

AROGYA – AN INTELLIGENT AYURVEDIC HERB MANAGEMENT PLATFORM

Project ID: 2020-112

Mohamed Sahilan Fouzul Nilfa

IT17145930

Individual Final Project Thesis

Bachelor of Science Special (Honors) in Information Technology
Specializing in Software Engineering

Department of Computer Science & Software Engineering
Sri Lanka Institute of Information Technology
Sri Lanka

September 2020

AROGYA – AN INTELLIGENT AYURVEDIC HERB MANAGEMENT PLATFORM

Project ID: 2020-112

Mohamed Sahilan Fouzul Nilfa

IT17145930

Individual Final Project Thesis

(The dissertation was submitted in partial fulfilment of the requirement for the
Degree of Bachelor of Science Special (honors) In Information Technology
Specializing in Software Engineering)

Department of Computer Science & Software Engineering

Sri Lanka Institute of Information Technology

Sri Lanka

September 2020

DECLARATION OF THE CANDIDATES AND SUPERVISOR

I declare that this is our own work and this Dissertation does not incorporate without acknowledgement 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. Also, I hereby grant to Sri Lanka Institute of Information Technology, the nonexclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works.

Name	Student ID	Signature
Nilfa M.S. F	IT17145930	Nilfa M.S. F

The above candidates are carrying out research for the undergraduate Dissertation under my supervision.

Signature of the Supervisor

Date

.....

.....

Mrs. Lokesh Weerasinghe

Signature of the Co-Supervisor

Date

.....

.....

Ms. Ishara Weerathunga

ABSTRACT

Herbal remedies have been used for huge number of years like conventional medicine. In fact, herbal medicine is the establishment of modern medicine. Ayurvedic medicine prepared from the medicine are said to have no side effects. Each herb is unique in its medicinal properties with a good flavor. It acts as a perfect mechanism in bringing a balanced harmony between the mind and spirit. When compared to other synthetic drugs, ayurvedic herbal medicine works effectively fighting against various infections and diseases and thereby gaining quick recovery. In terms of Arogya, the process of medicinal leaves identification is followed by displaying a detailed description of the identified medicinal plant such as value of it, the medicinal recipes which can be prepared from it, for what diseases it would be useful. So, to achieve that Arogya has used real time object detection and classification. And also, users can view more information on each and every ayurvedic plant and receive a dynamic summary report on it. There is an explosive growth of information on Internet that makes extraction of relevant data from various sources, which leads tedious task for its users. Web documents can be extracted using information extraction and presented in a structured format. So, to generate a summarized report on every ayurvedic plant, information extraction along with Natural Language Processing and Deep Learning techniques were applied. Information can be extracted from structured, semi-structured, and unstructured data and need to be summarized by using abstractive summarization methods. Apart from information visualization, geographical location of the newly identified medicinal plants also can be tracked by Arogya. In consequence people will be able to find the locations of important medicinal plants through Arogya. Since most of the people are not well aware of the locations of rare medical plants, this would be a very prominent and useful function for the system as Arogya is kind of a social media platform of ayurvedic medicinal plants. The Locations will be displayed as a view of Sri Lankan geographical map with the identified medicinal plants.

Keywords – Abstractive, Deep Learning, Extractive Summarization, Geographical Information System, Information Extraction, Natural Language Processing

ACKNOWLEDGEMENT

I sincerely thank our Supervisor, Mrs. Lokesha Weerasinghe, Lecturer, Department of Computer Science and Software Engineering, and Sri Lanka Institute of Information Technology. I was motivated by her fullest support. That has led the project in a successful direction. I was able to achieve more efficient results because of her valuable feedback.

And also, I am very grateful to our research panel for encouraging us to drive in the correct path to reach our final goals. I would also like to thank Dr. Dharshana Kasthurirathna, the Senior Lecturer from the Department of Computer Science and Software Engineering, for giving us the necessary guidance and advice without any hesitation throughout the project, whenever we are in need. Finally, I thank all those people whose names were not mentioned, but who have helped me directly and indirectly throughout the project life span.

TABLE OF CONTENTS

DECLARATION OF THE CANDIDATES AND SUPERVISOR	i
ABSTRACT	ii
ACKNOWLEDGEMENT	iii
TABLE OF CONTENTS.....	iv
LIST OF FIGURES	vi
LIST OF TABLES.....	viii
1.0. INTRODUCTION.....	1
1.1. Background	2
1.2. Literature Review	3
1.2.1. Analysis of research papers	3
1.2.2. Analysis of existing systems	6
1.3. Research Gap	7
Research Problem	7
1.4. Objectives.....	8
1.4.1. Main Objectives	8
1.4.2. Specific Objectives	8
2.0. RESEARCH METHODOLOGY	9
Training phase.....	11
Encoder	11
Decoder.....	12
Inference Phase	13
2.1. System Architecture.....	14
2.2. Project Requirements	15
2.3. Expected Wireframes for the mobile application.....	18
2.4. Work Breakdown Structure	19
2.5. Gantt Chart.....	21
2.6. Commercialization	22
2.6.1. Target Audience	22
2.6.2. Market Space	22
2.6.3. Revenue Earning	22
2.7. Testing & Implementation	23
2.7.1. Implementation	23

2.7.2.	Testing.....	33
3.0.	RESULTS & DISCUSSION	43
3.1.	Results – Actual Interfaces.....	43
3.1.1.	Main Dashboard View.....	43
3.1.2.	User Login	44
3.1.3.	User Registration.....	44
3.1.4.	User Password Reset.....	45
3.1.5.	Dashboard.....	45
3.1.6.	Side Bar Menu	46
3.1.7.	Herbal Plants Info Summarizer	46
3.1.8.	Summary Comparison.....	48
3.1.9.	Location Tracker.....	49
3.1.10.	Insert Herbal Plant details.....	50
3.1.11.	Summary Results.....	52
3.2.	Research Findings	53
3.3.	Discussion.....	56
	Future Work	57
4.0.	THE SUMMARY OF MY CONTRIBUTION.....	58
5.0.	CONCLUSION.....	59
	REFERENCES.....	60
	GLOSSARY.....	62
	APPENDICES	63
1.	Use Case Diagram	63
2.	Class Diagram.....	64

LIST OF FIGURES

	Page
Figure 2. 1: Abstractive summarization process	9
Figure 2. 2: Seq2Seq model	11
Figure 2. 3: Encoder Process.....	12
Figure 2. 4: Decoder Process	12
Figure 2. 5: Inference Process	13
Figure 2.1. 1: System overview diagram	14
Figure 2.3. 1: Expected Mobile Interfaces	18
Figure 2.4. 1: Work Break Down Structure	20
Figure 2.5. 1:Gannt Chart	21
Figure 2.6.1. 1:Frontend Website Folder Structure.....	23
Figure 2.6.1. 2:Frontend Mobile Folder Structure	24
Figure 2.6.1. 3: Packages in pubspec.yaml	25
Figure 2.6.1. 4:User Data	26
Figure 2.6.1. 5:Geographical Location Details	27
Figure 2.6.1. 6:Herbal Plant Details.....	27
Figure 2.6.1. 7: Imported packages.....	29
Figure 2.6.1. 8: Reading Time & Fetch Text	31
Figure 2.6.1. 9:Summary Generation Function for Custom Text	31
Figure 2.6.1. 10:Extract Text from Single URL	32
Figure 2.6.1. 11:Extract Text from Multiple Sites	33
Figure 3.1.11. 1: Instance 1 - Word count of Generated Summary in comparison to Customized text.....	52
Figure 3.1.11. 2: Instance 2 - Word count of Generated Summary in comparison to Customized text.....	52
Figure 3.2. 1:Abstractive Text Summarization	53
Figure 3.2. 2: Search for Random Plants	54
Figure 3.2. 3:Location Tracking and GIS Technology	54

Appendices

Figure 1:Use Case Diagram	63
Figure 2:Class Diagram.....	64

LIST OF TABLES

	Page
Table 1.2.2.1: Comparing with existing system.....	6
Table 2.2. 1: Functional & Non-Functional Requirements.....	15
Table 2.6.2. 1:Test Case 1	34
Table 2.6.2. 2:Test Case 2.....	35
Table 2.6.2. 3::Test Case 3.....	36
Table 2.6.2. 4::Test Case 4.....	36
Table 2.6.2. 5:Test Case 5.....	37
Table 2.6.2. 6::Test Case 6.....	38
Table 2.6.2. 7:Test Case 7.....	39
Table 2.6.2. 8:Test Case 8.....	39
Table 2.6.2. 9:Test Case 9.....	40
Table 2.6.2. 10:Test Case 10.....	41
Table 3.1.1. 1:Interface 1	43
Table 3.1.2. 1:Interface 2	44
Table 3.1.3. 1:Interface 3	44
Table 3.1.4. 1:Interface 4	45
Table 3.1.5. 1:Interface 5	45
Table 3.1.6. 1:Interface 6	46
Table 3.1.7. 1:Interface 7	46
Table 3.1.8. 1:Interface 8	48
Table 4. 1: Summary of My Contribution.....	58

1.0. INTRODUCTION

In Sri Lanka there is an assortment of plant species that have been consumed for generations as herbal treatments, for control of diseases. Some of the diseases with complicated etiologies such as diabetes, arthritis, and cancer (for which a permanent cure is not in sight at present) have been known to be completely controlled or cured using these herbal remedies alone [1]. This traditional medicinal system, which has more than 3000 years of tested and proven efficacy, is still in use and generally the first approach for disease control by the locals, especially those who have been contracted with the stated diseases [2]. Typically, the herbs being used for medicinal purposes are evergreen in nature and are grown in the backyards of houses, and very little nurturing effort is required for their growth. Some of these herbs are even considered as weeds due to their high growth rates. Most Sri Lankans are familiar with the traditional medicinal system and are even able to identify or administer the herbs growing within their area of residence. Thus, the locals can be observed consuming these herbs to control a disease without the advice of a traditional medicinal practitioner, as they are familiar with the usage of these herbs because of the traditional knowledge, which has been passed down by their ancestors. With the arrival of western medicine, the traditional system of medicine was drastically declined to its lowest ebb. However, treatment of skeletal fractures, eye diseases, boils and carbuncles, snake bites and mental diseases remained under the preservation of a few dedicated medical men (Vedamahaththayas) who passed down these practices from generation to generation.

With these main concerns regarding Ayurveda we propose a system called Arogya is an herbal management platform which classifies, identifies and tracks the locations of ayurvedic plants in Sri Lanka through a single unit of mobile application. Since it is a kind of information sharing social media hub which can manipulate all three functionalities classification, location mapping, summary generation and visualization, in a single component. With the rise of internet, information readily available to us. If only someone could summarize the most important information for us Deep Learning is getting there. Through the latest advances in sequence to sequence

models, we can now develop good text summarization models. With the help of deep learning and sequence to sequence models summarization of information on medicinal plants will be generated.

1.1. Background

In Sri Lanka there is a lot of significance on herbal and endemic medicine. Majority prefer Ayurvedic medicine over Western medicine according to versatile factors such as less or rather no side effects. The fact that Ayurvedic medical system stood the test of time sums up to its worth as an alternative course of treatment for versatile ailments. However, a major drawback in the practice of endemic medicine is the difficulty in finding ingredients in Ayurveda recipes. Some of the benefits of Ayurvedic plants are, more affordable than western medicine, easier to obtain than prescription medicine, stabilizes hormones and metabolism, natural healing, strength in immune system, fewer side effects and cost effective. Today, people are more prone to consume western medication over Ayurvedic medication for their health issues because of their daily tight schedule, they do not have much time to spend in finding required recipes as the Ayurvedic physician said. Most of the patients begin to take western medication as soon as their diagnoses are made, so Ayurvedic treatments are usually undergone alongside and/or after western medical approach.

The existing systems like PlantSnap, Plantex and MedLeaf only focus on the medicinal leaves identification process. So that the users do not have multiple features like summarized information visualization and Location access of these herbals. Actually, the proposed system is more like a social media information sharing hub related to medicinal plants. Thus, the people who are looking for ayurvedic plants will have the ability to find the rare medicinal plants as well as they can get the information on locations too. So, through Arogya users can achieve many tasks at a single point and it prevents users from wasting time.

1.2. Literature Review

1.2.1. Analysis of research papers

According to many known sources, researchers have tried many methodologies to follow up dynamic information extraction and information summarization.

Most of the present Ayurvedic based applications follows only the medicinal plants identification process. The proposed system Arogya is capable of generating dynamic summaries regarding the identified medicinal plants. So, the system has to work with the online database, whenever the web pages get updated, the generated summary also should be adjusted in order to the current version of the changed website page. Hence it is kind of a real time information extraction, users also will get updated automatically. Currently the existing systems are just managing with the offline database of their own system, so the information will not be changed according to the timeline. So, the users will not get updated with the real time information of that particular medicinal plants.

In 2016, the research paper “An Automatic Multi document Text Summarization Approach Based on Naive Bayesian Classifier Using Timestamp Strategy” [3], discusses about an automatic text summarization approach is proposed which uses the Naive Bayesian Classification with the timestamp concept. This summarizer works on a wide variety of domains varying between international news, politics, sports, entertainment, and so on. Another useful feature is that the length of the summary can be adapted to the user’s needs as can the number of articles to be summarized. The compression rate can be specified by the users so that they can choose the amount of information he wants to imbibe from the documents.

The research presented in 2012, “A Semantic-Based Framework for Summarization and Page Segmentation in Web Mining” [4], introduces a framework that can effectively support advanced Web mining tools. The proposed system addresses the analysis of the textual data provided by a web page and exploits semantic networks to achieve multiple goals, the identification of the most relevant topics, the selection of the sentences that better correlates with a given topic, the automatic summarization of

a textual resource. The eventual framework exploits those functionalities to tackle two tasks at the same time, text summarization and page segmentation.

In 2017, the research paper “Text Summarization Techniques: A Brief Survey” [5], emphasizes various extractive approaches for single and multi-document summarization. It described some of the most extensively used methods such as topic representation approaches, frequency-driven methods, graph-based and machine learning techniques. Although it is not feasible to explain all diverse algorithms and approaches comprehensively in this paper, it provides a good insight into recent trends and progresses in automatic summarization methods and describes the state-of-the-art in this research area.

In 2018, the paper “Multi-Document Text Summarization Using Deep Learning Algorithm with Fuzzy Logic” [6], presented a multi-document text summarization scheme using an unsupervised deep learning algorithm along with fuzzy logic. Feature matrix with seven features from the set of sample dataset from DUC2002(Document ID: AP880911- 0016). The feature matrix is applied through the various levels of the RBM and finally the efficient text summary is generated. The result indicates that this method generates efficient text summary when compared to previous methods based on evaluation metrics.

In 2019, the paper “Deep Learning Architecture for Multi-Document Summarization as a cascade of Abstractive and Extractive Summarization approaches” [7], extends the state-of-the-art abstractive summarization architecture for multi-document summarization. The proposed architecture produces comprehensive summary on a topic by combining the Abstractive and Extractive summarization approaches in a cascade. The state-of-the-art approach for abstractive summarization using pointer-generator model is limited to single-document summarization. Summarization of multiple news articles on a topic can be handled one by one independently which results in multiple summaries for the same topic with possible redundancy. In order to avoid redundancy, the authors propose Extractive summarization of the multiple summaries as the second phase in the proposed cascade framework. The effectiveness of the framework was established using ROUGE metric.

In 2011, the paper “Frequently Asked Questions Web Pages Automatic Text Summarization” [8], presented the preliminary results of our FAQ Web Pages automatic text summarization approach. Their approach is based on making use of the visual cues existing in the text of the questions and answers of the web page to detect Q/A segments. In addition, we devised a new combination of selection features to perform the actual summarization task. These features are namely, question-answer similarity, query overlap, sentence location in answer paragraphs and upper-case words frequency. The different document features were combined by a home-grown weighting score function. Pilot experimentations and analysis helped them in obtaining a suitable combination of feature weights. In fact, the evaluation results showed that in general this approach seems to be promising where the overall average for all pages indicates statistically significant improvements for their approach in 62% of the cases when compared with another summarization tool.

In 2004, this paper “World Wide Web Site Summarization” [9], developed a new approach for generating summaries of Web sites. The approach applies machine learning and natural language processing techniques to extract and classify narrative paragraphs from the Web site, from which key-phrases are then extracted. Key-phrases are in turn used to extract key-sentences from the narrative paragraphs that form the summary, together with the top key-phrases. The summaries are demonstrated, although not in proper prose, are as informative as human-authored summaries, and significantly better than browsing the home page or the site for a limited time. The approach should be easy to transform into proper prose by human editors without having to browse the Web site. The performance of method depends on the availability of sufficient narrative content in the Web site, and the availability of explicit narrative statements describing the site.

In consequence above reviews, Arogya acts as a single platform of resource access of multiple functionalities to overcome all these difficulties in the current products in the market. Arogya’s main goal is to be the feasible single unit of medicinal plants information hub which will satisfy all the user requirements regarding ayurvedic plants.

1.2.2. Analysis of existing systems

Table 1.2.2.1: Comparing with existing system

References	Smart Visualization	Real time Summary generation on information	Using GIS for Location tracking
H. P. Edmundson, "New methods in automatic extracting"[1]	✗	✗	✗
J. Kupiec, J. Pedersen, F. Chen, "A trainable document summarizer" [2]	✗	✗	✗
Neeraj Kumar, Peter N Belhumeur, Arijit Biswas, David W Jacobs, W John Kress, Ida C Lopez, João VB Soares, "Leafsnap: A computer vision system for automatic plant species identification" [3] research was done on Identification of plant leaf using computer vision techniques	✗	✗	✗
D. R. Radev, H. Jing, M. Stys, D. Tam, "Centroid-based summarization of multiple documents" [5], Information Processing and Management	✗	✗	✗
Arogya our proposed System	✓	✓	✓

1.3. Research Gap

Research Problem

Since Ayurveda is descending from generation to generation, the precious knowledge of Ayurveda is restricted for a group of specific people. So, most of the ordinary people are unable to get the information on medicinal plants. And also finding the geographical locations of the medicinal plants will be a tedious task due to this knowledge gap between ordinary people and the particular generalized group of people. In order to get the information of medicinal plants, people from urban areas have to travel back to villages to meet ayurvedic physicians (Weda Mahaththaya). In terms of proposed component, the issues we currently facing will be removed hopefully.

As mentioned in the literature review the existing systems are lack of many functionalities when compared to the proposed system Arogya. So, the problems in the existing systems will be the identified gaps which we need to satisfy through Arogya. Hence the proposed system has to cope with multiple functionalities such as visualization of the final output, generation of summary based on extracted real time web information on medicinal plants, tracking the geographical locations of the rare herbs to share with the required users.

Implementing the component as a single platform of accessing multiple functionalities, the targeted users can save their time and effort of finding other resources to fulfill their requirements related to Ayurveda.

1.4. Objectives

1.4.1. Main Objectives

The main objective of this proposed component is to cooperate extractive information summarization and location mapping with smart visualization on ayurvedic plants in Sri Lanka

1.4.2. Specific Objectives

- Summary generation with the support of Deep learning, Natural Language Processing and Information Extraction

Information of detected plants will be dynamically extracted from web pages and summarized using deep learning

- Tracking Locations with Geographical Information System so, the users can get the location information of plants

2.0. RESEARCH METHODOLOGY

Main research area of this component was based on natural language processing techniques. Target was to extract dynamic information from multiple web pages and generate a summary on ayurvedic plants. So, in order to achieve this goal, many techniques were followed to select the best model out of them. Thus, several existing algorithms were analyzed, and accuracy was tested in order to select the suitable algorithms to extract the required information from dynamic web pages to generate a summary.

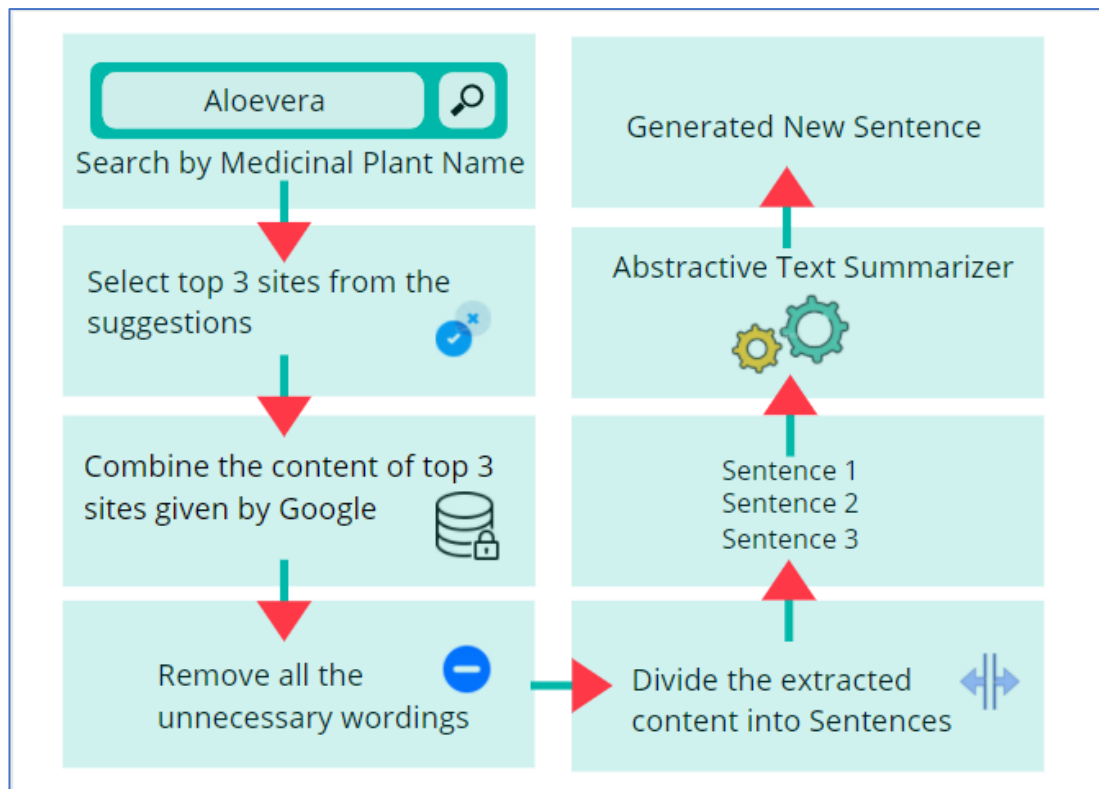


Figure 2. 1: Abstractive summarization process

The process of text summarization can be defined as generating an incisive and communicative synopsis while conserving the main content of the description and the overall outline. In general, for the process of text summarization, two main approaches are being executed called Extractive Summarization and Abstractive Summarization. As the methodology for this research, Abstractive Summarization was used. Instead of

existing sentences, new sentences were generated which were totally differentiated from the original text. In contrast, what extractive summarization did was prepare the summary with the most important sentences which were extracted from the original text.

According to our working plan, the input was a set of descriptions of medicinal plants which were extracted from highly ranked multiple websites and the output was a short summary of sequence of words. Hence, this was modeled as a Many to Many Seq2Seq problem. Encoder and the Decoder are the two major components of a Seq2Seq model. In this research component Long Short-Term Memory was used as the encoder and the decoder. The Encoder-Decoder was set in two phases as training phase and Inference phase. As the first step Encoder and the Decoder were set in the training phase. The model was trained to predict the target sequence offset by one timestep. The way Encoder and the Decoder were set was as follows: the whole input sequence was read by the long short-term memory model at each timestep, one word was counted. Information at each timestep was processed and related information present in the input was captured. Decoder also worked as a Long Short-Term Memory network. Entire target sequence was read word by word to predict the same sequence offset by the decoder. Decoder was capable of predicting the next word in the sequence given the previous word. At the inference phase after doing training, the model was evaluated on a new source sequence. Target sequence was an unknown one. Inference Architecture was to be set up to decode a test sequence. At the final stage, the user retrieved a summarized paragraph on the specific Ayurvedic plant.

Expectation is to build a text summarizer where the input is a long sequence of words extracted from web pages and the output is a short summary of sequence of words. Thus, we can model this as a Many-to-Many Seq2Seq problem. Below is a typical

Seq2Seq model architecture:

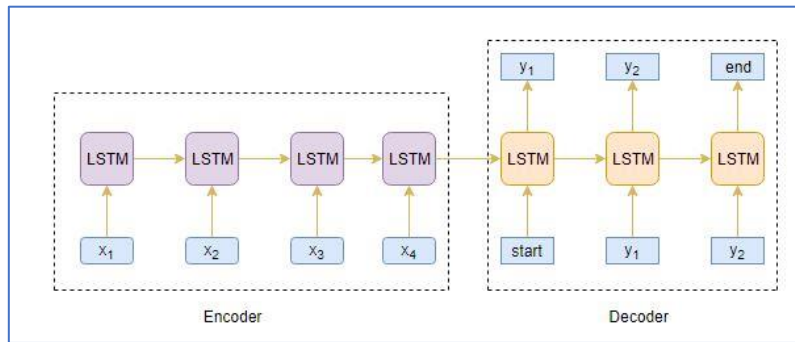


Figure 2. 2: Seq2Seq model

There are two major components of a Seq2Seq model as, Encoder and the Decoder.

Generally, variants of Recurrent Neural Networks (RNNs), i.e. Gated Recurrent Neural Network (GRU) or Long Short-Term Memory (LSTM), are preferred as the encoder and decoder components. This is because they are capable of capturing long term dependencies by overcoming the problem of vanishing gradient.

We can set up the Encoder-Decoder in 2 phases:

- Training phase
- Inference phase

Training phase

In the training phase, we will first set up the encoder and decoder. We will then train the model to predict the target sequence offset by one timestep. Let us see in detail on how to set up the encoder and decoder.

Encoder

An Encoder Long Short-Term Memory model (LSTM) reads the entire input sequence wherein, at each timestep, one word is fed into the encoder. It then processes the information at every timestep and captures the contextual information present in the input sequence.

Below diagram illustrates this process:

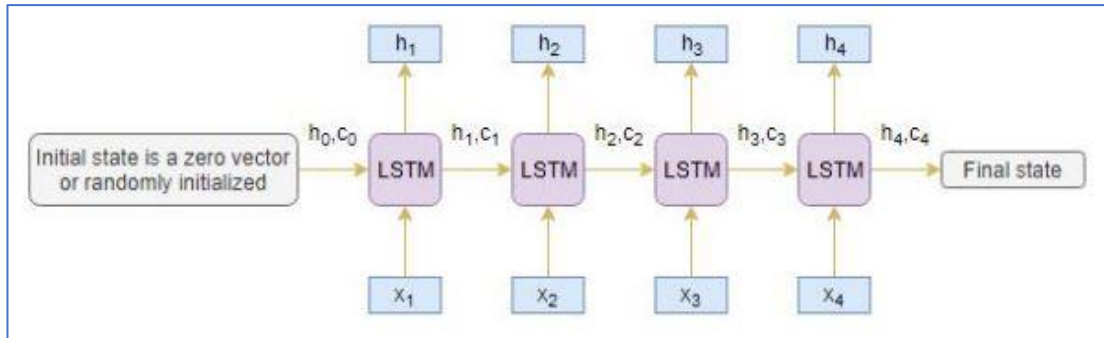


Figure 2. 3: Encoder Process

The hidden state (h_i) and cell state (c_i) of the last time step are used to initialize the decoder. Remember, this is because the encoder and decoder are two different sets of the LSTM architecture.

Decoder

The decoder is also an LSTM network which reads the entire target sequence word-by-word and predicts the same sequence offset by one timestep. The decoder is trained to predict the next word in the sequence given the previous word.

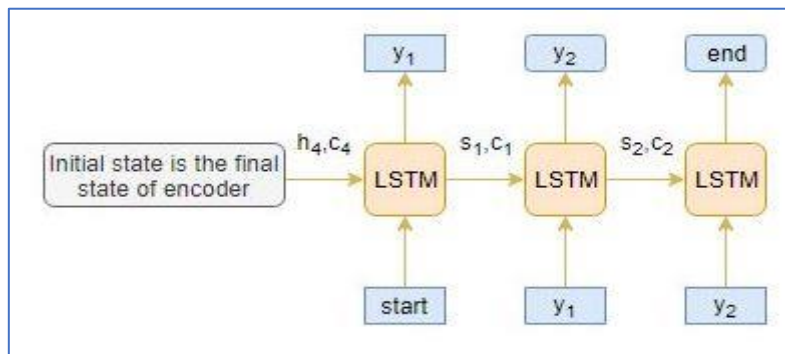


Figure 2. 4: Decoder Process

<start> and <end> are the special tokens which are added to the target sequence before feeding it into the decoder. The target sequence is unknown while decoding the test sequence. So, we start predicting the target sequence by passing the first word into the

decoder which would be always the <start> token. And the <end> token signals the end of the sentence.

Inference Phase

After training, the model is tested on new source sequences for which the target sequence is unknown. So, we need to set up the inference architecture to decode a test sequence:

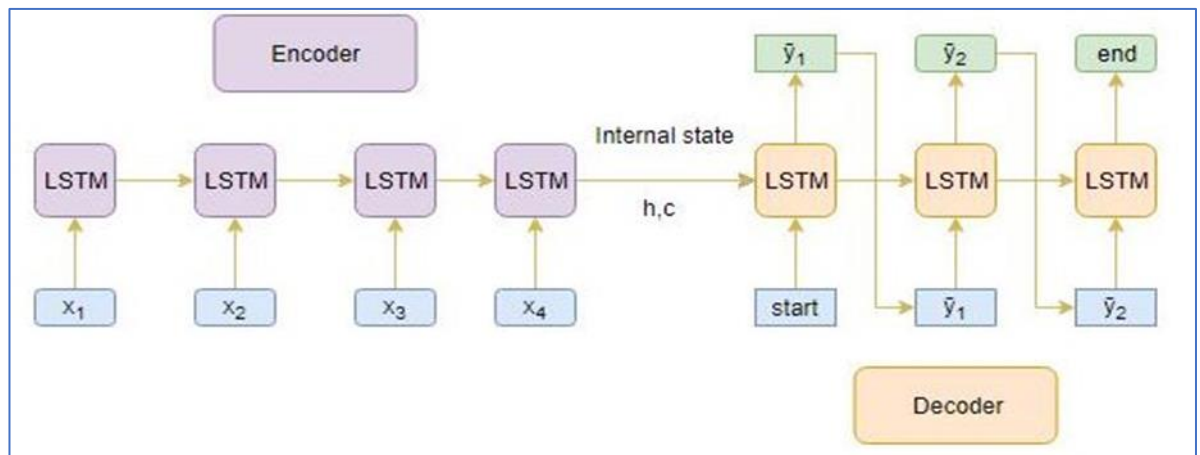


Figure 2. 5: Inference Process

Here are the steps to decode the test sequence:

1. Encode the entire input sequence and initialize the decoder with internal states of the encoder
2. Pass <start> token as an input to the decoder
3. Run the decoder for one timestep with the internal states
4. The output will be the probability for the next word. The word with the maximum probability will be selected
5. Pass the sampled word as an input to the decoder in the next timestep and update the internal states with the current time step
6. Repeat steps 3 – 5 until we generate <end> token or hit the maximum length of the target sequence

So that, with the support of encoder-decoder architecture Arogya will be able to manage the summary generation functionality.

2.1. System Architecture

Arogya is consisted with four main functionalities and this component is regarding visualization and information summarization. And also, the system is capable of location tracking

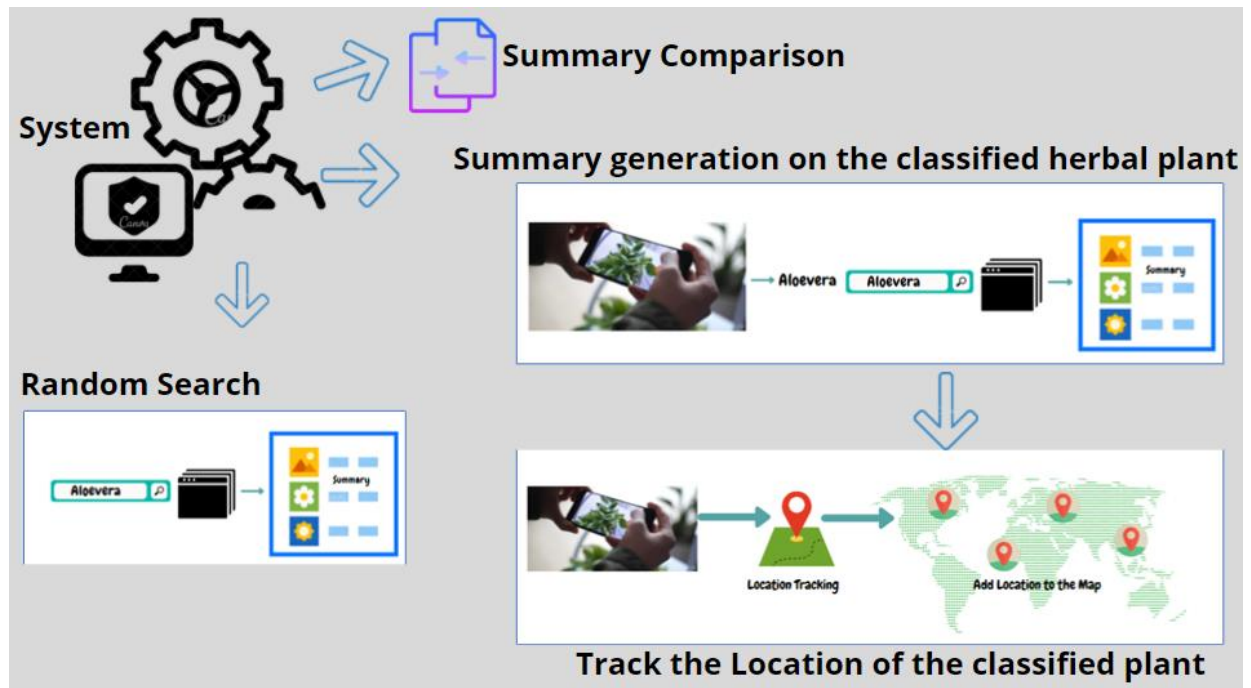


Figure 2.1. 1: System overview diagram

2.2. Project Requirements

Table 2.2. 1: Functional & Non-Functional Requirements

Functional	Non - Functional
Display information on top of the plant species	Application should be able to give the results as fast as possible
Summary should be generated regarding the medicinal plant	Higher Accuracy required
Application should be able to track the location of users	User friendly Interfaces should be provided
Medicinal plant's description should be available for offline view	User Experiences should be properly managed in order to achieve a specific functionality
Locations of plants should be able to view as a Map view	

Consideration of the aspects of the system

Standards: Coding standards were followed while doing the coding. Developers has been taught the coding techniques by the coding standards. Object-oriented concepts were followed to maintain coding standards. Important main points were included as comment lines within our project code. When writing reports and referencing, I followed IEEE format.

Social aspects

This mobile application can be used by any person who is not aware about but keen to experience Ayurveda medication worldwide. For example, people who use and wish to use ayurvedic medicines and treatment, researchers in the field of botany, medicine, chemical structure analysis, agriculture, ayurvedic medicinal practitioners, taxonomists, forest department officials, those who are involved in the preparation of ayurvedic medicines and others who are concerned with plant studies, as well as doctors, students, locals, foreigners, Ayurvedic plant sellers and many more can use this application wisely. Therefore, a great appreciation as well as a demand can be expected from the society for this mobile application. Especially, there is no age limitations for the users, no need of advanced computer literacy, as well as no need of advanced knowledge in Ayurveda field to use this app. Only a simple knowledge of how to use a smart mobile phone will be sufficient for this. In addition, the name of the identified plant is given in native as well as in the scientific name of the plant, which would support any person including foreigners. So, this would be very beneficial for the whole society which would do a great service in uplifting our ancient Ayurveda, which is anyway better than modern medicine due to its naturality.

Security aspects

Security of this mobile application is high, because only a registered user can use this app and they have to log in before they use this application. Also, user has to provide suitable username and password (correct credentials) to access this mobile app which

were given at the registration process. Credentials of all the registered users are stored in the firebase database with high security rules, so there will not be any unauthorized access for them because it is pretty similar to the security rules followed when creating the Facebook application. (this mobile application is created as a centralized social media platform) Server-side security is also very high because all the services are hosted in AWS (Amazon Web Services) with IAM (Identity Access Management) user credentials.

Ethical aspects

This application is not causing any harm, injury, or damage to the user or his/her mobile. No unethical behaviors, rules or principles have been followed in the implementation of this mobile application. This will be capable of identifying majority of ayurvedic plants through any part of it like leaves, root, fruit, flower, etc. if retrained with more datasets by professionals. However, this does not replace the role of a plant taxonomist or an Ayurveda doctor, but it supports any person without any background knowledge about Ayurveda to classify a particular Ayurveda plant and to know about the medicinal value of it hopefully. This would more be a learning resource for almost all persons who need to experience Ayurveda.

Limitations

- This application is currently built only in English. Further, this can be applied in native languages such as Sinhala, Tamil as well as in any of the other languages preferred.
- Since the system is designed only as a mobile application, later can be improvised to a web application with the same functionality and content.
- Currently this mobile application is developed only for Android users, so have to consider about other mobile platforms and operating systems as well.
- The development strategy in this approach is only to identify 5 categories of plants, but the same strategy and methodology can be used and extended to identify any ayurvedic herb worldwide.
- If the user ends up with doubts and clarifications regarding this procedure, this application can be facilitated with consultation help from Ayurveda doctors.

2.3. Expected Wireframes for the mobile application

Following are some of the wireframes expected to be implemented according to the function of leaf classification in Arogya.



Figure 2.3. 1: Expected Mobile Interfaces

2.4. Work Breakdown Structure

01. Problem Identification

1.1 Determining information requirements

02. Information Gathering

2.1 Reference online resources

2.2 Reference related books

03. Analysis the System

3.1 Preliminary Investigation

3.2 Further Analysis

04. Designing the system

4.1 Database designing

4.2 Screen Design

4.3 Test plan

4.4 Design Results Generating

05. Develop the System

5.1 Develop the User Interfaces

5.2 Reports Development

5.3 Validations

06. Testing and Debugging

6.1 Derivation of test plan

6.2 Execution of test plan

6.3 Correction of errors

07. Documentation

7.1 Develop the Documentation

7.2 Finalize the Documentation

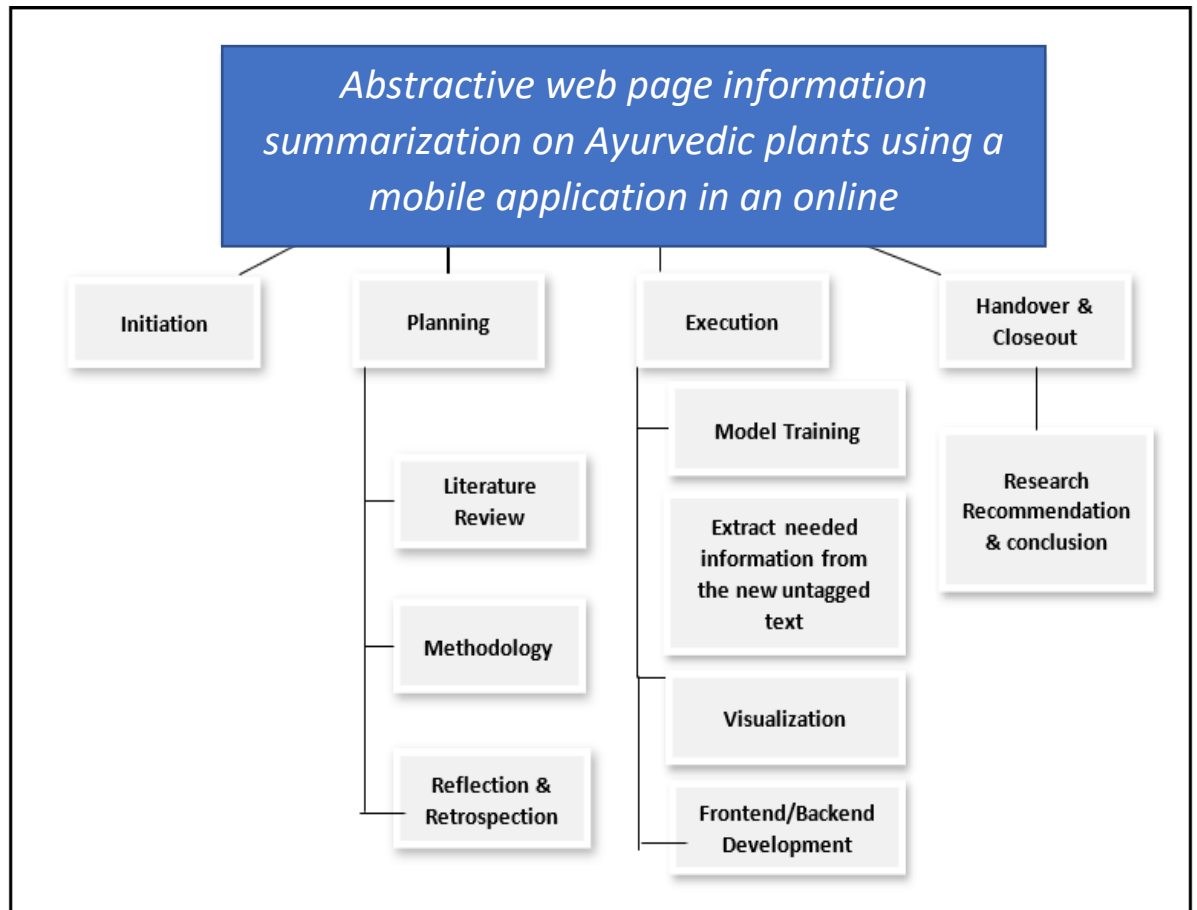


Figure 2.4. 1: Work Break Down Structure

2.6. Commercialization

2.6.1. Target Audience

- People who use ayurvedic treatment
- Researchers in the field of botany, medicine, chemical structure analysis, agriculture, ayurvedic medicinal practitioners, forest department officials, those who are involved in the preparation of ayurvedic medicines and others who are concerned with plant studies
- Doctors, Students, Locals and Foreigners
- Ayurvedic plant sellers

2.6.2. Market Space

- No age limitations for the users
- No need of advance computer literacy
- No need of advance knowledge in Ayurveda field

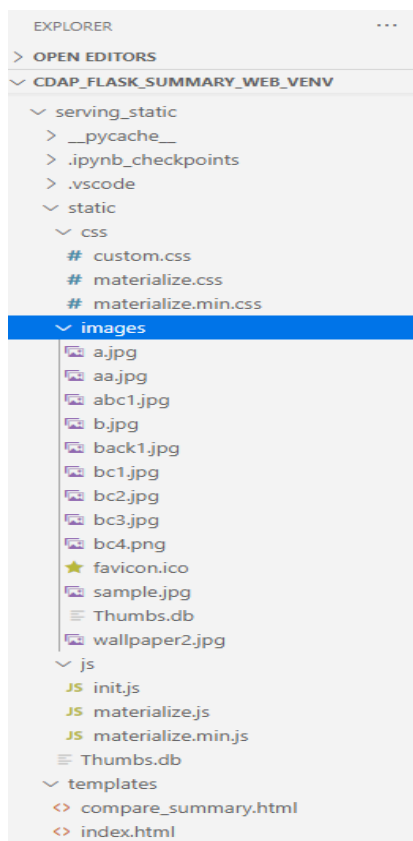
2.6.3. Revenue Earning

- Through subscription fee
- Revenue via additional services
- Add click sense

2.7. Testing & Implementation

2.7.1. Implementation

The client side runs as a mobile application which was built on top of Android and Flutter (Dart). The User Login and Registration along with Profile Management and Geographical Location Mapping run on Mobile client itself while the Summary Generation component runs on the web server. Front-end technologies are based on HTML, CSS. Apart from this Bootstrap has been used in order to create responsive webpages. Server side runs on Flask server, which is a python-based framework and the implementation continues with Python, TensorFlow environment and using Anaconda navigator. Front-end Flutter web view call the web API to run the Summary generator process, which run on flask server. SpaCy, NLTK, Sumy, and Gensim are natural language processing python libraries which are related to summary generation process.



Frontend Implementation

This component comes with two versions of Frontends

- 1.0. Web Application
- 2.0. Mobile Application

Web Application

Environment: VS Code IDE

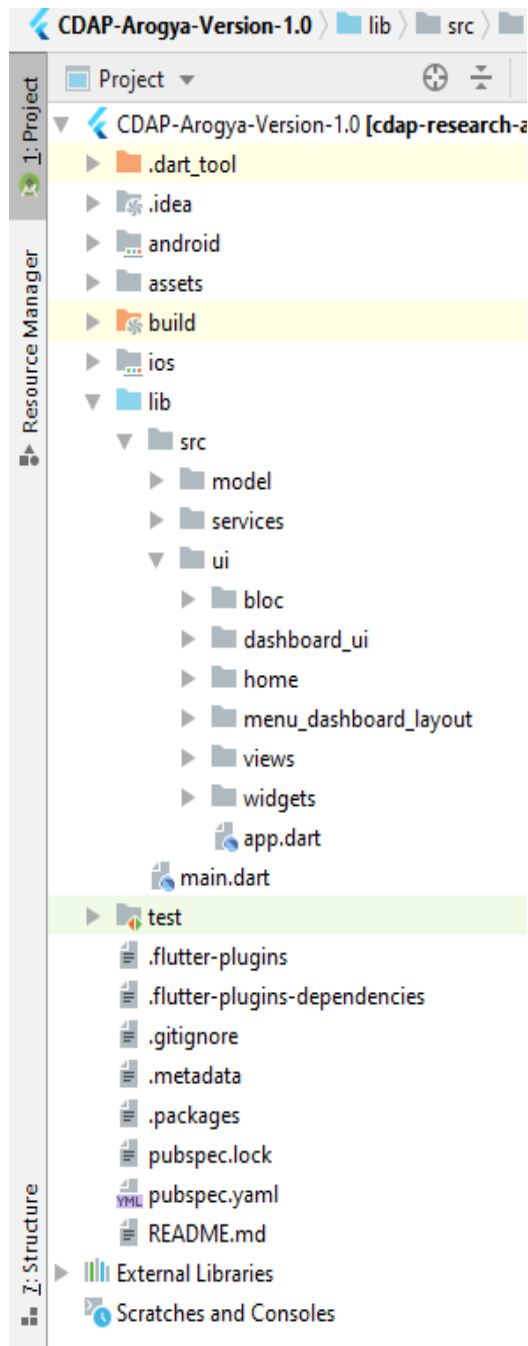
API end point of the Web Application is called by the Mobile Web View component.

Figure in the left side displays the folder structure of the web application.

compare_summary.html and **index.html** are the main web application templates.

Figure 2.6.1. 1:Frontend Website Folder Structure

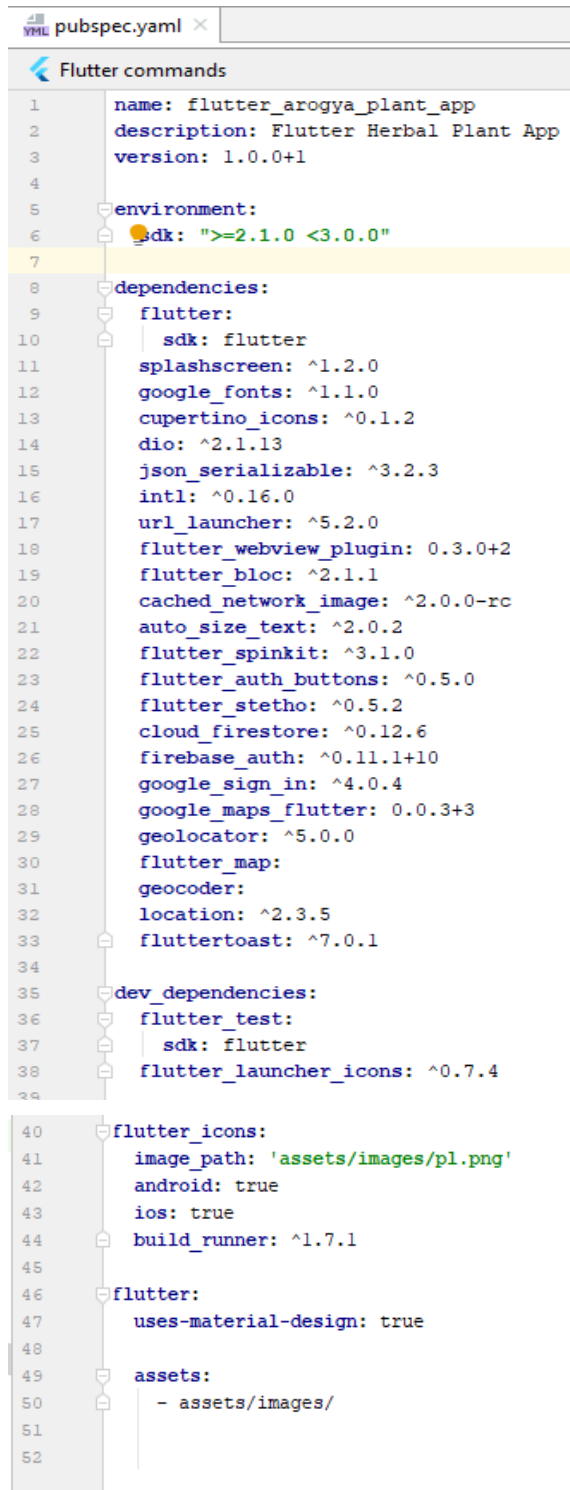
Mobile Application



Environment: Android IDE with Flutter (Dart Language)

Left side figure shows the Directory structure of the Android project

Figure 2.6.1. 2:Frontend Mobile Folder Structure



Many types of libraries were used in order to build up the application

Pubspec.yaml file indicates the all libraries which were used

Figure 2.6.1. 3: Packages in Pubspec.yaml

Backend Implementation

Database Structure

Firebase has been used as the Database to store User Profile Information, Herbal Plants and Geographical Locations details.

As shown in the following figure, **userData** is the collection which stores all the records related to a particular user on a time session.

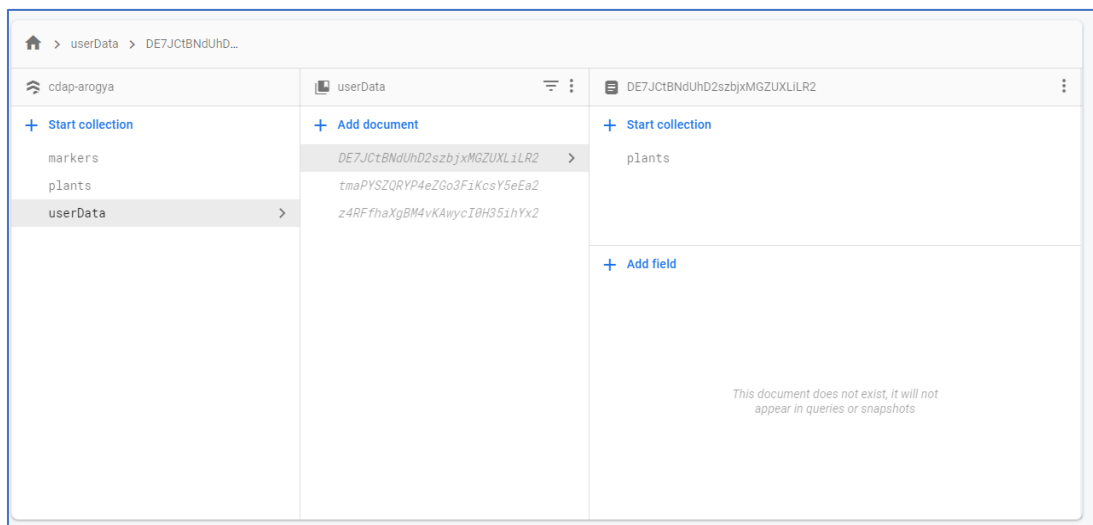


Figure 2.6.1. 4:User Data

As shown in the following figure, **markers** is the collection which stores all the records related to a particular Geographical Location.

Each document consists with three main fields:

- clientName
- location
- plantName

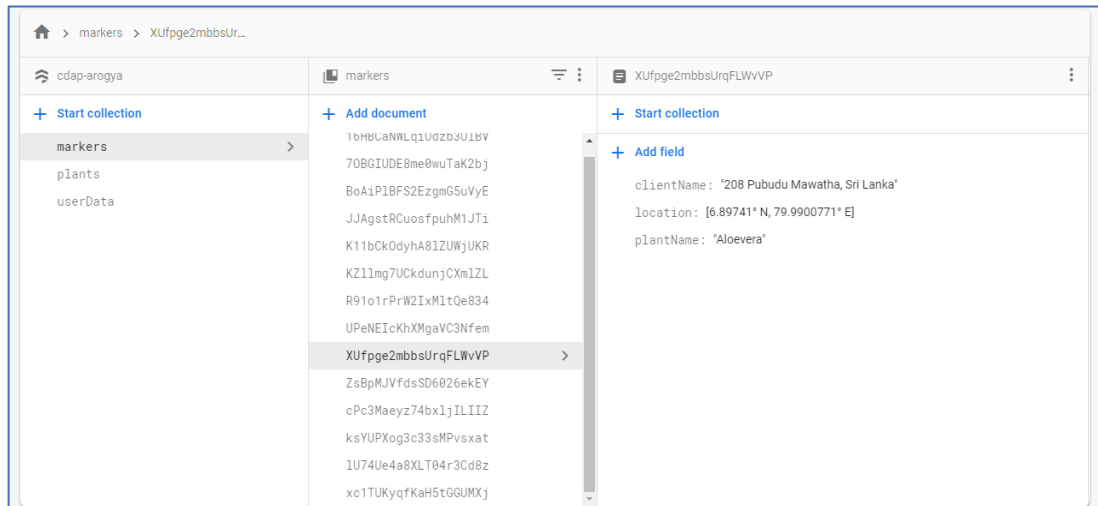


Figure 2.6.1. 5:Geographical Location Details

Following figure indicates the plants collection which is related to the information of the herbal plants

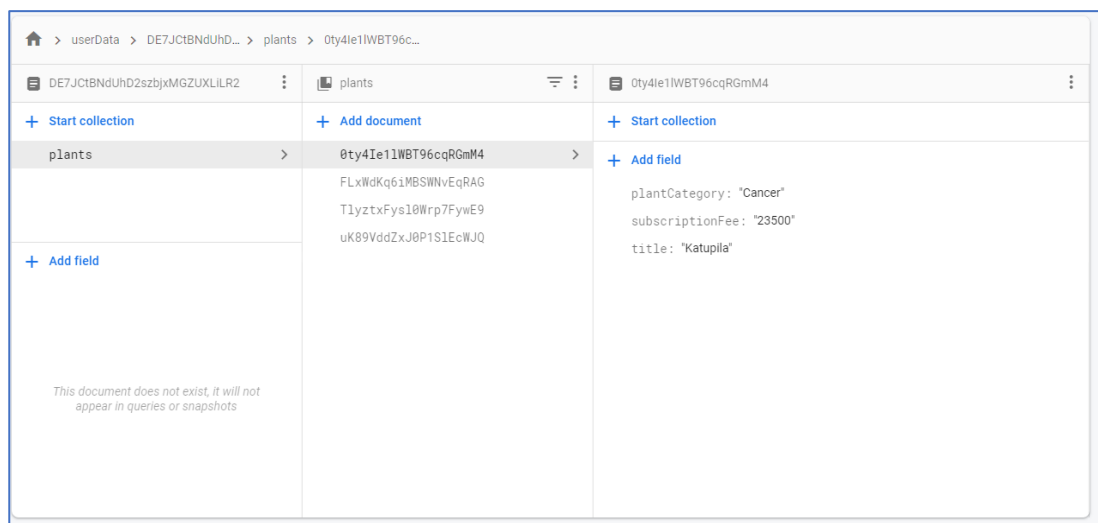


Figure 2.6.1. 6:Herbal Plant Details

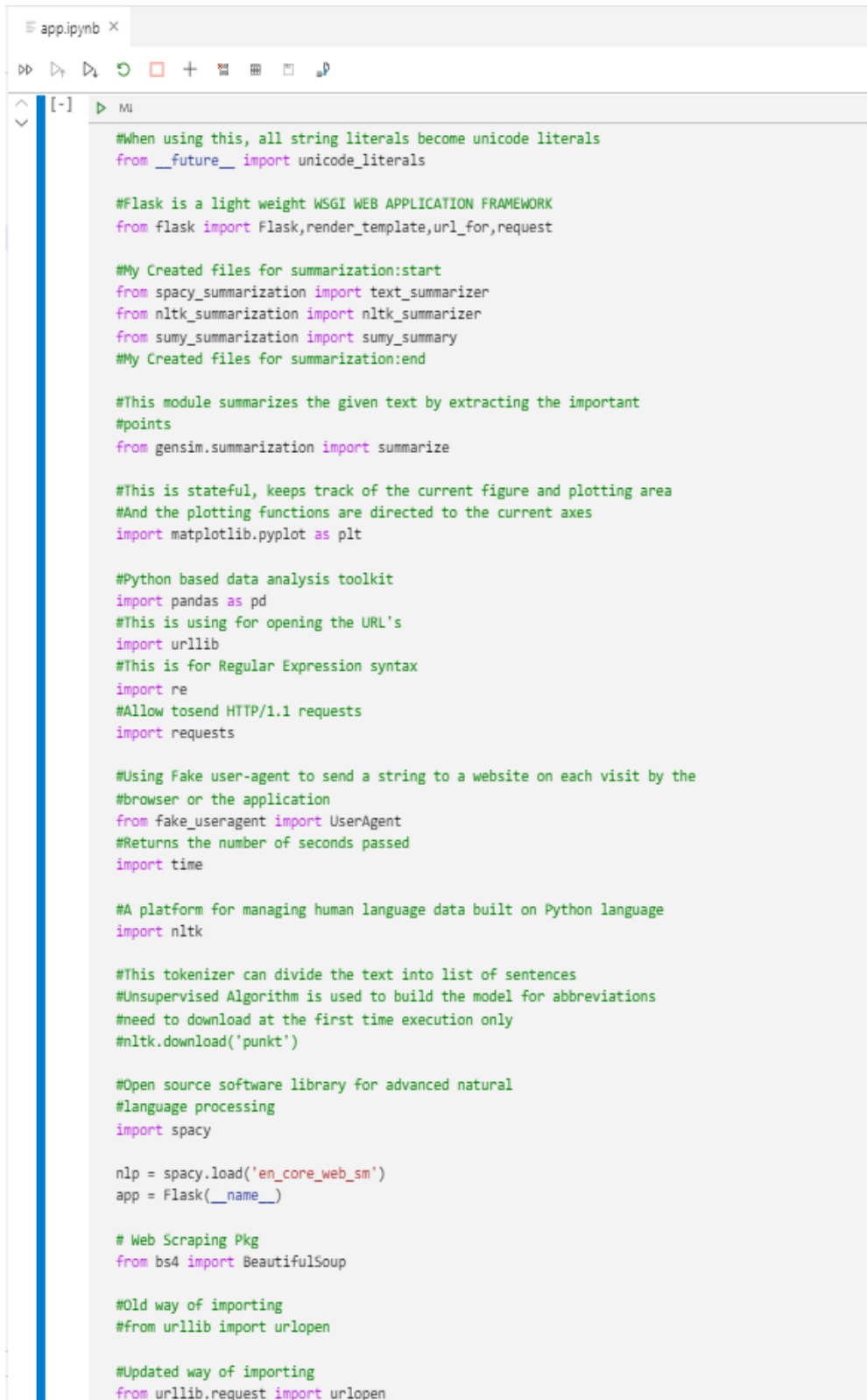
Multi-Level user authentication is also implemented in this system. Such as, Google Sign In, Email User Sign In, Anonymous Sign In.

Server-Side Implementation

Used Technologies:

- Python, TensorFlow-Backend installed Anaconda Navigator
- Flask web application Framework
- IDE: Visual Studio Code

Following code snippet indicates the imported libraries to the project, you can check the comments to get an idea about its contribution.



```

app.ipynb x
>> > > > + > > > >
[-] ▶ MI

#When using this, all string literals become unicode literals
from __future__ import unicode_literals

#Flask is a light weight WSGI WEB APPLICATION FRAMEWORK
from flask import Flask,render_template,url_for,request

#My Created files for summarization:start
from spacy_summarization import text_summarizer
from nltk_summarization import nltk_summarizer
from sumy_summarization import sumy_summary
#My Created files for summarization:end

#This module summarizes the given text by extracting the important
#points
from gensim.summarization import summarize

#This is stateful, keeps track of the current figure and plotting area
#And the plotting functions are directed to the current axes
import matplotlib.pyplot as plt

#Python based data analysis toolkit
import pandas as pd
#This is using for opening the URL's
import urllib
#This is for Regular Expression syntax
import re
#Allow to send HTTP/1.1 requests
import requests

#Using Fake user-agent to send a string to a website on each visit by the
#browser or the application
from fake_useragent import UserAgent
#Returns the number of seconds passed
import time

#A platform for managing human language data built on Python language
import nltk

#This tokenizer can divide the text into list of sentences
#Unsupervised Algorithm is used to build the model for abbreviations
#need to download at the first time execution only
#nltk.download('punkt')

#Open source software library for advanced natural
#language processing
import spacy

nlp = spacy.load('en_core_web_sm')
app = Flask(__name__)

# Web Scraping Pkg
from bs4 import BeautifulSoup

#Old way of importing
#from urllib import urlopen

#Updated way of importing
from urllib.request import urlopen

```

Figure 2.6.1. 7: Imported packages

nlk_summarization.py X

serving_static > nltk_summarization.py

```

1  import nltk
2
3  #Corpus is a large structured set of texts
4  #can be used to access the corpora in the NLTK data package
5  from nltk.corpus import stopwords
6
7  #Tokenization is a way to split text into tokens
8  #as paragraphs,sentences, words
9  from nltk.tokenize import word_tokenize, sent_tokenize
10
11 #Heap queue algorithm
12 #It's a special tree structure, each parent node is less than or equal to its child node
13 import heapq
14

```

spacy_summarization.py X

serving_static > spacy_summarization.py > text_summarizer

```

1  # NLP Packages
2  import spacy
3  nlp = spacy.load('en_core_web_sm')
4  # Packages for Normalizing Text
5  from spacy.lang.en.stop_words import STOP_WORDS
6  from string import punctuation
7  # Import Heapq for Finding the Top N Sentences
8  from heapq import nlargest
9

```

sumy_summarization.py X

serving_static > sumy_summarization.py > ...

```

1  #Module for automatic summarization of text documents and HTML pages
2  from sumy.parsers.plaintext import PlaintextParser
3  #Tokenizer
4  from sumy.nlp.tokenizers import Tokenizer
5  #LexRank is an unsupervised approach to text summarization based
6  #on graph-based centrality scoring of sentences. The main idea is
7  #that sentences "recommend" other similar sentences to the reader
8  from sumy.summarizers.lex_rank import LexRankSummarizer
9

```


The following code snippet indicates the Reading Time method and how to fetch the text from URL with the help of BeautifulSoup library

```
# Reading Time
def readingTime(mytext):
    total_words = len([ token.text for token in nlp(mytext)])
    estimatedTime = total_words/200.0
    return estimatedTime

# Fetch Text From Url
def get_text(url):
    page = urlopen(url)
    soup = BeautifulSoup(page,'lxml')
    fetched_text = ' '.join(map(lambda p:p.text,soup.find_all('p')))
    return fetched_text
```

Figure 2.6.1. 8: Reading Time & Fetch Text

The following code snippet indicates the main Summary generate function for copy & pasted text

```
@app.route('/analyze',methods=['GET','POST'])
def analyze():
    start = time.time()
    if request.method == 'POST':
        rawtext = request.form['rawtext']
        final_reading_time = readingTime(rawtext)
        final_summary = text_summarizer(rawtext)
        summary_reading_time = readingTime(final_summary)
        end = time.time()
        final_time = end-start
        text_word_count = []
        summary_word_count = []

        for i in rawtext:
            text_word_count.append(len(i.split()))

        for i in final_summary:
            summary_word_count.append(len(i.split()))

        b=len(text_word_count)
        print(b)

        a=len(summary_word_count)
        print(a)

        df = pd.DataFrame({
            "Charactor count in Text and Summary": ["Text", "Summary"],
            "Count": [b,a]})

        df.set_index("Charactor count in Text and Summary",drop=True,inplace=True)
        df.plot.bar()
        plt.show()

    return render_template('index.html',cstext=rawtext,final_summary=final_summary,final_time=final_time,
        final_reading_time=final_reading_time,summary_reading_time=summary_reading_time)
```

Figure 2.6.1. 9:Summary Generation Function for Custom Text

The following code snippet indicates the main Summary generate function for the extracted text from a single URL

```
@app.route('/analyze_url',methods=['GET','POST'])
def analyze_url():
    start = time.time()
    if request.method == 'POST':
        raw_url = request.form['raw_url']
        rawtext = get_text(raw_url)
        final_reading_time = readingTime(rawtext)
        final_summary = text_summarizer(rawtext)
        summary_reading_time = readingTime(final_summary)
        end = time.time()
        final_time = end-start
        text_word_count = []
        summary_word_count = []

        for i in rawtext:
            text_word_count.append(len(i.split()))

        for i in final_summary:
            summary_word_count.append(len(i.split()))

        b=len(text_word_count)
        print(b)

        a=len(summary_word_count)
        print(a)

        df = pd.DataFrame({
            "Character count in Text and Summary": ["Text", "Summary"],
            "Count": [b,a]})

        df.set_index("Character count in Text and Summary",drop=True,inplace=True)
        df.plot.bar()
        plt.show()
    return render_template('index.html',cstext=rawtext,final_summary=final_summary,final_time=final_time,
        final_reading_time=final_reading_time,summary_reading_time=summary_reading_time)
```

Figure 2.6.1. 10:Extract Text from Single URL

The following code snippet indicates the main Summary generate function for the extracted text from a multiple URL

```

@app.route('/analyze_multiple_url',methods=['GET','POST'])
def analyze_multiple_url():
    start = time.time()
    if request.method == 'POST':
        query = request.form['query']
        # print(query)

    # query = "'aloe vera'"
    query = urllib.parse.quote_plus(query) # Format into URL encoding
    number_result = 20

    ua = UserAgent()

    #google_url = "https://www.google.com/search?q=%27trade+war%27&num=20"
    google_url = "https://www.google.com/search?q=" + query + "&num=" + str(number_result)
    #print(google_url)
    response = requests.get(google_url, {"User-Agent": ua.random})
    soup = BeautifulSoup(response.text, "html.parser")
    export = str(soup)

```

Figure 2.6.1. 11:Extract Text from Multiple Sites

2.7.2. Testing

Testing phase is set to check the efficacy of the system and the final outputs, whether it satisfies the project requirements.

S = Set of words extracted by analyzing the sentences present in each document

There are many parameters against which you can evaluate your summarization system. like,

Precision = Number of important sentences / Total number of sentences summarized

Recall = Total number of important sentences Retrieved / Total number of important sentences present

Unit Testing

The purpose of executing unit testing is to make sure, all the independent components work as expected according to the work plan. The system should be divided into small components and tested in order to get a good quality report. The system can be divided into modules according to the design document. Finally, a verification process should

be run to check whether the system meets its specifications as discussed by the team at the beginning.

Test cases for the Arogya Application: Summary generation and Geographical Location Mapping

Table 2.6.2. 1:Test Case 1





Test case ID	1.1.	1.2.
Description	Login to the Application (If you have an account already)	
Input Data	<ul style="list-style-type: none"> • Username/E-mail • Password 	
Steps	<ol style="list-style-type: none"> 1. Select the Arogya icon from the app list 2. Wait for the Loading 3. Click the Sign In button 4. Type your E-mail and Password 5. Click on the Sign In button 	<ol style="list-style-type: none"> 1. Keep Password and E-mail empty 2. Click on the Sign In button
Expected Output	Successfully Redirect to the Dashboard	Validation Error: Required fields are empty
Actual Output	Successfully Redirect to the Dashboard	Validation Error: Required fields are empty
Status Pass -  Fail - 	Pass - 	Pass - 

Table 2.6.2. 2:Test Case 2





Test case ID	2.1.	2.2.
Description	Sign up to the Application (If you do not have an account already)	
Input Data	<ul style="list-style-type: none"> • Username/E-mail • Password 	
Steps	<ol style="list-style-type: none"> 1. Select the Arogya icon from the app list 2. Wait for the Loading 3. Click the Get Started button 4. Type your Name, E-mail and Password 5. Click on the Sign-Up button 	<ol style="list-style-type: none"> 1. Keep Name, Password and E-mail empty 2. Click on the Sign-Up button
Expected Output	Successfully Redirect to the Dashboard	Validation Error: Required fields are empty
Actual Output	Successfully Redirect to the Dashboard	Validation Error: Required fields are empty
Status Pass -  Fail - 	Pass - 	Pass - 

Table 2.6.2. 3::Test Case 3




Test case ID	3.
Description	Reset Password (If you forgot your password)
Input Data	<ul style="list-style-type: none"> • Username/E-mail
Steps	<ol style="list-style-type: none"> 1. Select the Arogya icon from the app list 2. Wait for the Loading 3. Click the Sign In button 4. Click the Forget Password link 5. Click on the Submit button
Expected Output	Receive an email with the password reset link
Actual Output	Receive an email with the password reset link
Status	
Pass - 	Pass - 
Fail - 	

Table 2.6.2. 4::Test Case 4

Test case ID	4.
Description	Add a New Location to Map (If the plant is classified as a Herbal One)
Input Data	<ul style="list-style-type: none"> • Geographical Latitude and Longitude coordinates • Plant Name




Steps	<ol style="list-style-type: none"> 1. Select the Arogya icon from the app list 2. Wait for the Loading 3. Click the Sign In button 4. Redirect to the Dashboard 5. Click on the Location Tracker button 6. Click the Plus Mark in the header 7. Enter the Plant Name 8. Click the add plant name into the Map
Expected Output	Location will be added to the Map view
Actual Output	Location is added to the Map view
Status Pass -  Fail - 	Pass - 

Table 2.6.2. 5:Test Case 5

Test case ID	5.
Description	View the Herbal Plant list (If the Login is successful)
Input Data	<ul style="list-style-type: none"> • No Input
Steps	<ol style="list-style-type: none"> 1. Select the Arogya icon from the app list 2. Wait for the Loading 3. Click the Sign In button 4. Redirect to the Dashboard 5. Click on the Top Right Corner Square button 6. You can view the List of Herbal plants
Expected Output	Display the Herbal Plants List view
Actual Output	Display the Herbal Plants List view

Status Pass -  Fail - 	Pass - 
---	--

Table 2.6.2. 6::Test Case 6




Test case ID	6.
Description	Add a New Herbal Plant as a post (If the Login is successful)
Input Data	<ul style="list-style-type: none"> Plant Name Plant Specialization
Steps	<ol style="list-style-type: none"> 1. Select the Arogya icon from the app list 2. Wait for the Loading 3. Click the Sign In button 4. Redirect to the Dashboard 5. Click on the Top Right Corner Square button 6. You can view the List of Herbal plants 7. Click the Plus Mark in the header 8. Enter the Plant Name 9. Click the Continue Button 10. Enter the Disease name it is specialized for 11. Click the Submit button
Expected Output	Display the newly added plat in the Herbal Plants List view
Actual Output	Display the newly added plat in the Herbal Plants List view
Status Pass -  Fail - 	Pass - 

Table 2.6.2. 7:Test Case 7




Test case ID	7.
Description	Edit a Plant from the List (If the Login is successful)
Input Data	<ul style="list-style-type: none"> No Input
Steps	<ol style="list-style-type: none"> 1. Select the Arogya icon from the app list 2. Wait for the Loading 3. Click the Sign In button 4. Redirect to the Dashboard 5. Click on the Top Right Corner Square button 6. You can view the List of Herbal Plants 7. Click on a record from the list 8. Change the values accordingly and click Update
Expected Output	Plant List will be updated with the new value
Actual Output	Plant List will be updated with the new value
Status	
Pass - 	Pass - 
Fail - 	

Table 2.6.2. 8:Test Case 8

Test case ID	8.
Description	Delete a Plant from the List (If the Login is successful)
Input Data	<ul style="list-style-type: none"> Select a plant




Steps	<ol style="list-style-type: none"> 1. Select the Arogya icon from the app list 2. Wait for the Loading 3. Click the Sign In button 4. Redirect to the Dashboard 5. Click on the Top Right Corner Square button 6. You can view the List of Herbal Plants 7. Click on a record from the list 8. Click Delete
Expected Output	Plant will be deleted from the Plant List
Actual Output	Plant is deleted from the Plant List
Status Pass -  Fail - 	Pass - 

Table 2.6.2. 9:Test Case 9

Test case ID	9.
Description	Compare the summary report on an herbal plant (If the Login is successful)
Input Data	<ul style="list-style-type: none"> • Input the Extracted text from the Internet
Steps	<ol style="list-style-type: none"> 1. Select the Arogya icon from the app list 2. Wait for the Loading 3. Click the Sign In button 4. Redirect to the Dashboard 5. Click on the Summarizer button 6. Click the left most Menu button 7. Copy and paste the text extracted from the Cinnamon Summarization 8. Click Summarize button




	9. Compare the Custom text area and Summary text area generated in each column
Expected Output	Summary will be generated
Actual Output	Summary is generated
Status Pass -  Fail - 	Pass - 

Table 2.6.2. 10:Test Case 10

Test case ID	10.
Description	Get a summary report on an herbal plant (If the Login is successful)
Input Data	<ul style="list-style-type: none"> Input the Extracted text from the Internet
Steps	<ol style="list-style-type: none"> Select the Arogya icon from the app list Wait for the Loading Click the Sign In button Redirect to the Dashboard Click on the Summarizer button Scroll down and click on the button Cinnamon Summarization Check the Custom text area and Summary text area
Expected Output	Summary will be generated
Actual Output	Summary is generated

Status	
Pass - 	Pass - 
Fail - 	

Integration Testing

Individual components are combined and tested into a single module to check the quality of the overall product and make sure, there's no other internal conflicts on the combination of whole product.

Object Detection component is followed by the Image classification component. Finally, the Information Summarizer comes into picture as it produces the detailed summary report on the classified herbal plant. Integration testing were done individually by the group members according to their component scope.

System Testing

Whole system is tested according to the specification, is said to be a System Integration. This makes sure, the system to be compatible with all the other modules and work as the execution plan without arising any issues. This is more like, black box testing type of testing. If there's any bugs or issues, the overall team has to take the responsibility and relaunch the application with the needed modifications

User Acceptance Testing


This is the phase, where the customer interacts with the developed trial product. The product will be tested by the customer to make sure whether it reaches the customer functional requirements. They will be given a demo version of the product, so that, they can test it. If the user accepts the products without making any issues and satisfied, that is the end of the testing life cycle. Otherwise, the team has to do some other modifications accordingly, and the cycle goes on until the user gets satisfied.

3.0. RESULTS & DISCUSSION

3.1. Results – Actual Interfaces

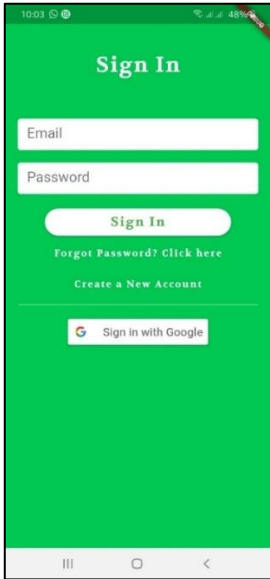
3.1.1. Main Dashboard View

Table 3.1.1. 1:Interface 1

Interface	Functionality
	<ul style="list-style-type: none"> • Main entry point to the Application-Main UI • Get Started button direct to the Sign-Up page • You get a chance to register into the system over there • Sign In button direct to the Sign In page • By providing the credentials, can login to the account

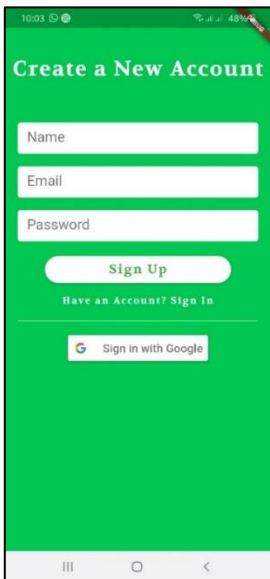
3.1.2. User Login

Table 3.1.2. 1:Interface 2

Interface	Functionality
	<ul style="list-style-type: none"> • By providing the credentials, can login to the account • If you click Sign in with Google, you will have a chance of login with the google account credentials


3.1.3. User Registration

Table 3.1.3. 1:Interface 3

Interface	Functionality
	<ul style="list-style-type: none"> • Sign Up to the system by providing Name, E-mail and Password • And also, you can use Google Sign Up method too

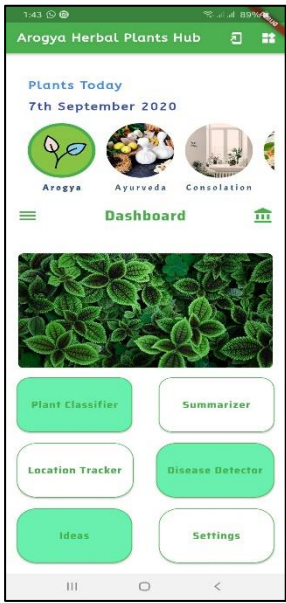
3.1.4. User Password Reset

Table 3.1.4. 1:Interface 4

Interface	Functionality
	<ul style="list-style-type: none"> If you forgot your Password, provide your e-mail address, so that you can receive a reset link

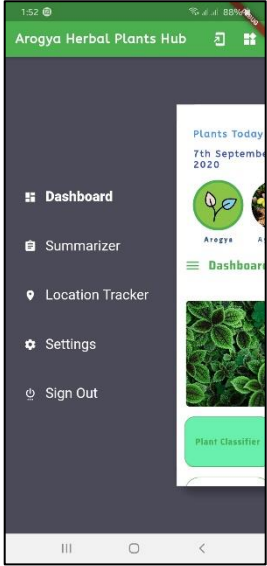
3.1.5. Dashboard

Table 3.1.5. 1:Interface 5

Interface	Functionality
	<ul style="list-style-type: none"> This is the Dashboard You can access every main menu from this board

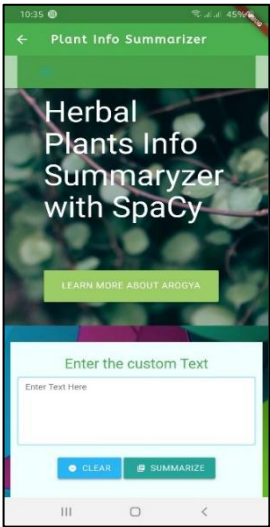
3.1.6. Side Bar Menu

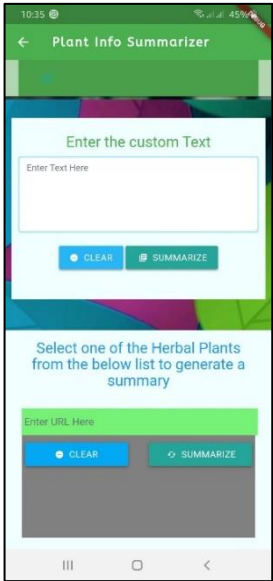
Table 3.1.6. 1:Interface 6


Interface	Functionality
 <p>The screenshot shows the 'Arogya Herbal Plants Hub' app. On the left, there is a dark grey side bar menu with white text and icons for 'Dashboard', 'Summarizer', 'Location Tracker', 'Settings', and 'Sign Out'. The main content area on the right is white and displays 'Plants Today 7th September 2020' with a green circular icon, a 'Dashboard' button, a plant image, and a 'Plant Classifier' button.</p>	<ul style="list-style-type: none"> • This is the side bar menu • Main menus are located in this view too


3.1.7. Herbal Plants Info Summarizer

Table 3.1.7. 1:Interface 7

Interface	Functionality
 <p>The screenshot shows the 'Plant Info Summarizer' app. It has a green header with a back arrow and the title 'Plant Info Summarizer'. Below the header, there is a large green area with the text 'Herbal Plants Info Summarizer with SpaCy' and a 'LEARN MORE ABOUT AROGYA' button. At the bottom, there is a white box with the text 'Enter the custom Text' and a text input field. Below the input field are two buttons: 'CLEAR' and 'SUMMARIZE'.</p>	<ul style="list-style-type: none"> • This is the Summarizer view • User has to enter a customer text to generate a summary

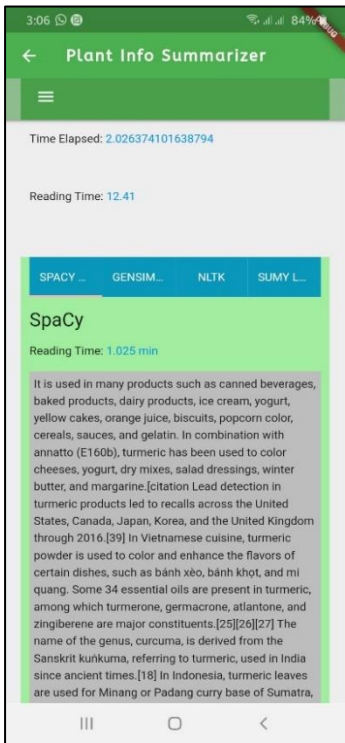
Interface	Functionality
	<ul style="list-style-type: none"> • User has to paste the required websites' URL to be summarized • Then it will extract the content from that website and generate a summary

Interface	Functionality
	<ul style="list-style-type: none"> • 6 types of herbal plants are already listed in the view • If you want, you can select one of the plants from the list menu • Then the system will generate a summary on it

Interface	Functionality
	<ul style="list-style-type: none"> • Search for the plant by its name • Then it will extract the information from multiple websites • Finally, the system will generate a summary on it

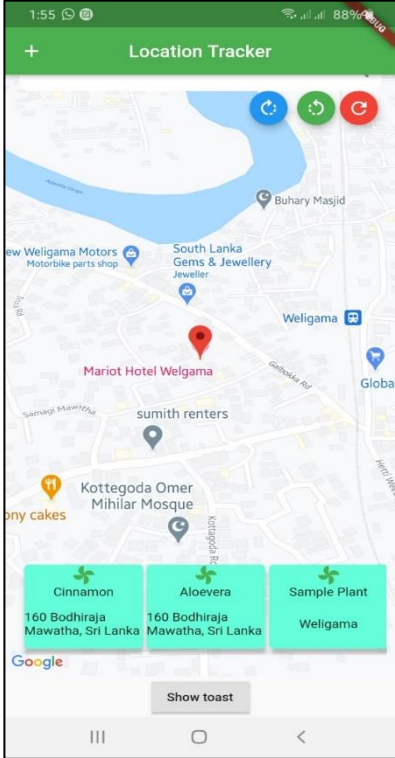
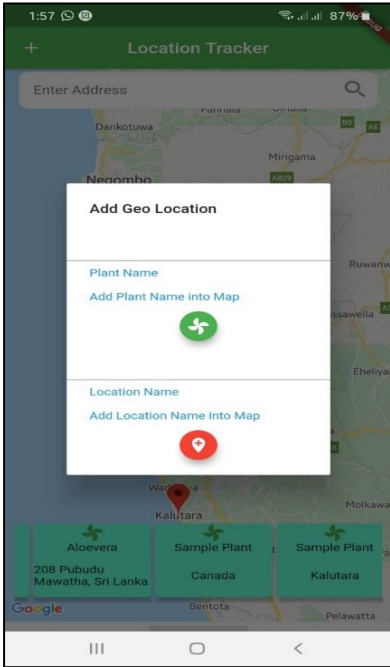
3.1.8. Summary Comparison

Table 3.1.8. 1:Interface 8

Interface	Functionality
	<ul style="list-style-type: none"> • Generated summary is compared with four other models • So that, the user can obtain the most accurate results comparatively

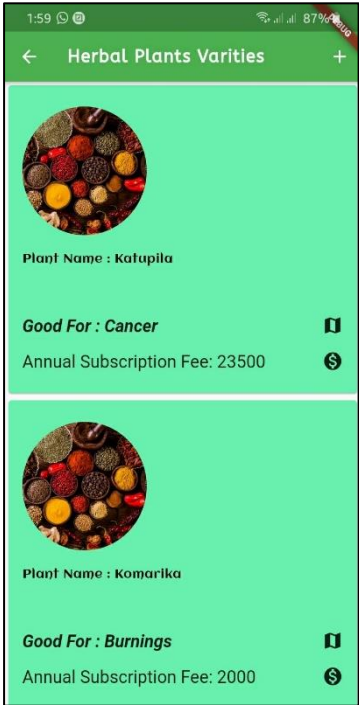
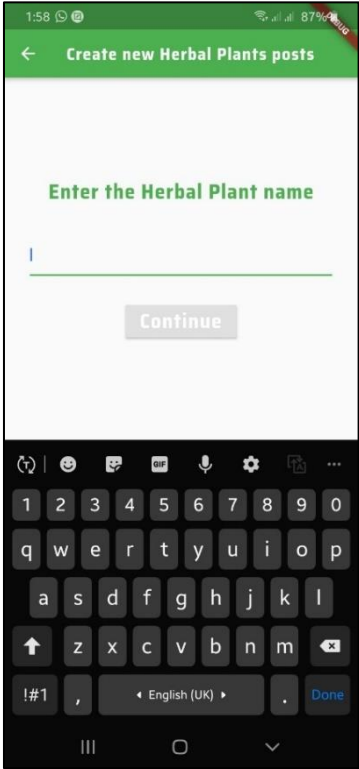
3.1.9. Location Tracker

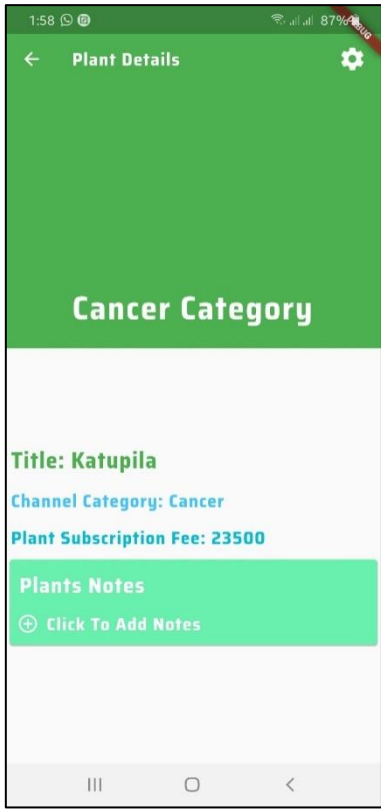
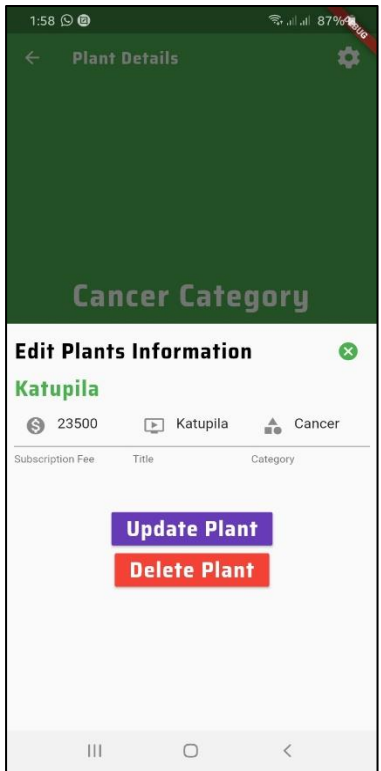
Table 3.1.9. 1:Interface 9

Interface	Functionality
	<ul style="list-style-type: none"> • Location View as a map • Same herbal plant can be grown in different locations • Many herbal plants can be grown in a one location
	<ul style="list-style-type: none"> • Track the current location • Add Plants into the Map view

3.1.10. Insert Herbal Plant details

Table 3.1.10. 1:Interface 10

	<ul style="list-style-type: none"> • Insert required herbal plants in to the system
	<ul style="list-style-type: none"> • Enter the plant details to add into the system

	<ul style="list-style-type: none"> • More Information on the specific plant
	<ul style="list-style-type: none"> • Update herbal plant and Delete herbal plant

3.1.11. Summary Results

The following graphs depict clear evidence to prove the experiment results that the Seq2Seq LSTM model obtained the highest accuracy. The graphs indicate the word count comparison between, the generated summary and the customized text which was extracted from the multiple web pages dynamically.

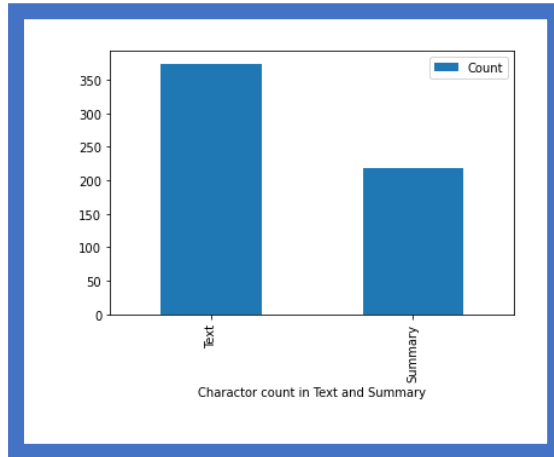


Figure 3.1.11. 1: Instance 1 - Word count of Generated Summary in comparison to Customized text

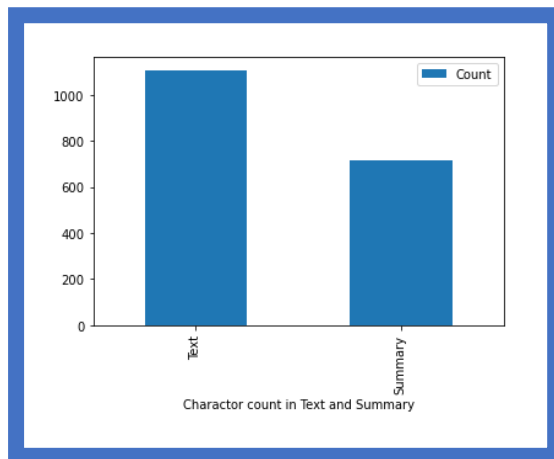


Figure 3.1.11. 2: Instance 2 - Word count of Generated Summary in comparison to Customized text

3.2. Research Findings

Our final solution **Arogya** is a mobile application, which includes all four components in a single point. It handles Object detection, Object Classification, Information Summarization and Geographical Location Mapping.

- Abstractive Text Summarization

The main research finding is based on Generating Summaries of the Herbal plants, which have been classified in the early stage by the object classifier. Information has been extracted from the internet and processed in order to get a productive output. Arogya is capable of preparing dynamic summaries on multiple web pages. So, to do that, information has been extracted from the first three top ranked web pages and combined all of them together by the system. Then starts the internal process of cleaning the content by removing all the unnecessary duplicates and passing back to the summarizer component to handle the rest of the process.

The task which was assigned was not easy. It was somewhat difficult to enter into the main flow of the function.



Figure 3.2. 1: Abstractive Text Summarization

- Can search for a random herbal plant

Apart from generating summaries on classified herbal plants, in Arogya there is an option for searching random herbal plants according to the user's preference.

User has to type the plant name and search for it. Then the system will be generated a summary report on it as well

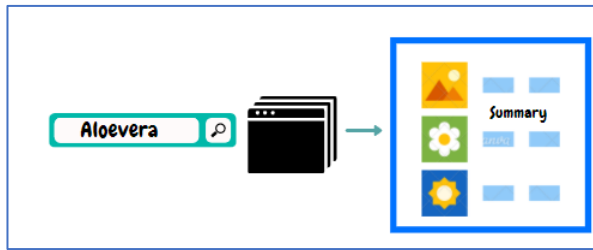


Figure 3.2. 2: Search for Random Plants

- Summary Comparison with Multiple models

And also, there is a function implemented to compare the generated summaries with four main models SpaCy, NLTK, Sumy, and Gensim. So that user would get a chance to choose the best summary document which they would prefer to go with as a solution for their prevailing requirement, based on Herbal plants information.

- Ayurvedic Plants' Location tracking and mapping with GIS technology

The Geographical Information System, which we have implemented in the proposed system Arogya, mainly focuses on the locations of the Ayurvedic plants all over Sri Lanka. Whenever a user identifies an ayurvedic plant, the system is capable of tracking the current location and mapping it to the Map view. And also, they can search for a particular ayurvedic plant and find the areas which grow an abundance.

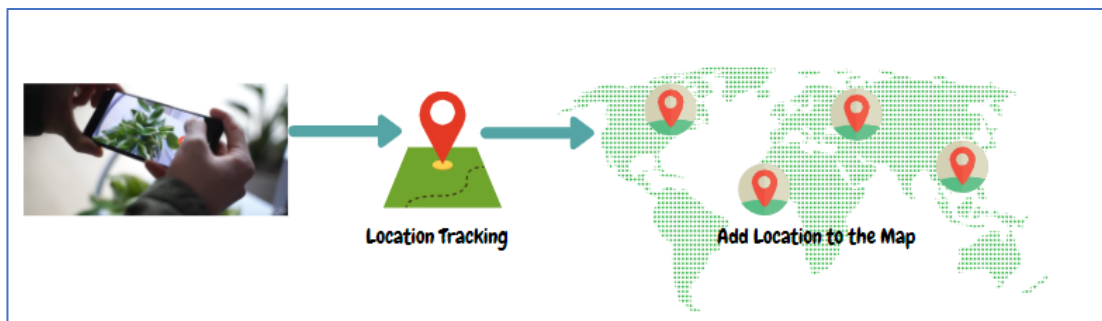


Figure 3.2. 3: Location Tracking and GIS Technology

- User Account Login and Registration

The Mobile Application is well organized with a highly secured authentication system. So, the customers' data will be secured. The system is occupied with the Firebase Backend. Since the system is implemented with multiple user-level authentications, every user of the system can maintain their information separately.

There are three main **Login** Options available in the mobile application.

1. Regular user Login with Email and Passwords

It is just the normal procedure user has to follow in order to login to the system. First of all, user has to Sign up in to the system by providing necessary information, email, and password as the credentials, then will be prompted to Login Interface.

2. Using Google - Gmail Login credentials

Here, user will be able to create a new account by using their existing google Gmail account credentials.

3. Using Anonymous Login option

Anonymous user login is especially available for random users, who do not want to maintain separate accounts for themselves. They just login to the system as outsiders and get the provided service by the system.

As an example,

Herbal Plants Classification is the main service provided by Arogya

So, the anonymous user login to the system and just get classified some random plants whether it is an herbal plant or not.

The user doesn't want to keep records after adding them to the database.

They need only to identify the random plant, that's it.

- Maintaining the Herbal Plants records in the system

If the classified plant is an Ayurvedic plant, and if it is very helpful for the user, they can add it to their account and keep a record of it. Later they can update the record or delete if it is not needed further.

1. Insert Herbal Plants
2. Update Plants
3. Delete Plants
4. Display Plants as a list view

3.3. Discussion

My research component, Summary Generation on Herbal Plants and Geographical Location Mapping was started by analyzing the existing systems which were related to my component and it was not an easy path to get here from the beginning. Several algorithms were studied and evaluated in order to get the best results. Sometimes, had to spend more time on testing the product quality to get the best results rather than concentrating on further implementation. As the first step, I had to prepare an effective dataset to train my model. So that, website information should be scrapped. At the beginning, even though I was able to scrape information from a single webpage at once, was not able to extract information from multiple websites with the help of python libraries. Even, there were many ups and downs throughout the project life span, finally I was able to manage the project scope somehow.

Sometimes, to provide an ideal unique solution for its consumers, I had to do many great commitments. The application has been successfully completed along with covering its main targets and objectives. Finally, it was able to fulfil the consumers' requirements on Ayurvedic Plants Information Summarization and find the locations of the specific plants. The Seq2Seq model well played the role in the project with the support of two other models Encoder and Decoder.

Future Work

I expect to expand this application according to some other consumer's needs too. So that, will be able to increase the usability and productivity of this Application for further extent. And also, it is required to increase the accuracy of the summaries and to keep the content more concise to the point. Current model Seq2Seq is capable of predicting the next word, but sometimes the accuracy is less, because of the size of the dataset. Hence, if it is possible to collect more data as the training dataset on this project, it will direct this project into a successful direction with more accurate results.

4.0. THE SUMMARY OF MY CONTRIBUTION

Table 4. 1: Summary of My Contribution

Functionality	Description
Summary Generation on Ayurvedic Herbal Plants	<ul style="list-style-type: none"> • Collect and prepare the dataset for training the model • Single webpage extraction and multiple webpage extraction • Cleaning the dataset by removing the redundant values • Summary comparison with three other models
Geographical Location Mapping	<ul style="list-style-type: none"> • Using google Map API to Live Location tracking • If the found plant is classified as an herbal plant, Add the current Location address for the Map as it is the plant found location.
User Login & Registration	<ul style="list-style-type: none"> • Implementation of User Levels Authentication with Firebase • Multiple ways of Login methods
User Profile Maintenance	<ul style="list-style-type: none"> • Create, Edit, Delete, and View functions implementation on ayurvedic plants

5.0. CONCLUSION

Our product mainly targets the Ayurvedic Herbal Industry in Sri Lanka. Since it is ayurvedic doctors' oriented for long time of period, we had a plan of expanding the ayurvedic industry towards the general public. So that, we can share the herbal related knowledge with the people who do not have much experience with the Ayurveda treatment. The plan was to spread the knowledge such as, herbal plants categories, herbal recipes can be prepared from them, and provide a quick snapshot of the summary of the classified plant. Summary will be generated from the information, which extracted from the highly ranked multiple websites. Hence, the user never required to have a doubt on the accuracy of the plant information. Rather than going with a static database information on herbal plants, it is really worth to work with the dynamic information extracted from top websites as the information is updated daily. In order to provide an effective solution for the consumers, data should be real time and accurate, so if there is any false data available in the web site, owners of that site will update the site on daily basis. That is the main idea to use a dynamic dataset from the websites.

Long sequence of words has been extracted from the web and finally output a short summary of sequence of words. This has been modeled as Many-to-Many Seq2Seq problem, with the help of Deep Learning. The Seq2Seq model has been built with two other main models called as, Encoder and Decoder. In this research Encoder and the Decoder are considered to be Long Short-Term Memory (LSTM). Encoder and the Decoder have been set up in two phases such as Training phase and Inference phase. The model has been trained on dataset and predicted the target sequence offset by one-time stamp. There is an option for Summary generation with multiple models. So that, users will have a chance of comparing the generated summaries and go for the best summary. Final Reading time for the extracted text from the website and summarized text is displayed in the system. It indicates the time comparison. Ayurvedic physicians, College students, Research students and General public will be the targeted consumers for this application.

REFERENCES

- [1] H. P. Edmundson, “New Methods in Automatic Extracting,” *J. ACM*, 1969, doi: 10.1145/321510.321519.
- [2] J. Kupiec, J. Pedersen, and F. Chen, “Trainable document summarizer,” 1995, doi: 10.1145/215206.215333.
- [3] N. Ramanujam and M. Kaliappan, “An automatic multidocument text summarization approach based on naïve Bayesian classifier using timestamp strategy,” *Sci. World J.*, 2016, doi: 10.1155/2016/1784827.
- [4] A. Leoncini, F. Sangiacomo, P. Gastaldo, and R. Zunino, “A Semantic-Based Framework for Summarization and Page Segmentation in Web Mining,” in *Theory and Applications for Advanced Text Mining*, 2012.
- [5] M. Allahyari *et al.*, “Text Summarization Techniques: A Brief Survey,” *Int. J. Adv. Comput. Sci. Appl.*, 2017, doi: 10.14569/ijacsa.2017.081052.
- [6] S. S. Lakshmi and M. U. Rani, “Multi-Document Text Summarization Using Deep Learning Algorithm with Fuzzy Logic,” *SSRN Electron. J.*, 2018, doi: 10.2139/ssrn.3165331.
- [7] A. K. Singh and M. Shashi, “Deep Learning Architecture for Multi-Document Summarization as a cascade of Abstractive and Extractive Summarization approaches,” *Int. J. Comput. Sci. Eng.*, 2019, doi: 10.26438/ijcse/v7i3.950954.
- [8] Y. M. Shaalan and A. Rafea, “Frequently asked questions web pages automatic text summarization,” 2011, doi: 10.2316/P.2011.717-095.
- [9] Y. Zhang, N. Zincir-Heywood, and E. Milios, “World Wide Web site summarization,” *Web Intelligence and Agent Systems*. 2004.
- [10] N. Kumar *et al.*, “Leafsnap: A computer vision system for automatic plant species identification,” 2012, doi: 10.1007/978-3-642-33709-3_36.
- [11] S. A. Babar and P. D. Patil, “Improving performance of text summarization,” 2015, doi: 10.1016/j.procs.2015.02.031.
- [12] S. P. Singh, A. Kumar, A. Mangal, and S. Singhal, “Bilingual automatic text summarization using unsupervised deep learning,” 2016, doi: 10.1109/ICEEOT.2016.7754874.
- [13] H. A. Chopade and M. Narvekar, “Hybrid auto text summarization using deep neural network and fuzzy logic system,” 2018, doi: 10.1109/ICICI.2017.8365192.
- [14] Y. R. Azeez and C. Rajapakse, “An Application of Transfer Learning Techniques in Identifying Herbal Plants in Sri Lanka,” 2019, doi: 10.23919/SCSE.2019.8842681.
- [15] R. G. De Luna *et al.*, “Identification of philippine herbal medicine plant leaf

using artificial neural network,” 2017, doi: 10.1109/HNICEM.2017.8269470.

- [16] H. Kaya, I. Keklik, T. Ensari, F. Alkan, and Y. Biricik, “Oak leaf classification: An analysis of features and classifiers,” 2019, doi: 10.1109/EBBT.2019.8742053.

GLOSSARY

Term	Definition
Deep Learning	Artificial Intelligence function that imitates human activities to process data and creating patterns for use in decision making
NLP	Natural Language Processing, is a way of conveying information and meaning with semantic cues using Artificial Intelligence
API	Application Programming Interface, is a software intermediary that allows two applications to talk to each other
LSTM	Long Short-Term Memory, is a Artificial Recurrent Neural Network (RNN) Architecture used in Deep Learning
IE	Information Extraction, extracts information from websites
GIS	Geographical Information System

APPENDICES

1. Use Case Diagram

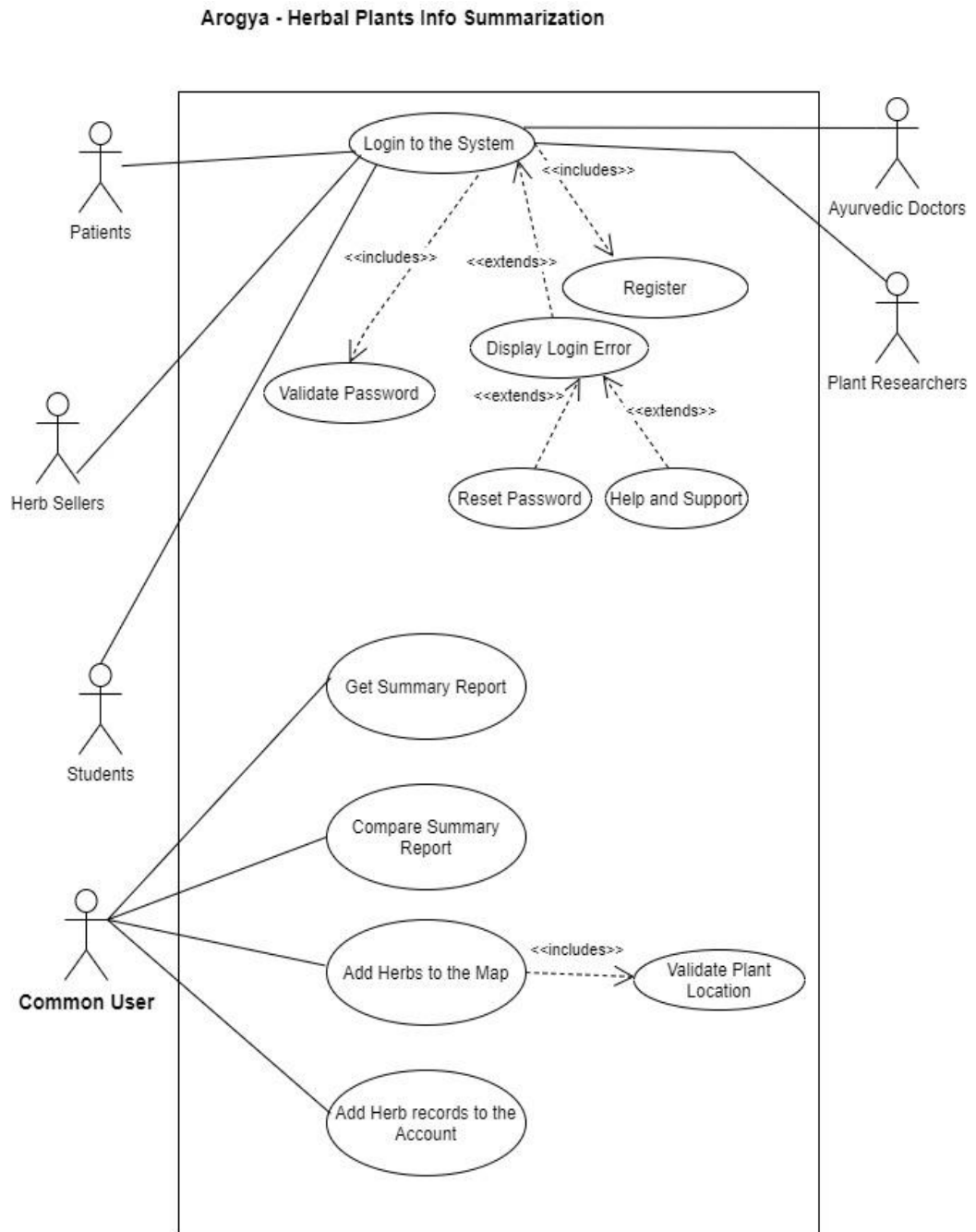


Figure 1:Use Case Diagram

2. Class Diagram

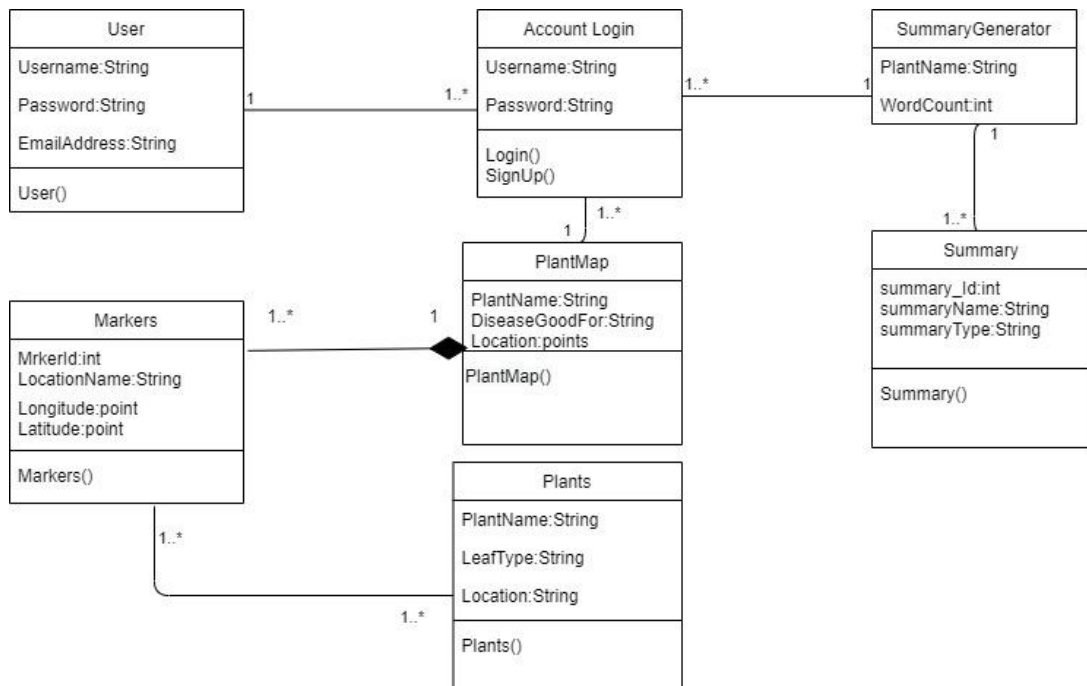


Figure 2:Class Diagram