki/Chatterbot
7. ALICE. 2002. A.L.I.C.E AI Foundation, http://www.alicebot.org/