CSE 4022 - Natural Language Processing

Project Report

Voice Based Digital Prescription

Team Members:

Ramprasath A - 17BCE1098
Aravind M - 17BCE1215
Srinath K R - 17BCE1217

Slot: TDD1 **Faculty:** Prof. Sridhar R

Project Demo Link: Voice Based Digital Prescription Demo.webm

Research paper taken as reference:

Mahatpure, J., Motwani, M., & Shukla, P. K. An Electronic Prescription System powered by Speech Recognition, Natural Language Processing and Blockchain Technology.

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

Aim

We propose a system that automates the process of writing medical prescriptions by recording the doctors' dictation, extracting the necessary parameters like symptoms, medicines, dosage, general medical advice etc. and sending a copy of the prescription to the patient through an email.

Objective

The proposed system, when adopted, would help in eliminating errors occurring due to indecipherable handwriting, drug interactions and confusing drug names as the prescription is digitized which would make it easy for pharmacists to read and give medicines to the patient. Also confusing medicine names can be resolved by mapping them to a database and assigning a unique code to each medicine. The code would resolve conflicts in drug names. This also offers comfort to the patient as it solves the problem of a patient losing their prescription as a copy of it will be emailed to them. Also, the doctor would have a copy of the patient's prescriptions from the previously sent emails. So, the patient need not carry the older records anymore.

Motivation

Studies[1] show that there have been a lot of medication errors due to pharmacists misinterpreting the doctors' prescription due to indecipherable handwriting, confusing drug names etc. As a result, the patients are given wrong medicines. This has led to serious health consequences and has even resulted in deaths.

A patient has to keep his/her prescription carefully and losing it means they'll have to visit the doctor again. Even in that case, the doctor might not exactly know what medicines and corresponding dosage was given to the patient in the previous visit and would end up giving a different medicine and/or dosage. The patient might have to buy the extra medicines too.

Also, the patient has to carry the prescription every time during consecutive visits so that the doctor could fully understand the entire medication procedure and the medicines given at each visit and how the patient has progressed.

Literature Survey

In voice-based mobile prescription application for healthcare services (VBMOPA), the authors have proposed a solution which is to develop a voice-based mobile prescription application (VBMOPA) that offers an alternative e-health platform to complement the existing mobile healthcare delivery system. Voice based mobile prescription system is a system of healthcare delivering services that can take place anytime, anywhere with the help of a mobile phone by dialing a telephone number that connects users to an e-prescription application that is present in a web server. [2]

In python tool for extracting and analysing semi-structured information from medical records, the authors have proposed a solution to the problem of extracting, storing, retrieving, and analysing information from health records, with respect to the Indian healthcare in the form of a Python-based tool, Healthcare Data Extraction and Analysis (HEDEA) which has been designed to extract structured information from various medical records using a regular expression-based approach. [3]

In information extraction from unstructured electronic health records and integration into a data warehouse, the authors have proposed an approach to extract information from medical texts and a workflow to integrate this information into a clinical data warehouse. The presented information extraction technique is based on Conditional Random Fields (CRF) and keyword matching with terminology-based disambiguation. [4]

In design and implementation of a voice-based medical alert system for medication adherence, the authors have proposed a voice-based mobile medical alert system (Voice MedAlert) for outpatient adherence. The adoption of voice-based applications could greatly reduce non-adherence of patients to treatment regimen because they allow appointments and prescription information to be captured and heard through voice response rather than in the physician’s handwriting. [5]

Proposed Methodology

The working of the application is as follows:

● Since no standard datasets containing symptoms, diseases and medicines were available, we created our own dataset through the process of web scraping. The dataset for list of symptoms, diseases and common medicines are scraped from Wikipedia[6], RxList[7] and Medscape[8].

● The doctor dictates the prescription and the application records it. Google Web Speech API[9] is used for the Speech-to-Text conversion. It helps to convert audio to text by applying powerful neural network models in an easy-to-use API. The Web Speech API aims to enable web developers to provide, in a web browser, speech-input and text-to-speech output features that are typically not available when using standard speech-recognition or screen-reader software. The API is designed to enable both brief (one-shot) speech input and continuous speech input. Speech recognition results are provided to the web page as a list of hypotheses, along with other relevant information for each hypothesis.

● The doctor then makes changes, if any, to the transcript (like the dosage, duration the medicine has to be taken or any other changes) and submits the transcript for extraction along with the patient ID.

● Then the extraction process happens. With the help of the patient ID received, the name and age of the patient are extracted from the database.

● Then the transcript is tokenized and Parts-of-Speech (POS) tagging is done. This helps in identifying different parts of speech like common nouns, proper nouns, adjectives, adverbs etc.

● Then by identifying the keyword ”symptoms” , we scan the transcript around the word transcript looking for nouns, which are the names of the diseases we require. The identified nouns are then verified and validated with the list of symptoms available in the database. If a match is found, then that symptom is added to the list of symptoms of the patient.

● Similarly the process is repeated to identify the diagnosed disease. The system looks for the presence of words like ‘diagnosed with’, ‘suffering from’ and other similar words.

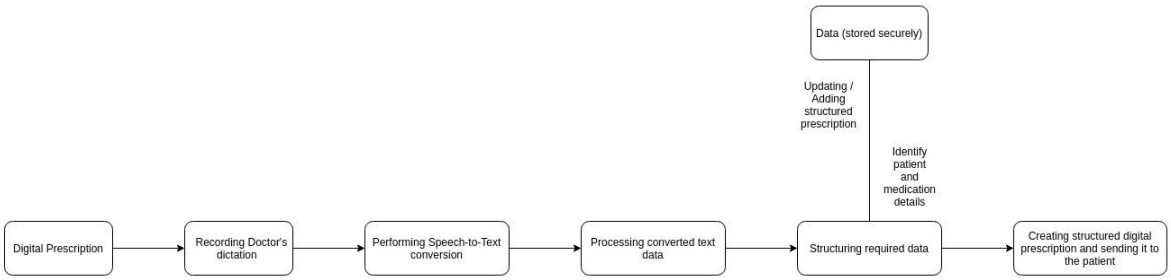
● For extraction of medicines, 4 parameters are needed - the name of the medicine, the dosage (in case of tablets), for how many days it needs to be taken, the time of the day to be taken (morning/afternoon/evening/night) and whether it should be taken before or after food. General

advice (like not to eat spicy food, not to eat food outside, not to eat cold food, etc.) which the doctor gives is also extracted.

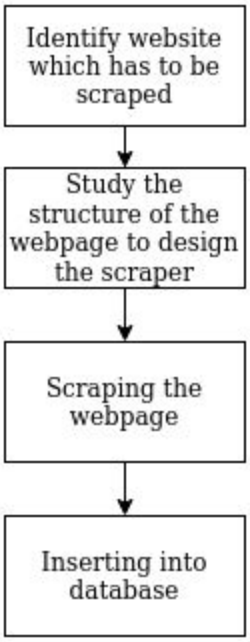
● The extracted information is then presented to the doctor for confirmation and if any changes are to be made, the doctor makes it and then authorises it. The prescription is then sent to the patient through an email which is obtained using the patient's ID.

Block diagrams

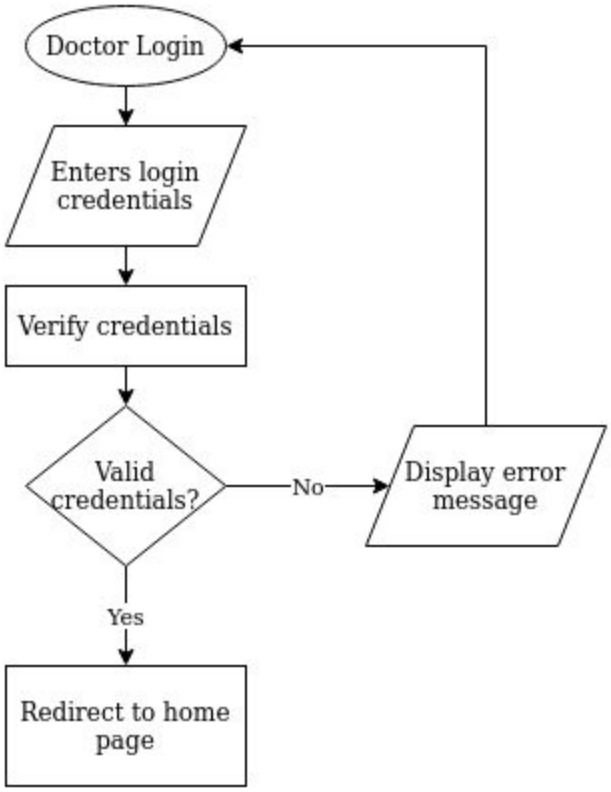
1. Overall diagram



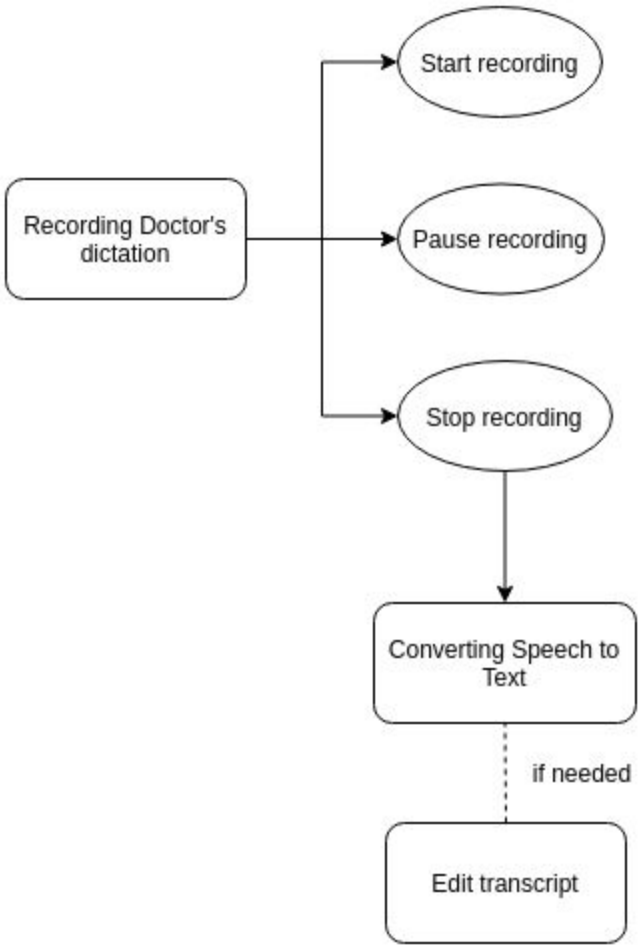
2. Scraping Web to create Dataset



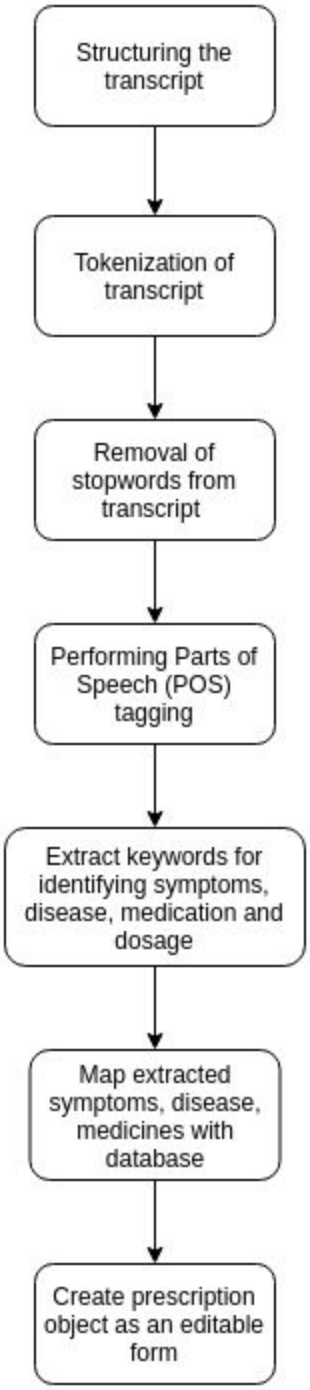
3. **Login**


Doctor Login
Enters login credentials
Verify credentials
Valid credentials?
No
Display error message
Yes
Redirect to home page

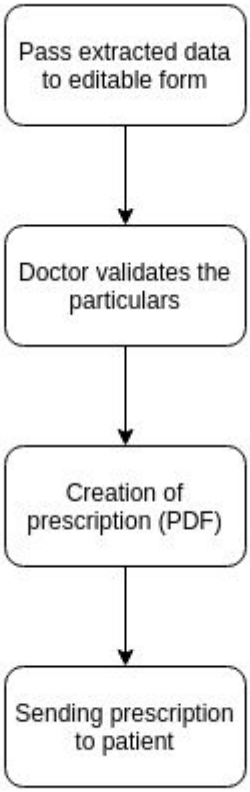
4. **Recording Doctor’s dictation and conversion of speech to text**



5. Structuring transcript and extracting information



6. **Doctor validation and sending email of the prescription to the patient**



Implementation

Tech Stack used: HTML, CSS, Bootstrap, Javascript, JQuery, Django, MySQL.

1. Scraping Data

Since there is no standard dataset available for the list of Symptoms, Diseases and Medicines, it was scraped from 3 different websites. We built a web scraper to create the dataset. 'BeautifulSoup' package was used to parse the webpage. The symptoms, diseases and medicines were identified and we put them into different tables in the database.

2. Login

This module is for the doctors to login to the portal after which they start recording the prescription. The doctors enter their login credentials into a web form and the details are sent to the server for validation. The server then validates the credentials from the database and the doctors are redirected to the homepage on successful authorisation.

3. Recording Doctor's dictation and conversion of speech to text

In the homepage there is a button which when clicked starts recording the dictation. We use Google Web Speech API that dynamically converts the dictation to text and it appears in a text box. The doctor can pause his dictation anywhere in between and the resume after a small pause. After the doctor completes dictation, the entire dictation gets converted to a transcript and would appear in a text box. The doctor can make a few changes, if there are any errors in the transcript. Then, the doctor enters the Patient ID. The Patient ID is an unique ID assigned to each person and it is used for accessing documents related to that patient easily. The doctor then submits the transcript for processing.

4. Structuring transcript and extracting information

This process is done on the server. Once it receives the transcript from the doctor's browser, structuring of the transcript happens. We make use of NLTK (Natural Language Processing ToolKit)[6] here. The first step in this process is tokenization. In this step, the transcript is split into individual tokens (words). Then the stop words in the tokenized list, like 'is', 'a', 'as', etc are removed. The Parts-of-Speech (POS) Tagging is done. It tags each word into a Common Noun, Proper Noun, Adjective, Adverb, Verb etc.

Then keywords are identified for extracting the parameters. For example, to identify symptoms, we search around the keywords like 'having symptoms', 'has the following symptoms' etc. We take the nouns around it and map each one of them with the symptoms table in the database. The matches that are found from the database are then assigned to the symptoms list. Similarly the disease is identified by looking for keywords like 'is suffering from', 'is diagnosed with' etc. For Medicines, we also need to identify the dosage, time of the day when it is to be taken (morning/afternoon/evening/night) and whether it is to be taken before/after food. Here we search around a particular medicine for identifying them.

After identifying all the parameters, a prescription object is created and send to the browser for the doctor to validate

5. Validating prescription and sending email to the patient

Once the browser receives the prescription object, it is parsed by the browser (using JavaScript) and the prescription is displayed to the doctor. The doctor looks for errors and corrects them if there are any. Then the doctor approves the prescription and it is ready to be emailed to the patient

For sending email, we make use of Django's email facility. Using the Patient ID, the email of the patient is identified from the database. The prescription is then sent to the patient's email.

Sample Code

1. **Using Google Web Speech API**

#Start Recording Dictation

This function starts recording the doctor's dictation

```
function StartRecording() {
    if (recognizing) {
        recognition.stop();
        if (final_text1.innerHTML == "" && interim_span.innerHTML == "")
        final_text1.innerHTML = final_span.innerHTML;
        else if (final_text1.innerHTML == "" && final_span.innerHTML == "")
        final_text1.innerHTML = interim_span.innerHTML;
        else if (final_text1.innerHTML != "" && final_span.innerHTML == "")
        final_text1.innerHTML += " " + interim_span.innerHTML;
        else
        final_text1.innerHTML += " " + final_span.innerHTML;
        reset();
    } else {
```

```
            final_span.innerHTML = "";
            recognition.start();
            recognizing = true;
            document.getElementById("start").removeAttribute("class");
            document.getElementById("start").setAttribute("class", "far fa-pause-circle");
            final_span.innerHTML = "";
            interim_span.innerHTML = "";
            }
        }
```

#Stop Recording dictation

This function stops recording the dictation

```
        function StopRecording() {
        if (document.getElementById("start").getAttribute("class") == "far fa-play-circle") {
            final_text2.innerHTML = final_text1.innerHTML;
        } else {
            reset();
            final_text2.innerHTML += final_text1.innerHTML + " " + final_span.innerHTML;
        }
        $("#final_upload").show();
        document.getElementById("start").setAttribute("class", "far fa-play-circle");
        }
```

#Show Recording

This function is to view the transcript of the recorded dictation

```
function ShowRecording() {
$("#modal-body").fadeIn(500);
$("#initiate-recording").fadeOut(500);
}
```

2. Structuring the Transcript to a prescription

Tokenize the transcript

```
sample_list = word_tokenize(sample)
```

Mapping the medicines from the Database

```
for i in range(ind, len(sentence)):
    mlist = Medicine.objects.filter(name__iexact=sentence[i])
```

Identifying Dosage, when the medicine has to be taken, whether it is to be taken before/after food and duration

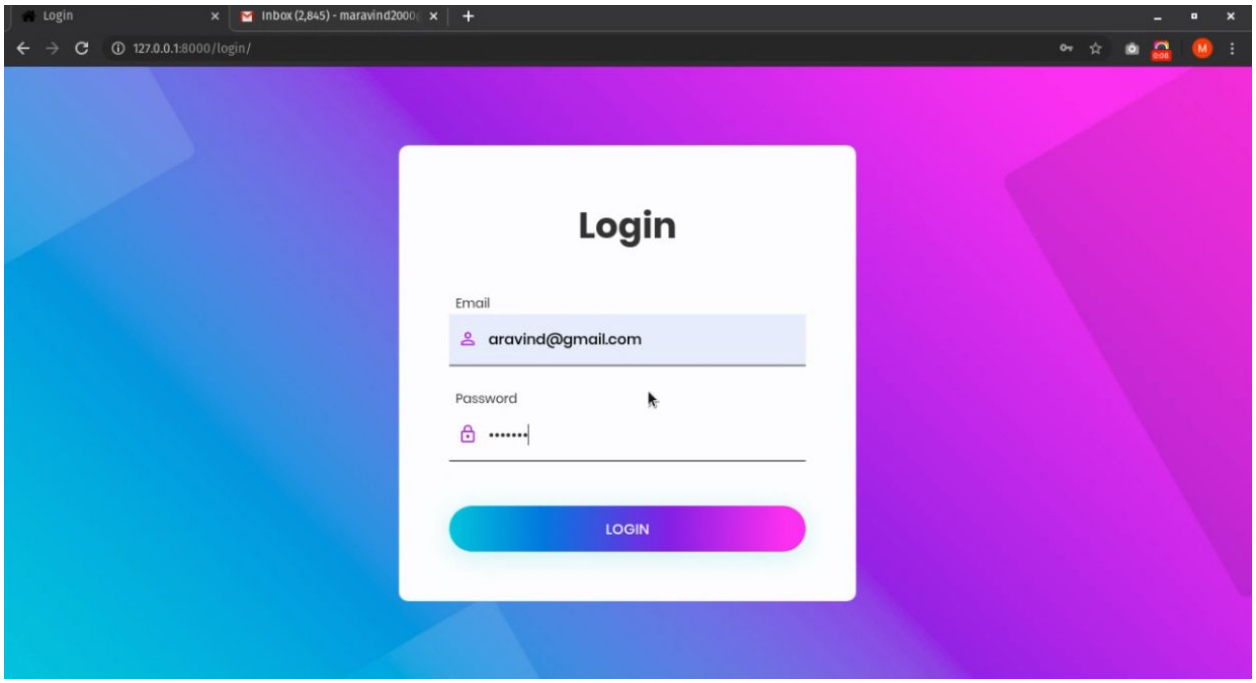
```
if len(mlist) == 1 and POS[i] == "NNP":
    if sentence[i-1].lower() == "tablet":
        med.append(sentence[i-1][0].upper()+". " +
        mlist[0].name+" "+sentence[i+1]+"mg")
    else:
        med.append(sentence[i-1][0].upper()+". "+mlist[0].name)

if sentence[i].isdigit() and sentence[i+1].lower() == "days" and POS[i] == "CD":
    duration.append(sentence[i]+" "+sentence[i+1])
```

```
    for i in range(len(med)):
        ind = sample_list.index(med[i].split(' ')[1])
        ind_daynight = ind

        foundDayNight = False
        while foundDayNight == False and ind_daynight <= len(sample_list):
        if sample_list[ind_daynight].lower() in ["night", "afternoon", "evening", "morning"]:
            foundDayNight = True
            day_night.append(sample_list[ind_daynight])
        ind_daynight += 1

    foundBeforeAfter = False
    ind_beforeafter = ind

    while foundBeforeAfter == False and ind_beforeafter <= len(sample_list):
    if sample_list[ind_beforeafter].lower() in ["before", "after"]:
        foundBeforeAfter = True
        before_after.append(sample_list[ind_beforeafter]+" food")
    ind_beforeafter += 1

    med_data = []
    d = {"At": "", "Medicine_Name": "", "Taken": "", "Duration": ""}
    for i in range(len(med)):
        d["Medicine_Name"] = med[i]
        d["At"] = day_night[i]
        d["Taken"] = before_after[i]
        d["Duration"] = duration[i]
```

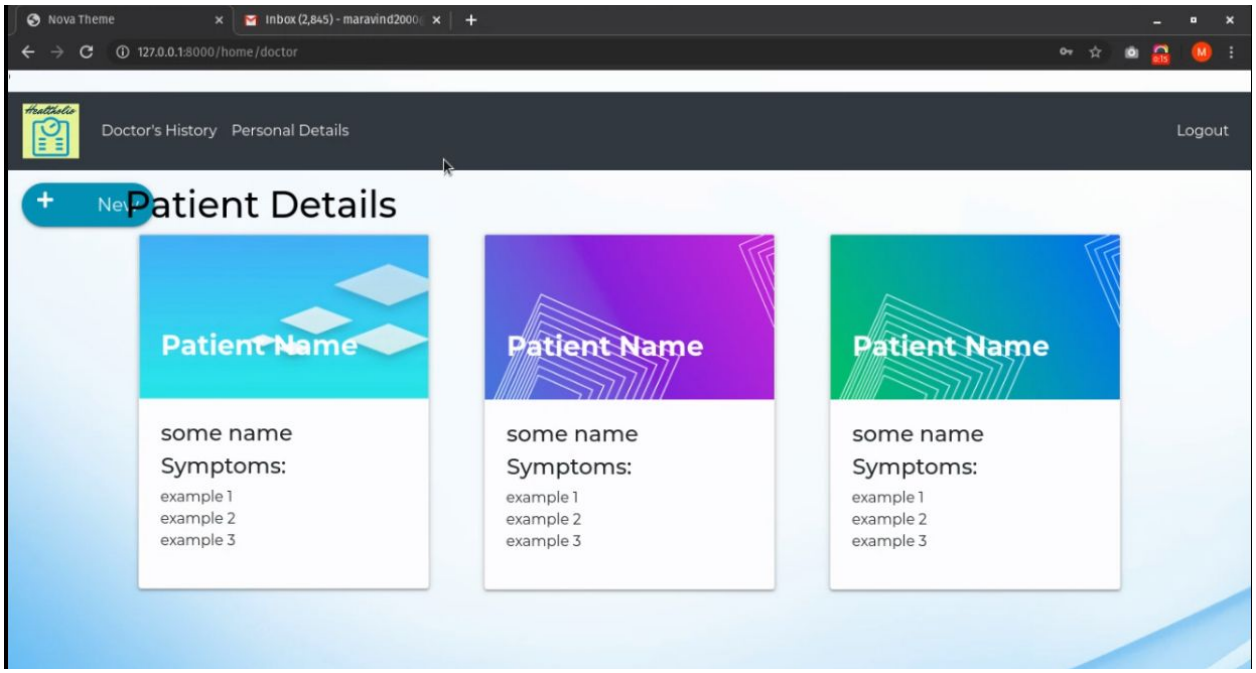
```
            med_data.append(d)
            d = {}

    Structured["Medicines"] = med_data
```

Results with screenshots

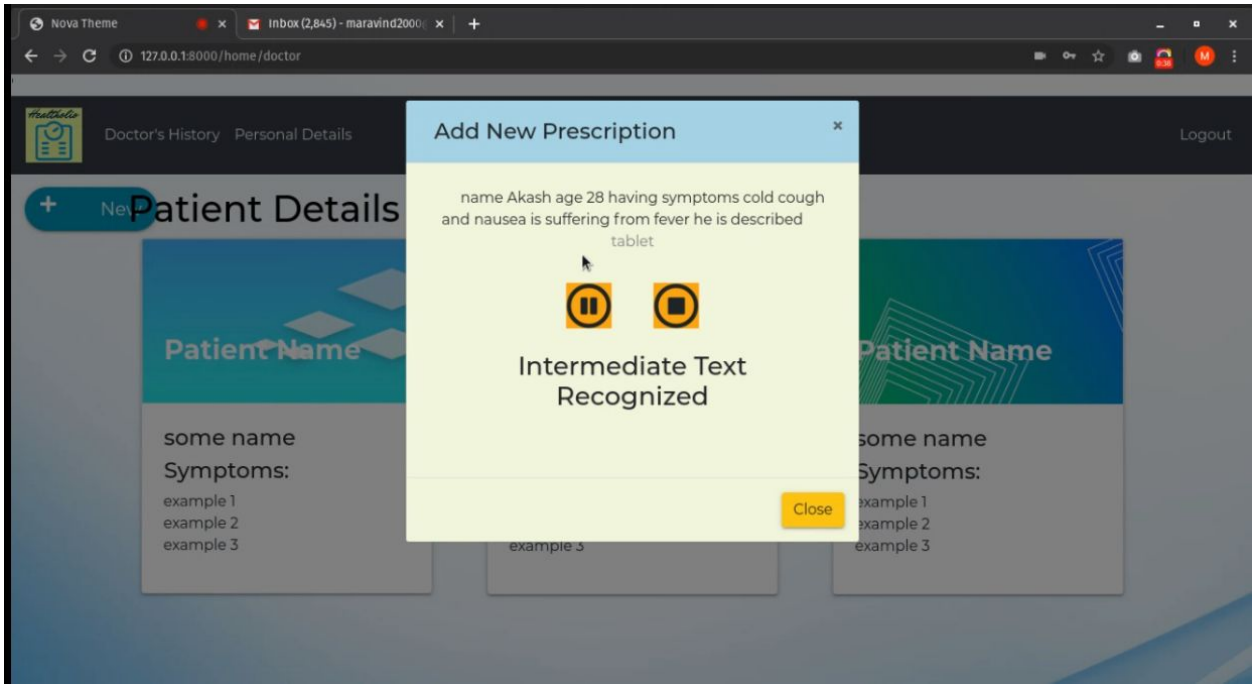
1. **Login screen**



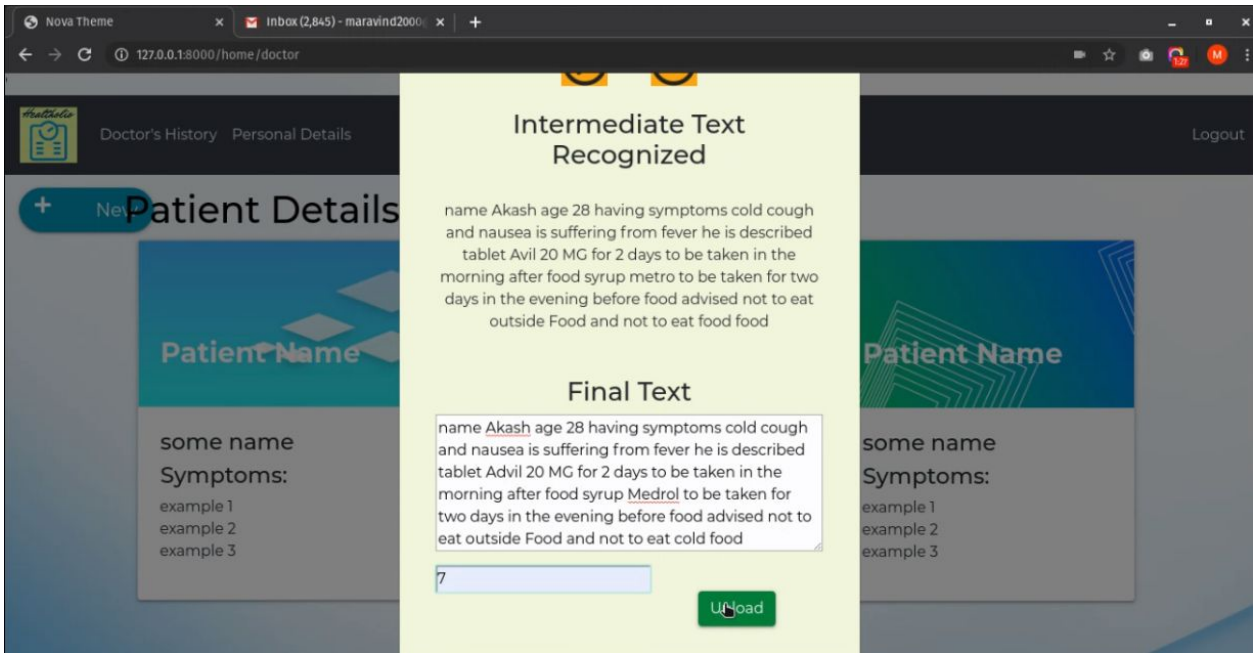
2. Doctor's Home page



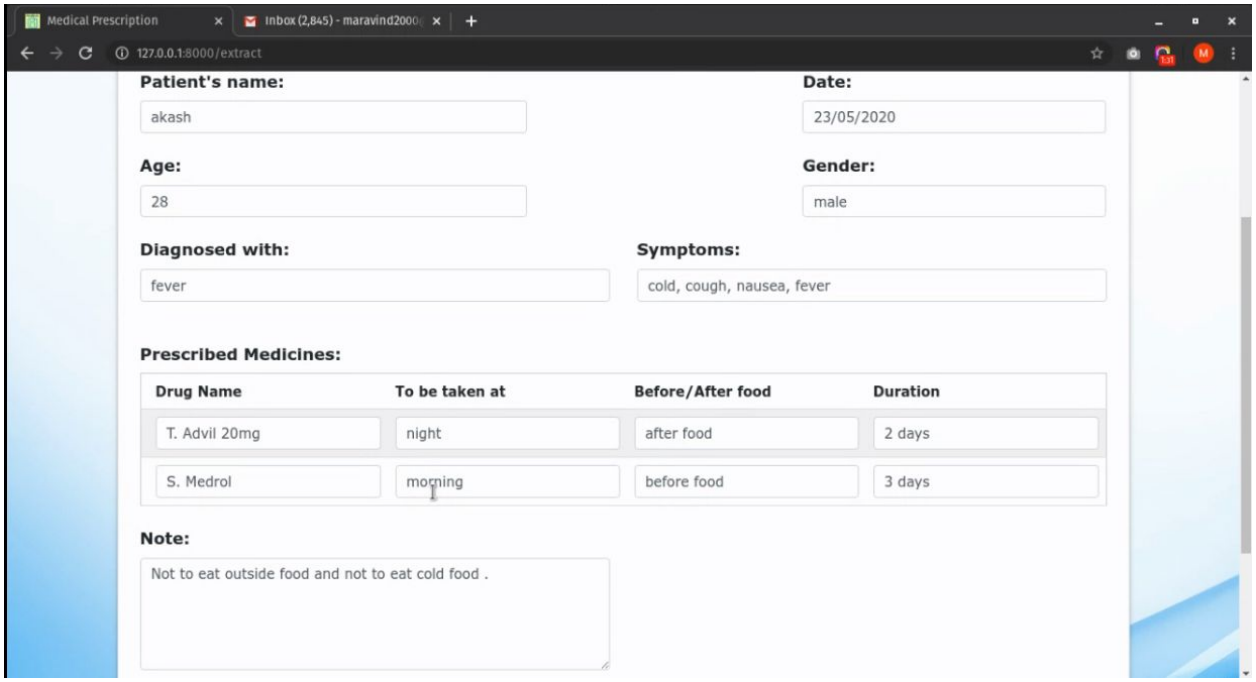
3. Speech to Text conversion



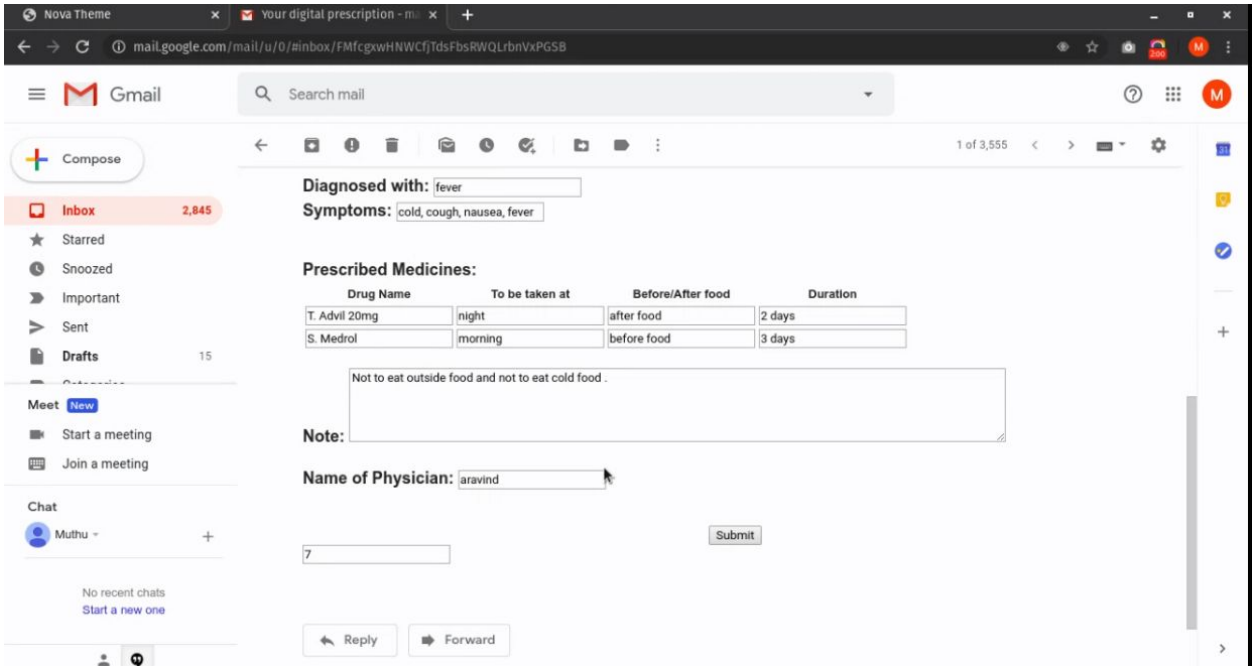
4. Final transcript



5. Final prescription



6. **Email of the prescription received by the patient**



Conclusion

The process of writing a prescription has been completely digitized and delivered directly to the patient's email inbox without any hassle. The proposed system, when adopted, would help in eliminating errors occurring due to indecipherable handwriting, drug interactions and confusing drug names as the prescription is digitized which would make it easy for pharmacists to read and give medicines to the patient.

Future work

● Voice assistant for patients to remind prescribed medication details, provide further assistance regarding the ongoing treatment such as follow-up visit, drug refill reminder and a feature to interact with the doctor for additional queries.

● Ensuring privacy and restricting unauthorized access to the medical records of the patient by incorporating block-chain technologies.

● A mobile app for patients and doctors and a desktop app for doctors.

References

1. Migowa, A. N., Macharia, W. M., Samia, P., Tole, J., & Keter, A. K. (2018). Effect of a voice recognition system on pediatric outpatient medication errors at a tertiary healthcare facility in Kenya. Therapeutic advances in drug safety, 9(9), 499-508.

2. Omoregbe, N. A., & Azeta, A. A. (2010). A voice-based mobile prescription application for healthcare services (VBMOPA). International Journal of Electrical and Computer Sciences, 10(2), 73-78.

3. Aggarwal, A., Garhwal, S., & Kumar, A. (2018). HEDEA: A Python Tool for Extracting and Analysing Semi-structured Information from Medical Records. Healthcare informatics research, 24(2), 148-153.

4. Fette G, Ertl M, Worner A, Kluegl P, Stork S, Puppe F. Information extraction from unstructured electronic health records and integration into a data warehouse. Proceedings of the 57th Annual Meeting of the German Society for Medical Informatics, Biometry and Epidemiology (GMDS); 2012 Sep 16-20; Braunschweig, Germany. p. 1237-51.

5. Eyesan, O. L., & Okuboyejo, S. R. (2013). Design and implementation of a voice-based medical alert system for medication adherence. Procedia Technology, 9, 1033-1040.

6. Wikipedia - https://en.wikipedia.org/wiki/List_of_medical_symptoms

7. RxList- https://www.rxlist.com/symptoms_and_signs/alpha_a.htm

8. Medscape - https://emedicine.medscape.com/

9. Google Web Speech API -https://developers.google.com/web/updates/2013/01/Voice-Driven-Web-Apps-Introduction-to-the-Web-Speech-API

10. NLTK Documentation - https://www.nltk.org/api/nltk.html