AUTOMATED DOCTOR APPOINTMENT MANAGEMENT SYSTEM FOR MEDICAL DOMAIN IN SINHALA USER(ADAMS).

2020-175

Project Proposal Report

Nishshanka N.A.B.D

B.Sc. (Hons) Degree in Information Technology

Department of Information Technology

Sri Lanka Institute of Information Technology
Sri Lanka

March 2020

RASA Framework

2020-175

Project Proposal Report

B.Sc. (Hons) Degree in Information Technology

Department of Information Technology

Sri Lanka Institute of Information Technology
Sri Lanka

March 2020

DECLARATION

We declare that this is our own work and this proposal does not incorporate without acknowledgement of any material previously submitted for a degree or diploma in any other university or Institute of higher learning and to the best of our knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Name	**IT Number**	**Signature**	Date
Nishshanka N.A.B.D	IT17043656		25/02/2020

Supervisor

---------- ----------

Prof. Koliya Pulasinghe Date

Co-Supervisor

---------- ----------

Ms. Vijani Piyawardana Date

ABSTRACT

Medical treatment is an important thing in the world. Every patient needs some medical treatment for their diseases. Accessibility to medical knowledge and without proper doctor appointment method are the two major impediments for human. For that reason, want some medical assistance to manage this work. **A**utomated **D**octor **A**ppointment **M**anagement **S**ystem (**ADAMS**) is a proper solution for those problems. ADAMS is a fully automated system. This system develops for the Sinhala users. User didn't want to type anything. They can give a voice command and do their medical assistance needs. It will be the first disease identification and appointment management mobile application using Sinhala voice command in Sri Lanka.

This function describes the implementation of RASA framework for Sinhala language using json queries. This implemented as a sub objective of ADAMS. Purpose about this RASA framework is managing Sinhala dialogue conversation about between user and system. It gives the relevant output what user asked by using the RASA Core.

Keywords -: RASA, RASA NLU, RASA Core, Json

LIST OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

This system will be developed with the help of RASA framework. RASA is an open source framework to develop these kinds of the systems. It offers a set of open source learning tools to create contextual system help with voice. There are two major parts in RASA framework. First one is RASA NLU. RASA NLU is an open-source natural language processing tool. Purpose of this NLU is intent classification and entity extract. If the user given some voice command to the ADAMS. It can identify user input based on intent classification and extract entities. Second part is RASA core. Purpose of this RASA core is deciding what happens next inside this conversation and mange patient medical questions and response answers for this system.

1.1 BACKGROUND & LITERATURE SURVEY

Now a day's medical domain systems are developed for the RASA framework. Quinn and Diabot are the examples for the systems which are developed using RASA Framework. Most of the RASA projects are develop based on English language.

Purpose of ADAMS sub objective is connecting a Sinhala response text into to RASA NLU. But in Quinn system some of the parts are already done using English language [2]. Purpose of this bot is interacting with the user to analyze their thoughts or state of mind and suggests appropriate solutions. Below the steps are done inside the Quinn chatbot.

The implementation of Quinn involves the creation of two models which are the NLU model, as well as, the Dialogue management model. A bot cannot understand the unstructured human language. Hence it is necessary to convert the high level and unorganized human language to a structured form that is understandable and distinguishable by the bot. This can be achieved by training an NLU model that takes training data with all the data concerned with a particular intent listed under that intent[2].

A large number of data can be added under each intent to increase the efficiency of the intent classification[2]. One can also specify the information that needs to be extracted and the name of the slot to which the extracted information is stored. Synonyms, as well as, regex patterns can also be specified in the data file. Thus, the NLU data file consists of user messages and its intents with entities to be extracted. After the NLU model has been trained, a test sentence can be provided to check if it performs the intent classification accurately[2].

A large number of data can be added under each intent to increase the efficiency of the intent classification. One can also specify the information that needs to be extracted and the name of the slot to which the extracted information is stored. Synonyms, as well as, regex patterns can also be specified in the data file. Thus, the NLU data file consists of user messages and its intents with entities to be extracted. After the NLU

model has been trained, a test sentence can be provided to check if it performs the intent classification accurately[2].

To train the dialogue management model, an Agent class that takes a domain file and policies as parameters are used. Memoization Policy, Keras Policy, Form Policy and Fallback Policy are the policies used. Memoization policy is used to remember the events as per story and Keras Policy is used to predict the next action if the user asks something different from stories. Fallback Policy checks whether the Rasa Stack component Rasa NLU responsible for understanding user messages was able to classify the user message confidently. If the classification confidence is below a certain threshold, it triggers a fallback action .Form Policy is specified to implement slot filling in an effectively. It extracts information from the user before any utterance or action is executed. The model is thus trained with the help of agent with all the sufficient parameter and it gets persisted in the specified model path[2].

Custom actions are defined in python. Quinn extracts the name of the user and greets by his/her name. It sends mail to the doctor when the user expresses any suicidal thoughts. It also assists in booking an appointment with the doctor. All these are custom actions are made available by running the custom action server[2].

Now, the rasa core server is run. For this, an agent class is required which takes the dialogue management model as domain, NLU model as an interpreter and an action endpoint. Action endpoint is url where the trained model is loaded and ready to work as a chatbot. An input channel is created with a port number that can be connected to the designed user interface. The bot can in parallel be integrated with Google Assistant just by adding another input channel with another port number. But since the Google Assistant is an internet service and the bot resides in the localhost, it requires ‘ngrok’ that acts as a tunnel between the bot and the Google Assistant. ngrok is run at the port number specified in the input channel which was defined earlier. This way one can connect the trained bot to various platforms like Slack, Messenger, Twilio etc[2].

Second research paper is Diabot [1]. And it also done for the medical domain. Purpose of this Diabot is developed a generic text-to-text ‘Diabot’ – a DIAgnostic chatBOT which engages patients in conversation using advanced Natural Language Understanding (NLU) techniques to provide personalized prediction using the general health dataset and based on the various symptoms sought from the patient[1].

This is the System module in Diabot chatbot,

At a high level, the system consists of a front-end User Interface (UI) for the patient to chat with the bot. It is developed using React UI platform which uses HTML and JavaScript. The chatbot communicates with the NLU engine at the backend via API calls. The NLU engine is developed using RASA NLU platform. Two models- one for generic health prediction and another for advanced diabetes prediction are trained at the backend using the general health dataset and the Pima Indian diabetes dataset which provide the necessary diagnosis decision to the NLU engine which is trained to

provide output based on user queries. A brief schematic of the system is shown in Fig 1[1].

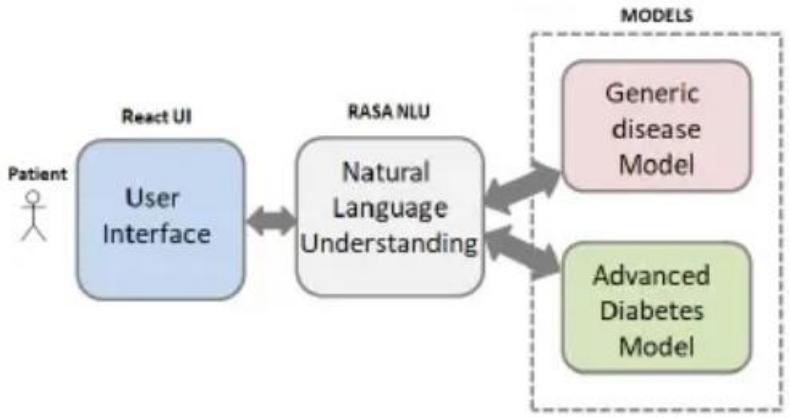

Figure 1. Brief schematic of the system

The detailed architecture of the advanced diabetes model – its training, testing and ensemble learning is shown in Fig 2. The same applies to the generic disease model also albeit different classifiers[1].

Figure 2. Procedure of training, testing & Ensemble learning for Diabetes dataset

The detailed architecture of the Chatbot is shown in Fig 3. It uses a client-server technology. The client is developed using the semantic UI React library, which is an open-source JavaScript library. It uses HTML for development framework. It communicates with a trained NLU engine at the backend. The NLU engine is

connected to both the generic health and the advanced diabetes models and their outputs (Model data) are saved in 'Pickle' file format. On receiving an API call, the model data in the pickle file format is loaded and made available to the UI via the Flask webserver[1].

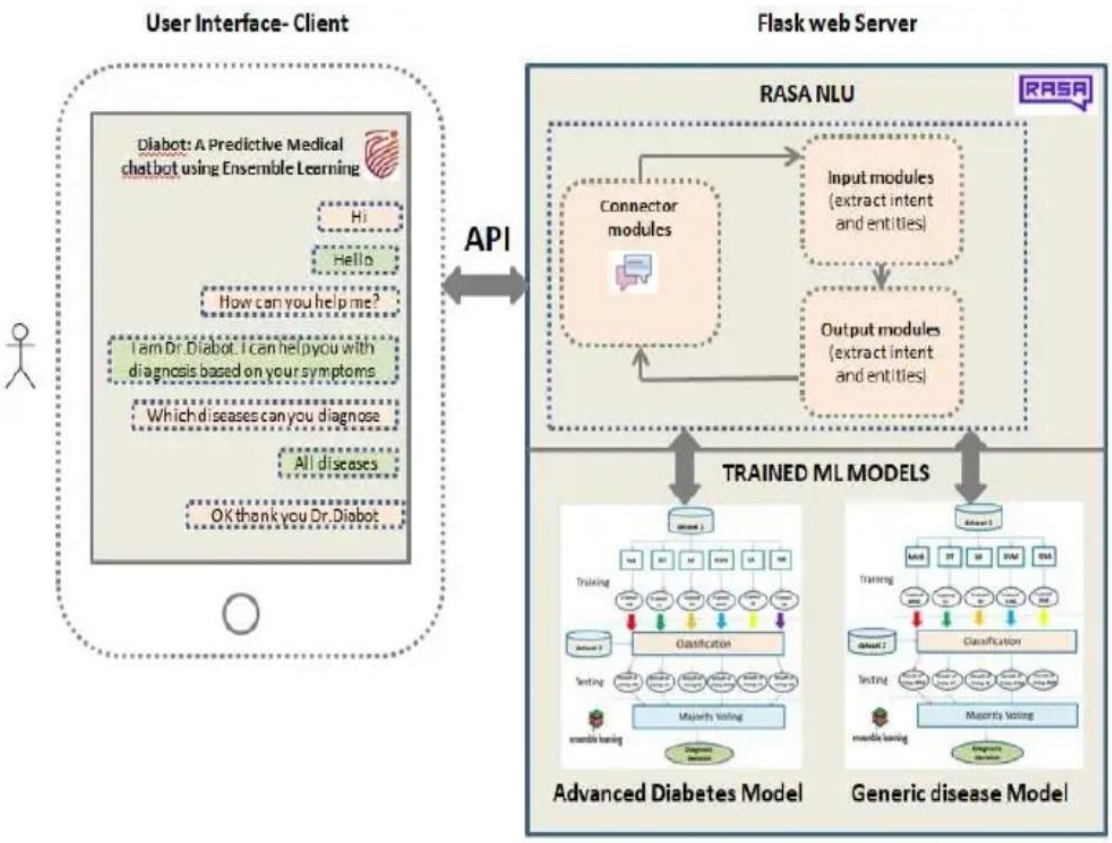

Figure 3.Detailed Chatbot architecture

1.2 RESEARCH GAP

Using RASA frame work already done some of features which are mention below. Those features implemented in a chatbot application based on English language. Through ADAMS done all the developments under mobile application using RASA framework in medical domain.

None of these applications didn't able to develop a functionality to convert Sinhala text in to json format. Also, that json query didn't generating automatically.

Table 1.Research gap between existing system and ADAMS

Features	Quinn Chatbot [2]	Diabot chatbot [1]	La Liga Chatbot [3]	RASA Framework on Flutter mobile application
Convert Sinhala Dialogue into the Json format	✗	✗	✗	✓
Automatically generate a json query	✗	✗	✗	✓
Identify entities and intents	✓	✗	✗	✓
Json format move to the RASA NLU	✗	✗	✗	✓
Extract the Json query inside a RASA NLU	✗	✗	✗	✓
RASA NLU move to the RASA core	✓	✗	✗	✓
Extract the RASA NLU inside a RASA core	✓	✗	✗	✓
Handel the conversation using RASA core	✓	✗	✗	✓

2. RESEARCH PROBLEM

Nowadays in Sri Lanka, all of the e-channelling systems are based on the English language. Mostly e-channelling systems are used by people in western province comparing to other provinces. The reason is the erudition that they have is very impecunious in the English language.

Sometimes, people having diseases such as Short-Term and Long-Term Incapacitation and People with dyslexia, are incapable of using communicate an E-channel system.

Most of the Sri Lankan e-channelling systems are web-predicated and utilizing the web-predicated system in mobile is Minimize Celerity, sometimes browser support is very impuissant and not facile to utilize on mobile. It is not user-friendly for mobile.

Most of the patients don't know what the specialization of the doctors is, and who are the best doctors of their diseases. They only know the diseases that they have at that moment.

Some of the patients know doctor's name but they will not know the details about the doctor, such as the hospital where, the doctor is available, time schedules of the doctors.

Some patients know all the details and they want to get an appointment with doctors, but they couldn't make it, because they will face interaction problems with a system, the system will perform the English language they can't understand it.

Patient have to pace on above difficulties in this kind of a challenging situation.

3. OBJECTIVES

3.1 MAIN OBJECTIVE

Connecting RASA NLU with dialogue management system and handling user conversations using RASA CORE.

3.2 SPECIFIC OBJECTIVES

Convert dialog into the json format

The dialogue which includes patients' medical questions and response medical answers are move to the RASA NLU. In that case want some proper method to move this dialogue. There are two methods available. First one is Json and the other one is Markdown method. In here most suitable method is Json format. Because Jason format is compatible for the mobile application development rather than Markdown format. By using Json format method system will enrich training data. Then convert that Sinhala dialogue into the Json format.

Automatically json format generation

Using Json format that data will move to the RASA NLU. After the retrieve of the dialogue into this stage, that Sinhala dialogue automatically convert in to the json format. This json query generation is a temporary process. It will not be storing any database. Intent and Entities available inside this json query.

Json query move to the RASA NLU

Then dialogue automatically convert into the Json format and it will move to the RASA NLU. Inside this RASA NLU, extract that Json query. Intent and Entities available

inside this Json query. If the Json query extract inside this RASA NLU both entity and intent parts automatically move.

RASA NLU move to the RASA core

Rasa NLU will be moved inside to the RASA core and it will extract inside this RASA core. Purpose of the RASA core is continuing the conversation. Rasa core will be managed patient's medical questions and response answers. It will identify the correct questions and generate reply for the related questions that patient asked.

4. METHODOLOGY

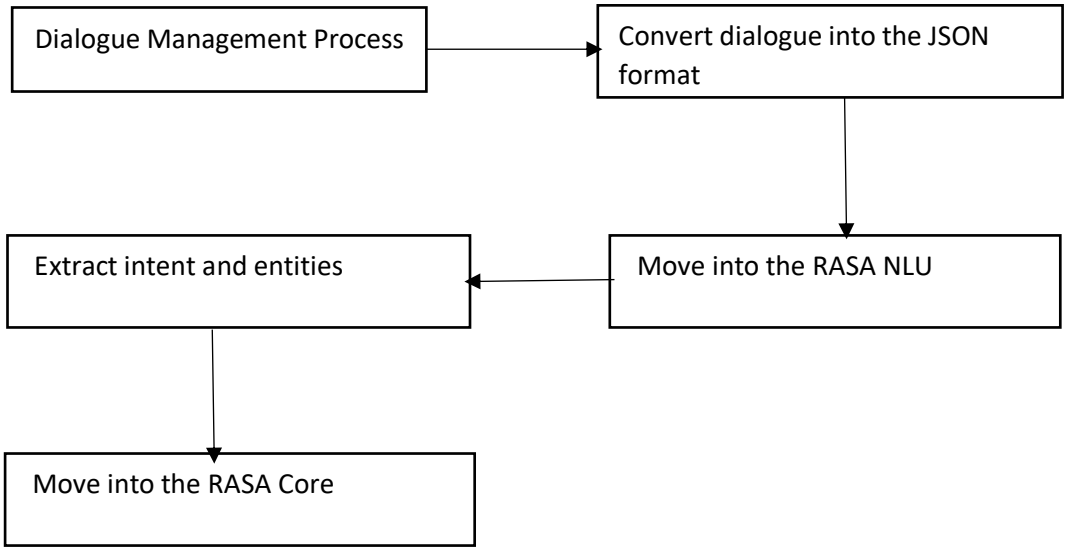

Figure 4.System Overview Diagram

In here Sinhala text message should move to the RASA NLU (Natural Language Understanding). For that case rasa provides two data training types. First one is Markdown format and second one is Json format.

Markdown format is the easiest data training format available inside the Rasa NLU. Listed using the unordered list syntax, e.g. minus -, asterisk *, or plus + examples for this markdown format. These examples can group by intent and entities. e.g. [entity] (entity name) [9].

Second data training type is Json format. In this Json format includes a top-level object called Rasa NLU data. "common examples", "entity synonyms" and "regex features" are the keys of RASA NLU data. The important part is "common examples".

```
{
  "rasa_nlu_data": {
    "common_examples": [ ],
    "regex_features" : [ ],
```

```
    "lookup_tables"  : [ ],
    "entity_synonyms": [ ]
  } }
```

Trained the model using “common examples”. After that include all of training examples inside to the “common examples” array. “Regex features” tool help the classifier to detect entities or intents in the RASA NLU and improve the performance of RASA NLU [9].

For those reasons JSON format is the most appropriate solution. And also, Jason format is compatible for the mobile application development rather than Markdown format. By using Json format method system will enrich training data so this project prefers using a User Interaction to do so. However, some power users interested in Markdown as it is really easy to understand, some points to consider.

The text message wants to move into the RASA NLU. That text message automatically convert into the JSON format. In that Json format has included intent (describes what type of messages) and entities (what specifically a user is asking about).

Below example sentence is for RASA NLU which is an open source natural language processing tool intent classification and entity extraction.

“ශ්‍රී ලංකාවේ අද කාලගුණය කෙසේද?”

This example’s intent is “ask_weather_location”

After the moving inside a RASA NLU it will extract entities and intents which are inside in a JSON format. Nlu.md file use to write the training data. Training data is usually stored in a markdown file. There are number of different components available in Rasa NLU, which together make a pipeline. This process can include it into the NLU model pipeline, once the training data is ready [10].

RASA NLU move to the RASA Core. Can build AI assistance using dialogue engine is the purpose of RASA CORE. It’s part of the open source RASA framework. RASA CORE use as a machine learning model trained to decide what to do next.

5. DESCRIPTION OF PERSONAL AND FACILITIES

- Understanding user Natural Language
- Natural Language Process
- Dialogue Management
- Connecting an NLU into Rasa framework

This project would be done by a group of four members and the research and development workload of the project is being distributed among all the members.

Detailed explanation of the assigned components has been discussed in the previous sections of the document.

Table 2.Member details with main objective

Member	Component
D.D.S Rajapakshe	Understanding user Natural Language
Kudawithana K.N.B	Natural Language Process
U.L.N.P. Uswatte	Dialogue Management
Nishshanka N.A.B.D	Connecting an NLU into Rasa framework

6. PROJECT REQUIREMENTS

Require engineering is a process of establishing

- The services that a customer requires from a system an
- The constraints under which it operates and is developed.

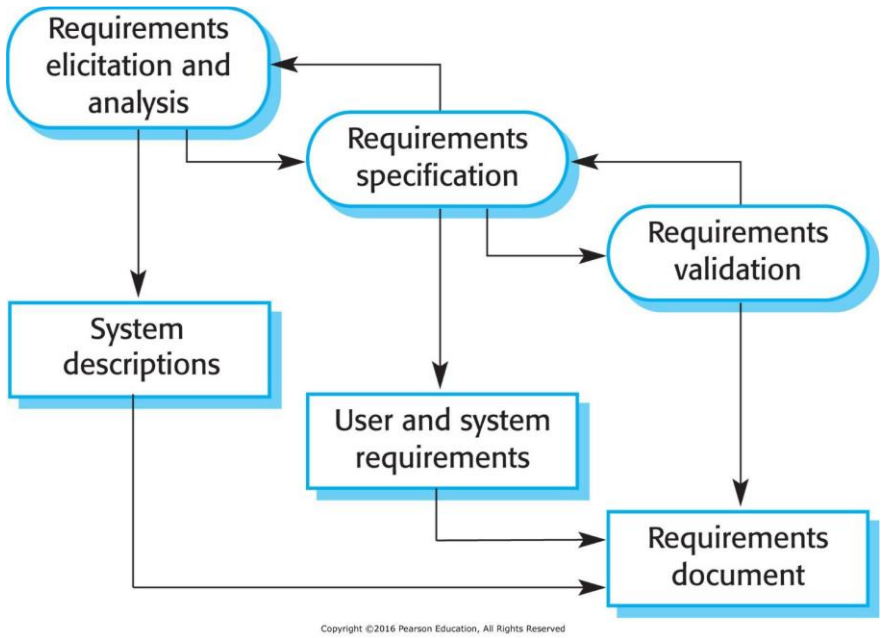

Figure 5.Requirements Engineering Process

Two types of Requirements Levels

1. User requirements
   - Platform to patient to e-channeling
   - Platform to dispensary/ hospital to register their system
   - Platform to doctor to register their system

2. System requirements

   A structured document setting out detailed descriptions of the system's functions, services and operational constraints

   System requirements are the configuration that a system must have in order for a hardware or software application to run smoothly and efficiently. Failure to meet these requirements can result in installation problems or performance problems.

   The former may prevent a device or application from getting installed, whereas the latter may cause a product to malfunction or perform below expectation or even to hang or crash.

   There are two types of System Requirements

   1. Functional requirements
      - ❖ Statements of services the system should provide, how the system should react to inputs and how the system should behave in particular situations.
      - ❖ Describe functionality or system services.
      - ❖ Depend on the type of software, expected users and the type of system where the software is used.
      - ❖ Functional user requirements may be high-level statements of what the system should do.
      - ❖ Functional system requirements should describe the system services in detail.

        - Patient registration to the system
        - Patient assistant through the system
        - User interact with system using Sinhala voice command
        - Doctor appointment for the patients
        - Patients can get the information about doctors, hospitals and channeling
        - Doctor channeling through the system
        - Medical service centers registration to their mobile application

   2. Non-functional requirements

- ❖ Constraints on the services or functions offered by the system often apply to the system rather than individual features or services. These define system properties and constraints

- ❖ Non-functional requirements may more critical that functional requirements. If these are not met, the system may be useless
  - Automated conversational mobile application
  - The quality of Sinhala Voice commands and the performance like System response time, throughput, utilization, static, volumetric
  - Reliability of the system data is the most important fact for both Patients and Medical Service Centres
  - System and its whole functionality (doctor channelling, information accessibility) should be available anytime for the patients.
  - How much easier to users when interact with this system's functionalities and
  - Total cost for system implementation, developments and use of the services
  - How the system localization for Sinhala users with interaction of Sinhala voice commands

Table 3.Metrics for specifying non-functional requirements

Property	Measure
Speed	Processed transactions/second User/event response time Screen refresh time
Size	Mbytes Number of ROM chips
Ease of use	Training time Number of help frames
Reliability	Mean time to failure Probability of unavailability Rate of failure occurrence Availability
Robustness	Time to restart after failure Percentage of events causing failure Probability of data corruption on failure
Portability	Percentage of target dependent statements Number of target systems

7. WORK BREAKDOWN STRUCTURE

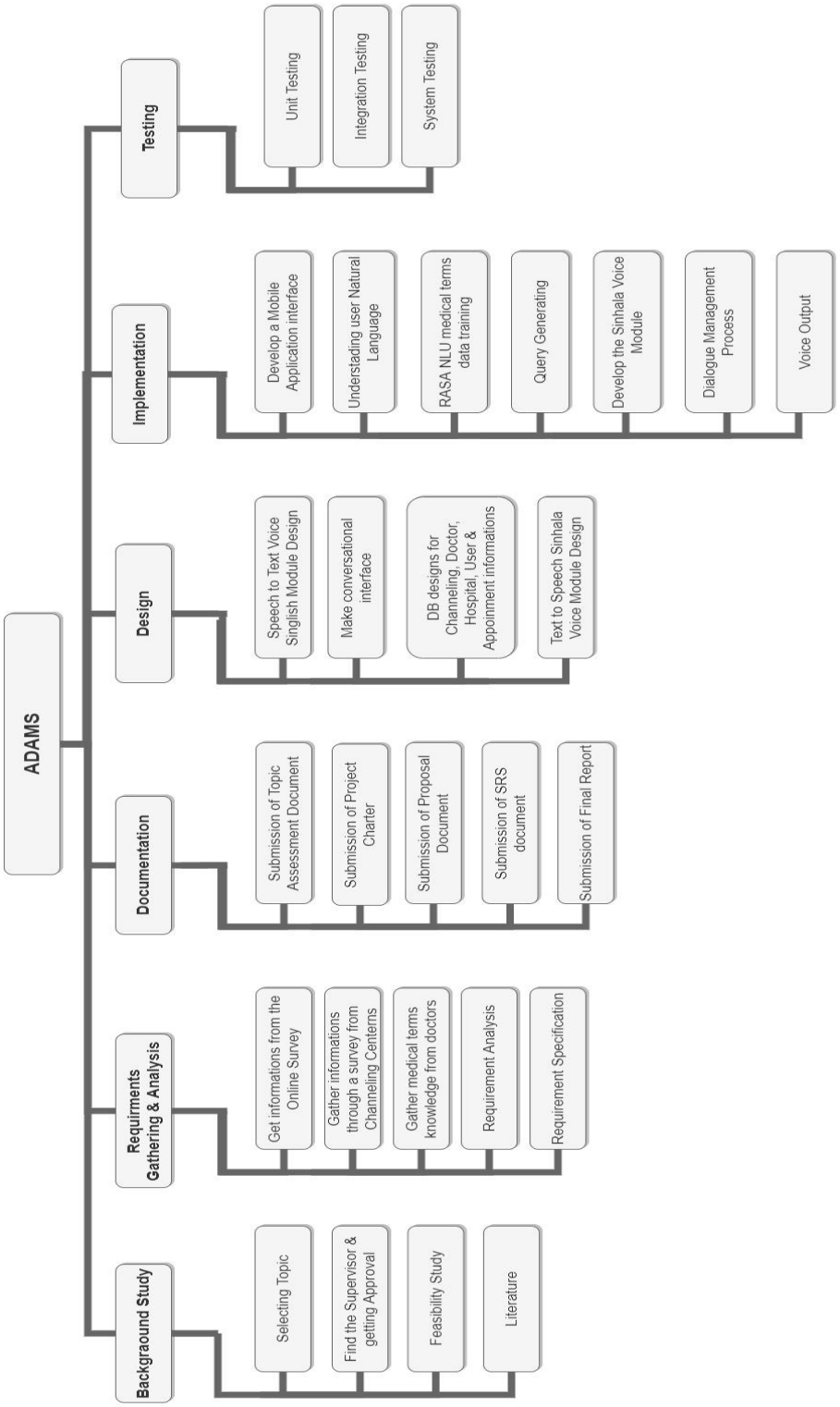

Figure 6. Work Breakdown Structure

8. BUDGET AND BUDGET JUSTIFICATION

This is the budget justification of this project. It's a sub part of the ADAMS. Stationaries, Communication and Printing costs are the three type of subcategory in this budget. Communication has the highest budget range. These prices can be changed according to the quantity and the value of dollar.

Table 4.Budget Analysis

Budget		**Cost**
1. Stationaries (12 months)		
1.1.A4 sheets	$0.38	
1.2.Rough sheets	$0.20	
1.3.Binding	$0.38	
1.4.Pens & Pencils	$0.20	
1.5.Other Stuff	$0.50	$1.66
2. Communication (12 months)		
2.1.Internet service	$5.78	
2.2.Phone Cost	$5.54	$11.32
3. Printing Cost (12 months)		
3.1.Reports	$0.76	
3.2.Photocopy cost	$1.51	$2.27
Total Cost		**$15.25**

9 . TIME PLAN

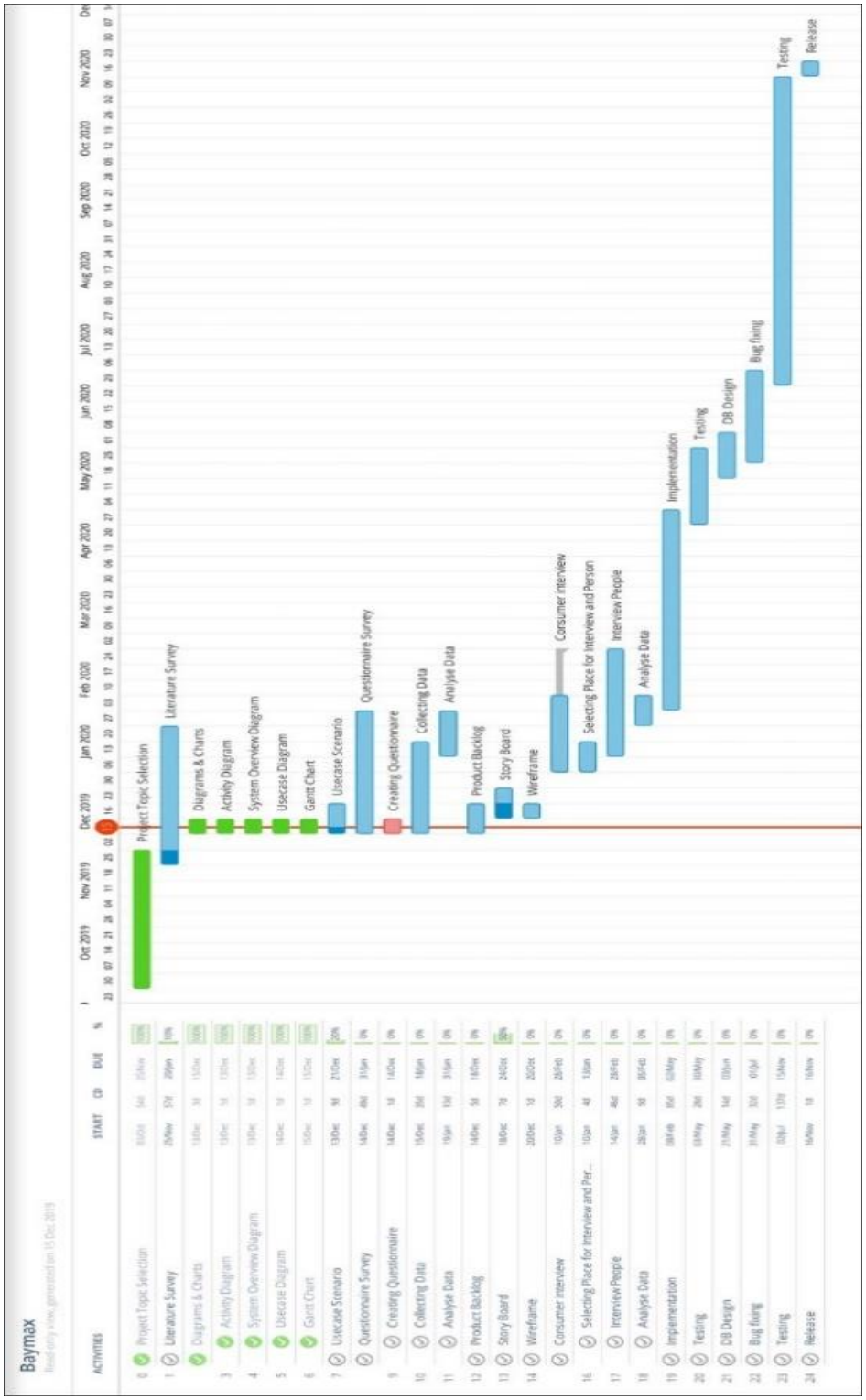

Figure 7.Gantt Chart

(https://drive.google.com/open?id=1GXsOwinBr2liyusBCX8MbBiUnvq0dlom)

10. REFERENCE

[1] M. B. M. S. P. Dr.Subarna Chatterjee, “Diabot,” Ramaiah University, Bangalore, 2019. [Online]. Available: https://www.academia.edu/39258471/Diabot_A_Predictive_Medical_Chatbot_using_Ensemble_Learning.

[2] V. V. Sandra V A, “Quinn: Medical Assistant for Mental Counseling using Rasa Stack,” Master of Computer Applications, College of Engineering, Trivandrum, Kerala, India, 06 June 2019. [Online]. Available: https://www.academia.edu/40173703/IRJET-_Quinn_Medical_Assistant_for_Mental_Counseling_using_Rasa_Stack.

[3] J. L. E. B. R. C.-j. P. Carlos Segura, “Chatbol, a chatbot for the Spanish “La Liga”,” May 2018. [Online]. Available: https://www.researchgate.net/publication/327426302_Chatbol_a_chatbot_for_the_Spanish_La_Liga.

[4] J. E. M. M. N. Vladimir Vlasov, “Dialogue Transformers,” 19 November 2019. [Online]. Available: https://arxiv.org/pdf/1910.00486.pdf.

[5] R. Jain, “Build a Conversational Chatbot with Rasa Stack and Python— Rasa NLU,” 30 March 2019. [Online]. Available: https://medium.com/@itsromiljain/build-a-conversational-chatbot-with-rasa-stack-and-python-rasa-nlu-b79dfbe59491.

[6] M. S. Z. RIZVI, “Learn how to Build and Deploy a Chatbot in Minutes using Rasa (IPL Case Study!),” 29 APRIL 2019. [Online]. Available: https://www.analyticsvidhya.com/blog/2019/04/learn-build-chatbot-rasa-nlp-ipl/.

[7] M. Pethani, “Making of Chatbot using Rasa NLU & Rasa Core,” 7 November 2019. [Online]. Available: https://chatbotslife.com/making-of-chatbot-using-rasa-nlu-rasa-core-part-1-7138c438581f.

[8] C. G. Giancaterino, “Emergency Chatbot using Rasa on Jupyter Notebook/Google Colaboratory,” 10 May 2019. [Online]. Available: https://towardsdatascience.com/emergency-chatbot-using-rasa-on-jupyter-notebook-google-colaboratory-2be9059f87cc.

[9] B. Sundaray, “Create Chatbot using Rasa Part-1,” 28 August 2019. [Online]. Available: https://towardsdatascience.com/create-chatbot-using-rasa-part-1-67f68e89ddad.

[10] S. Khan, “Building a Weather Chatbot with RASA NLU and RASA Core,” 29 May 2019. [Online]. Available: https://mc.ai/building-a-weather-chatbot-with-rasa-nlu-and-rasa-core/.