ACKNOWLEDGEMENT

This satisfaction that successful completion of any task would be incomplete without the mention of people whose ceaseless cooperation made it possible, whose constant guidance and encouragement crown all efforts with success. We are grateful to our guide **Prof. Om Prakash Pal** for the guidance, inspiration and constructive suggestions that helpful us in the preparation of this project. We also thank our colleagues who have helped in successful completion of the project.

ABSTRACT

The Health Care Sector is one of the most prominent research fields in the current scenario with rapid improvement of technology and data. Health Issues are now considered as an important factor after the pandemic. Ignoring symptoms and not taking care can lead to some Life- Threatening Causes. Identifying Disease at the initial stage helps doctors in Diagnosis and can also benefit the patient Financial wise as well as Health Wise. Machine Learning is also playing a key role in the advancement of Healthcare Technologies. With help of ML doctors are able to find out COVID 19 Strands using patient's X Ray and much more. Our Project principally targets disease prediction upon patient symptoms using various ML Models like Decision Tree, KNN. These ML models predict disease and using a voting classifier we came to the verdict of which disease is the patient suffering.

The other Major Problem for the Indian Clinics is to maintain patient records and appointment system. Indian Clinics still lack behinds in advancement of record management and booking appointments. Without a well-established booking pattern patients have to wait in long queues, just waiting for their turn to come to get treatment from the doctor. The old page recording data of patients may cause many problems like records get missing, wear and tear and many more. Thus, this needs to be strictly replaced with well managed recording application or software. Our Project principally targets improvement in the Record management System in Clinics.

Our Project is a web-based application which consists of HTML, CSS, JavaScript, Bootstrap for Frontend and uses Flask as Backend. For Databases we use MYSQL

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

1.1 Purpose

The Purpose of creating this project is to create a Clinic Management Environment alongside with features like Disease Prediction which is beneficiary for doctors to give fast diagnosis and record management for the clinics.

The Purpose to make the record management work for the Clinics Easy and also appointment booking helps to maintain order in the clinics thus people do not need to wait for long to get treatment. Digital Human Feature also creates virtual receptionists which can be used for Patient Appeal towards the Clinic.

1.2 Scope of the Project

The scope of the project lies in bound to the health care of people, the fastest and the most efficient way available to enable people who are un ease at health into proper guidance of the Doctors.

Accurate domain for us is to give the clinic and small healthcare infrastructure a boost in gratifying their reach with their skilled doctors via VITAL-AI. The entire project is accurately viable on both the side of the Doctor's clinic infrastructure and it also has a little impact on the patient's side.

The accurate scope is what we defined earlier in short. On a detailed note, the project involves various endpoints and users like doctors and patients, while other points include databases. The overall system revolves around civil and small-scale healthcare stops. That system is what we called VITAL-AI. The patient can interact with Elixir (our Digital-Human) and even with our symptom selection section, while the users who have already registered can view their registered slot and predicted disease via Elixir. On the other hand, the Doctors can view all booked slots and names, predict disease and Symptoms entered by the patient.

1.3 Objective

The objective involved behind making the VITAL-AI is that this helps the doctor-patient interaction and use time efficiently to book appointments over this. As it aims to help people not just book slots to meet doctors but, also self-analyze the symptoms that they have been feeling and list out to make the predictions about their possible disease as well as help the doctor to gain a quick look over the symptoms and pre-analyze the symptoms for possible diseases and then interact thus, making this instantaneously fast process for either of them.

It aims to be a reliable link directly between Doctors and Patients of small scale like private and small clinics, dispensaries and other small scale healthcare micro-enterprises.

1.4 About Disease Prediction

1.4.1 Classification

- Classification algorithm is a Supervised Learning technique that is used to identify the category of new observations on the basis of training data.
- In Classification, a program learns from the given dataset or observations and then classifies new observations into a number of classes or groups. Such as, Yes or No, 0 or 1, Spam or Not Spam, cat or dog, etc. Classes can be called as targets/labels or categories.
- Unlike regression, the output variable of Classification is a category, not a value, such as "Green or Blue", "fruit or animal", etc.
- Since the Classification algorithm is a Supervised learning technique, hence it takes labeled input data, which means it contains input with the corresponding output
- Example: Email Spam Detector

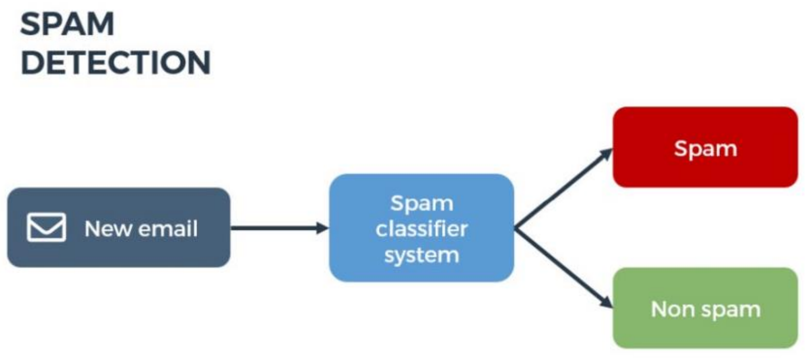


Figure: Spam Detection

1.4.2 Classification Type

- **Binary Classifier**: If the classification problem has only two possible outcomes, then it is called a Binary Classifier.
  EX: 1. YES or NO
  2. MALE or FEMALE
- **Multi-class Classifier**: If a classification problem has more than two outcomes, then it is called a Multi-class Classifier.
  EX: Classification of types of music, Classification of types of disease

1.4.3 Library Used

- NumPy
- Pandas
- Sklearn

1.4.3.1 NumPy

- NumPy is the core library of scientific computing in python. It provides powerful tools to deal with various multi-dimensional arrays in python. It is a general-purpose array processing package.
- It makes it easier to create a n dimensional array just by using np.zeros()

1.4.3.2 Pandas

- It is the most popular python library used for data analysis. It provides highly optimized performance with back-end source code purely written in C or python.

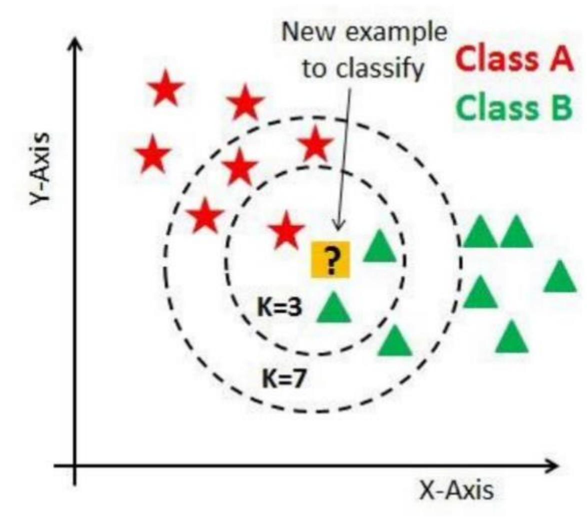
- Data in python can be analyzed with 2 ways
  1. Series: Series is a one-dimensional array defined in pandas used to store any data type.
  2. Data frames: Data frames are two-dimensional data structures used in python to store data consisting of rows and columns.

1.4.3.3 Sklearn

- It features various simple and efficient tools for data mining and data processing.
- It features various classification, regression and clustering algorithm such as support vector machine, random forest classifier, decision tree, gaussian naïve-Bayes, KNN

1.4.4 K – Nearest Neighbors (KNN)

- Classify by using "Majority Votes" for its neighbor's class
- Assigned to the most common class amongst its K- nearest neighbors
- Simple, but a powerful classification algorithm used in pattern recognition
- Simple Analogy: Tell me about your neighbors and I will tell you who you are



If we proceed with K=3, then we predict that test input belongs to class B,

If we continue with K=7, then we predict that test input belongs to class A.

Figure: KNN

1.4.5 Decision Tree

- Graphical representation of all the possible solutions to a decision
- Decisions are based on some conditions
- Decision made can be easily explained
- Most popular example of a decision tree is the toll-free number of particular services which redirects you to A.I. Assistant

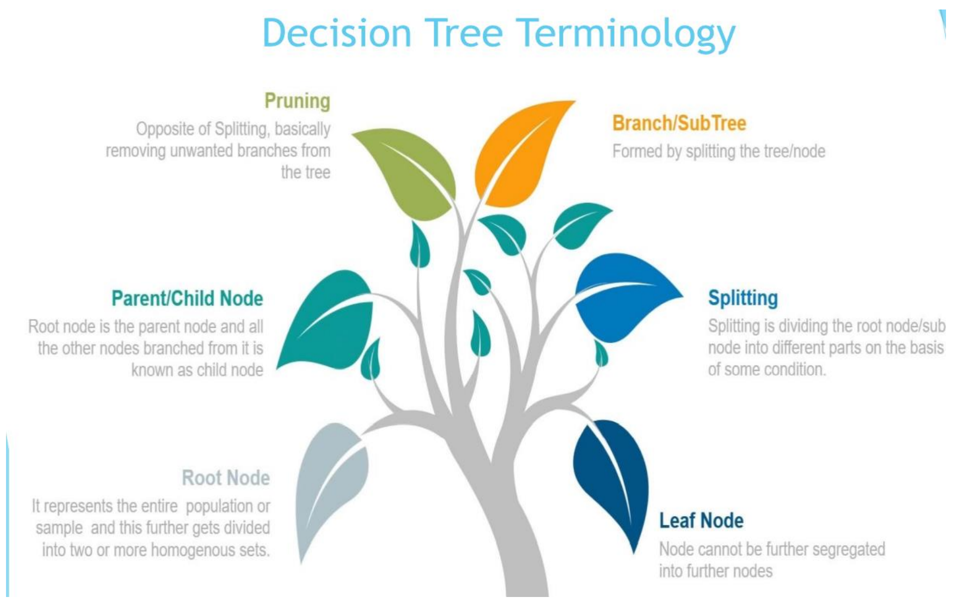

Decision Tree Terminology
Pruning
Opposite of Splitting, basically removing unwanted branches from the tree
Branch/SubTree
Formed by splitting the tree/node
Parent/Child Node
Root node is the parent node and all the other nodes branched from it is known as child node
Splitting
Splitting is dividing the root node/sub node into different parts on the basis of some condition.
Root Node
It represents the entire population or sample and this further gets divided into two or more homogenous sets.
Leaf Node
Node cannot be further segregated into further nodes

Figure: Decision Tree

1.4.6 Voting Classifier

The voting classifier is an ensemble learning method that combines several base models to produce the final optimum solution. The base model can independently use different algorithms such as KNN, Random forests, Regression, etc., to predict individual outputs. This brings diversity in the output, thus called Heterogeneous ensembling. In contrast, if base models use the same algorithm to predict separate outcomes, this is called Homogeneous ensembling.

1.4.7 Soft Voting

In Soft voting, Classifiers or base models are fed with training data to predict the classes out of m possible courses. Each base model classifier independently assigns the probability of occurrence of each type. In the end, the average of the possibilities of each class is calculated, and the final output is the class having the highest probability.

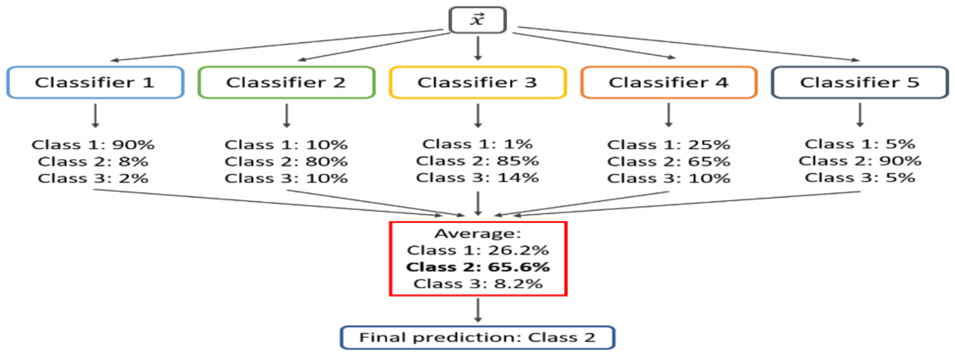

$\vec{x}$
Classifier 1
Classifier 2
Classifier 3
Classifier 4
Classifier 5
Class 1: 90% Class 2: 8% Class 3: 2%
Class 1: 10% Class 2: 80% Class 3: 10%
Class 1: 1% Class 2: 85% Class 3: 14%
Class 1: 25% Class 2: 65% Class 3: 10%
Class 1: 5% Class 2: 90% Class 3: 5%
Average: Class 1: 26.2% Class 2: 65.6% Class 3: 8.2%
Final prediction: Class 2

Figure: Soft Voting

1.5 Problem Statement

Keeping all the inherent limitations of the existing technologies in mind and sensing the need of a better encryption model for multimedia data, in terms of security as well as time, a system is proposed, which addresses the problems such as:

- **Record Management and Booking Appointment**:
  The MYSQL database will be used to store user details like (First name, Last name, EmailID, Password). Alongside Patient booking details are also stored which can be accessed by Patient and Doctor but CRUD Operations can be performed by Admin.

- **Disease Prediction**:
  The Disease Prediction can be achieved using various Machine Learning Models like Decision Tree, K Nearest Neighbor which takes symptoms as input from the user and predicts the disease at certain Accuracy.

- **Patient Appealing**:
  Digital Human integrating with dialog flow is used to create a Virtual Receptionist and clinic environment which will be assigned to take patient's previous medical history and other stuff.

2. FEASIBILITY STUDY

2.1 Study of the Current System:

2.1.1 Ada

- It is a useful app that provides quite a lot of services under the health section and also predicts the disease or reason for uneasiness.
- **Features:**
    - fast and accurate disease prediction
    - doctors can be booked even without predicting or analyzing your disease
    - smooth UI
    - readily available on android and IOS
    - constant updates (app updates)
    - trusted by many people
    - uses data hierarchy for predicting diseases accurately
    - contact in house medical experts.
- **Drawbacks**
    - smooth UI but unpredictable behavior sometimes.
    - no local advisors exist whom you can meet in person.
    - physical examination not possible, as it is online (virtual meets)

2.1.2 Practo

- A similar internet-based application that connect patients world-wide to doctors and medical experts
- **Features:**
    - books appointment and saves history of it as well.
    - quick and closest doctor bookings
    - almost all of the doctors available are experts
    - global scale application
    - dashboard of booking available on login instantly
- **Drawbacks**
    - experts charge high price
    - not used too frequently as is a third-party app
    - dashboard of booking available on login instantly but doesn't show disease that is predicted
    - 3rd party thus avoided due to fear of exposing personal data
    - patient counts visible to every doctor of each doctor's patients which is not ideal
    - no use of ML
    - no prediction happening here. It is just a booking system; it cannot predict any disease or even take in symptoms as list and show to doctors
- **Problem of the weakened system:**
    - requires login to show the predicted disease
    - not available on all the Operating Systems
    - Complex to use (not easy to use for naive users/innocent/sophisticated /standard users)
    - needs to be heavily configured

- needs all of its servers and databases in sync and also manage data failures or breaches
    - needs training before you can use it fully for doctors specially.
    - unpredictable behavior sometimes.
    - doesn't us ML
    - 3rd party applications and websites have trust issues
    - unreliable behavior when the whole system experiences stress (flooded by traffic)

2.2 Requirement of the better system:

- **Reachability**: better reachability produces better results, if the application is reachable and is used by almost every small clinic it's reachability will increase as everyone will have a location to go over to book a slot for themselves.
- **Accessibility**: even though applications are reliable but if they are not over multiple platforms, not everyone can access it, like some use other OS then Android or IOS then the app is not accessible by them, like if they use Kai OS or Linux, Windows, Windows phone OS, Blackberry OS, Symbian OS, Chrome OS, Tizen OS, etc.
- **Pricing**: not all apps give predictions that are correct to the users directly, the users need to pay for them which might not be a good idea for everyone. Once in a while it's ok but paying for every prediction it's' kind of prizy.
- **Trust**: As the global apps grow there are more and more issues with 3rd party data breaching, information leaks, etc. As not all third parties provide accurate protection on data leaks, breaches or other un-trustable works like data mining. Selling of private data.
- **Complexity to use**: to use these online apps you need enough training for doctors especially as it's not easy for them to understand compiled data given to them easily.

2.3 Technical Feasibility:

- The technical requirement for the system is economic and it does not use any additional Hardware and software.
- This is a very important aspect to be considered while developing a project. We decided on the technology based on the minimum cost factor.
- The system being developed is economic with respect to the user's point of view. It is cost effective in the sense that it has eliminated the paperwork & transportation cost completely.
- The system is also time effective as the system allows the user to see the doctor virtually extremely fast based upon the schedule that user has chosen for, adding to that it also gives the doctor the symptoms beforehand that gives him time to analyze time to diagnose the user (patient) to some extent.

2.4 Operational Feasibility:

- There is no doubt the proposed system is fully GUI based and is very user friendly. It is very simple and overhauling it completely has made it a very attractive interface after rework. Users do not require any additional training to use this software.

- Operating this is not at all a huge task as it comes to be extremely user friendly in addition to its UI. The seps are already shown on the home page about how to use it. As far as the Reliability is concerned it's up as per the clinic/dispensary specific timelines and guidelines as they are the ones that are being operational on the server. Its' customizable for doctors as per their timings and their schedules.

2.5 Requirement Validation:

- E-mail ID
- Contact No

2.6 Features of New System:

- Vital-Ai Assistant system and digital human called Elixir.
- Symptom Prediction
- Booking system

2.7 Hardware and Software Requirement:

2.7.1 User side hardware requirement:

- Device: Any Device that supports Web browsing and internet-base
- RAM: Minimum 2 GB on system (just for better Performance, it has nothing to do with Vital-AI)
- Operating System: Any OS that supports Web browsing and internet-base

2.7.2 User side hardware requirement:

Component	**Minimum**	**Recommended**
Processor	Intel Core i3 5th Gen or equivalent to AMD	Intel Core i5 7th Gen or equivalent to AMD or over
RAM	4 GB	8 GB or above
Display	800 * 600	1920 * 1080 or more
HDD/SSD	20 GB	40 GB and above

Network	100kbps between client computers and server	100kbps or faster between client computers and server and over
OS	Windows 7 or 8	Windows 10 or 11

2.8 Literature Survey:

Data taken from University of Columbia:
http://people.dbmi.columbia.edu/~friedma/Projects/DiseaseSymptomKB/index.html

Research Papers:
Disease Prediction From Various Symptoms Using Machine Learning by Rinkal Keniya, Aman Khakharia, Vruddhi Shah, Vrushabh Gada, Ruchi Manjalkar, Tirth Thaker, Mahesh Warang, Ninad Mehendale :: SSRN

https://www.researchgate.net/publication/347381005_Disease_Prediction_Using_Machine_Learning

2.9 Assumptions and Dependencies:

The Diseases predicted by our system i.e., the disease predicted by all four algorithms are not totally accurate, these predictions are made on the basis performing advanced processing of the dataset created by University of Columbia.

The Accuracy of all the 2 models cannot reach 100 percent as any model in the real world is 100 percent accurate.

User’s Internet can act as a Dependencies factor because if the user doesn't have stable network access some function may throw errors because they are Internet Dependent.

3. SYSTEM REQUIREMENTS STUDY

3.1 Functional Requirement

1. Proceed Module

Term	**Description**
Input	Click on the Proceed button available on Home Page or Hover the mouse over Proceed dropdown list in Navigation bar
Output	1. When clicking on the home page Proceed button, the Login Page will be viewed. 2. On Navigation Bar Event: 1. When Click on Login, Login Page will be viewed. 2. When Click on Register, Register Page will be viewed.

2. Login Module

Term	**Description**
Purpose	To authenticate the user
Input	Email, Password
Output	If the password matching is successful then the system will allow the user to perform operations.

Process	When a user enters email id and password, email id and password are checked in the database. If the details are correct then the user account is displayed.

3. Registration Module

Term	**Description**
Purpose	This module is specified for the users who are not registered to the application, and want to access additional features and functions. Here, at the time of registration, the user profile is created.
Input	First Name, Last Name, Email ID, Password
Output	User will navigate to the login Page.
Process	When the user inputs data into the fields and submits the form fields like email id, password and first name, last name, Password his/her account will be, his/her account is created and data is stored in the user's table. Passwords will be stored in hash form.

4. Logout Module

Term	**Description**
Purpose	Logout module is generally used when the user does not want to browse over the application and wants to be logged out.

Input	Click on Logout button
Output	User is logged out from the application and returns to the login page and cannot access any functions and flash message will be displayed to the user.
Process	When user clicks on the Log out Button, the request is sent to the backend to end the current user session. With the acceptance of the request, the session is terminated and the user is logged out of the application.

5. Book Now Module

Term	**Description**
Input	Click on Book Now button on Patient Dashboard
Output	Symptom Prediction Page is viewed to the user

6. Symptom Selection Module

Type	**What to do**	**Description**
How to reach	Press” Book Now” on Patient Dashboard	It will redirect you to the Symptom Prediction Page.

Input	Click on symptom 1	Select any symptom that you think you have
	Click on symptom 2	Select any symptom that you think you have
	Click on symptom 3	Select any symptom that you think you have
	Click on symptom 4	Select any symptom that you think you have
	Click on symptom 5	Select any symptom that you think you have
	Click on submit report	
Process	It sends the User to the Digital Human Page.	The Symptoms are pass as parameter in ML model in backend
Output	-	It will redirect you to Vital Ai Page

7. Digital Human Module

Term	**Description**
Purpose	The Digital Human is used to get some patient appeal by creating a Virtual persona which takes the additional details from the patient.

Input	User gives the answer to the questions asked by the Elixir.
Output	Elixir will give Disease Prediction Output.
Process	The Question asked by the Digital Human is coming from dialog flow and the answers are taken back to backend for storing and booking purposes.

3.2 Non-Functional Requirement

3.2.1 Software Flexibility

The main attribute of the system is its ability to enable flexible access to information and resources. Flexible access to resources means the Vital AI is going to be used on any desktop or laptop or any mobile devices with some basic requirements. The User can login and can book appointments from any part of the world.

3.2.2 Scalability Requirements

The system must be scalable enough to be able to add any additional functionality even after the project is developed once.

3.2.3 Usability Requirements

The application is user-friendly and provides ease of access for users. If users find some trouble in accessing a system, they can inform admins of that clinic.

4. SYSTEM DESIGN

4.1 USE CASE DIAGRAM

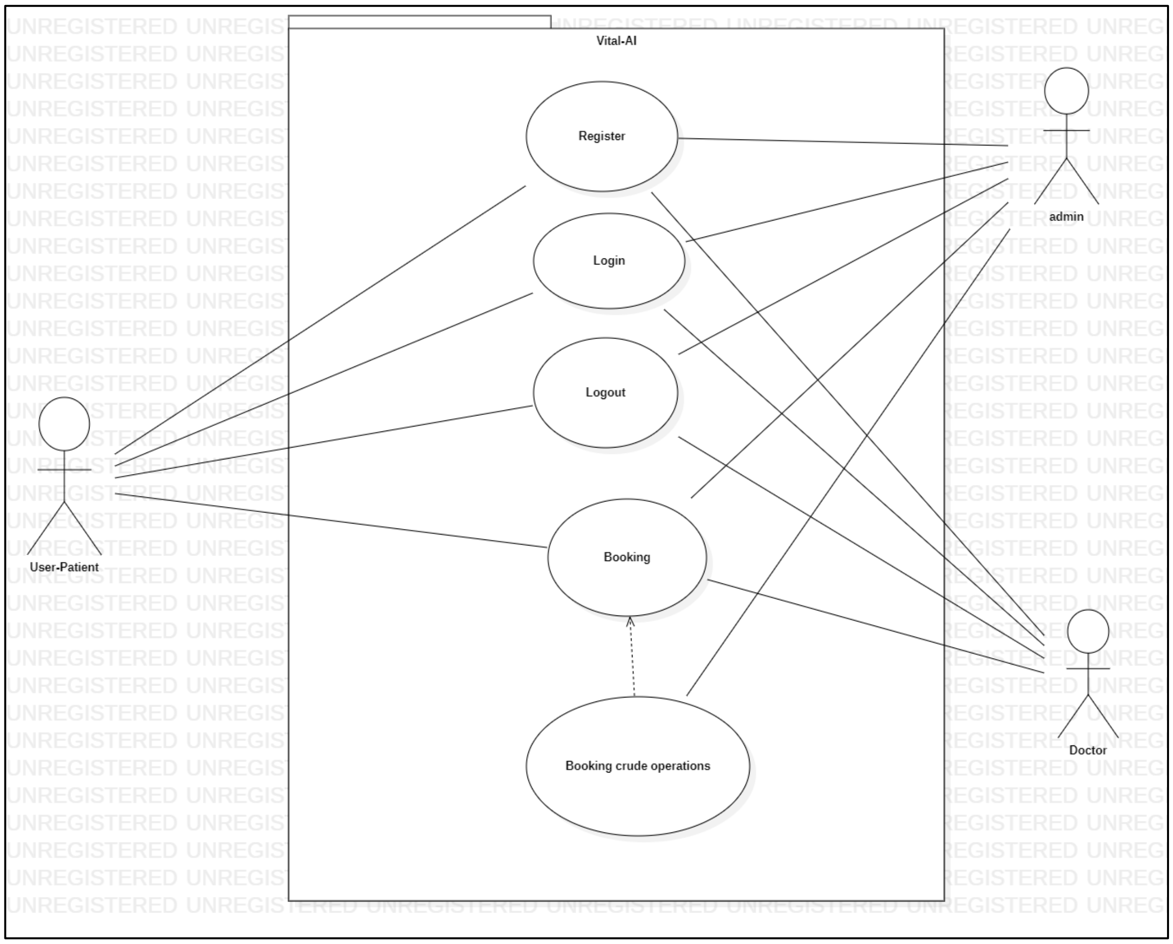


Figure: Use Case Diagram

4.2 ER DIAGRAM

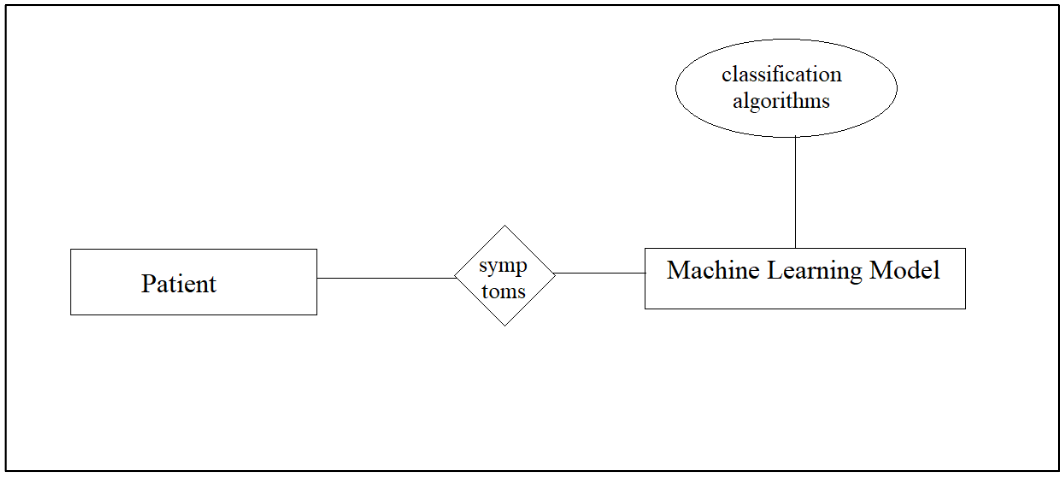


Figure: ER Diagram

4.3 Activity diagram

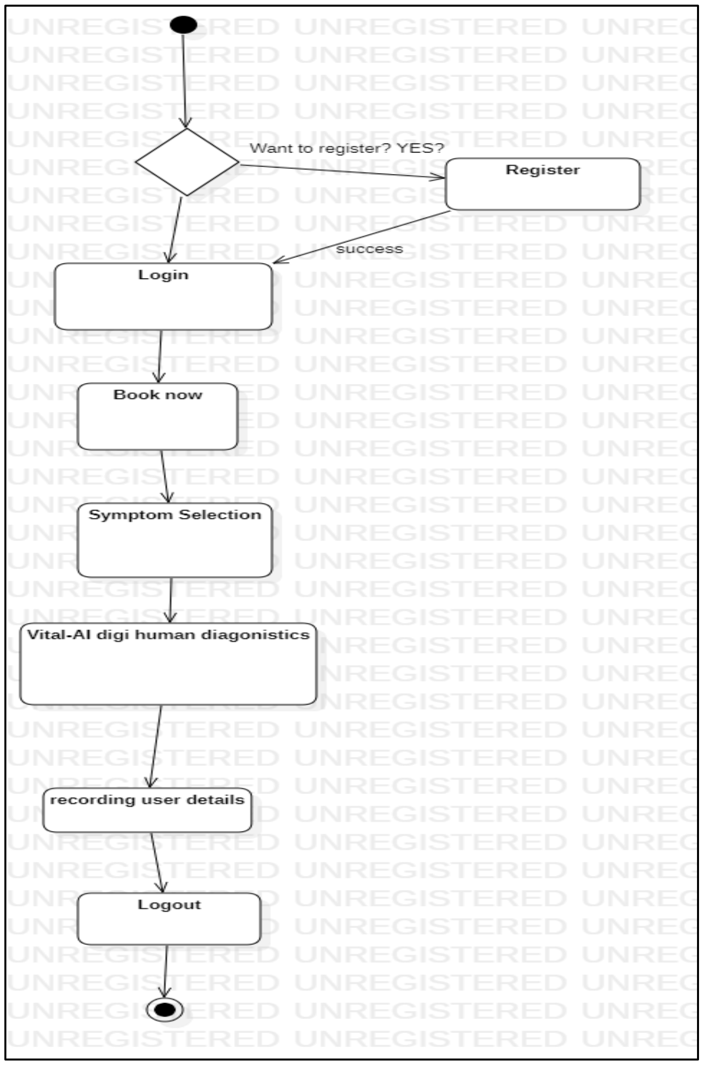


Figure: Activity Diagram

4.4 Class diagram

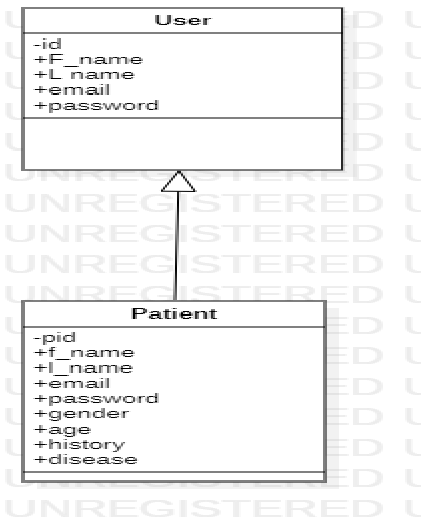


Figure: Class Diagram

4.5 State diagram

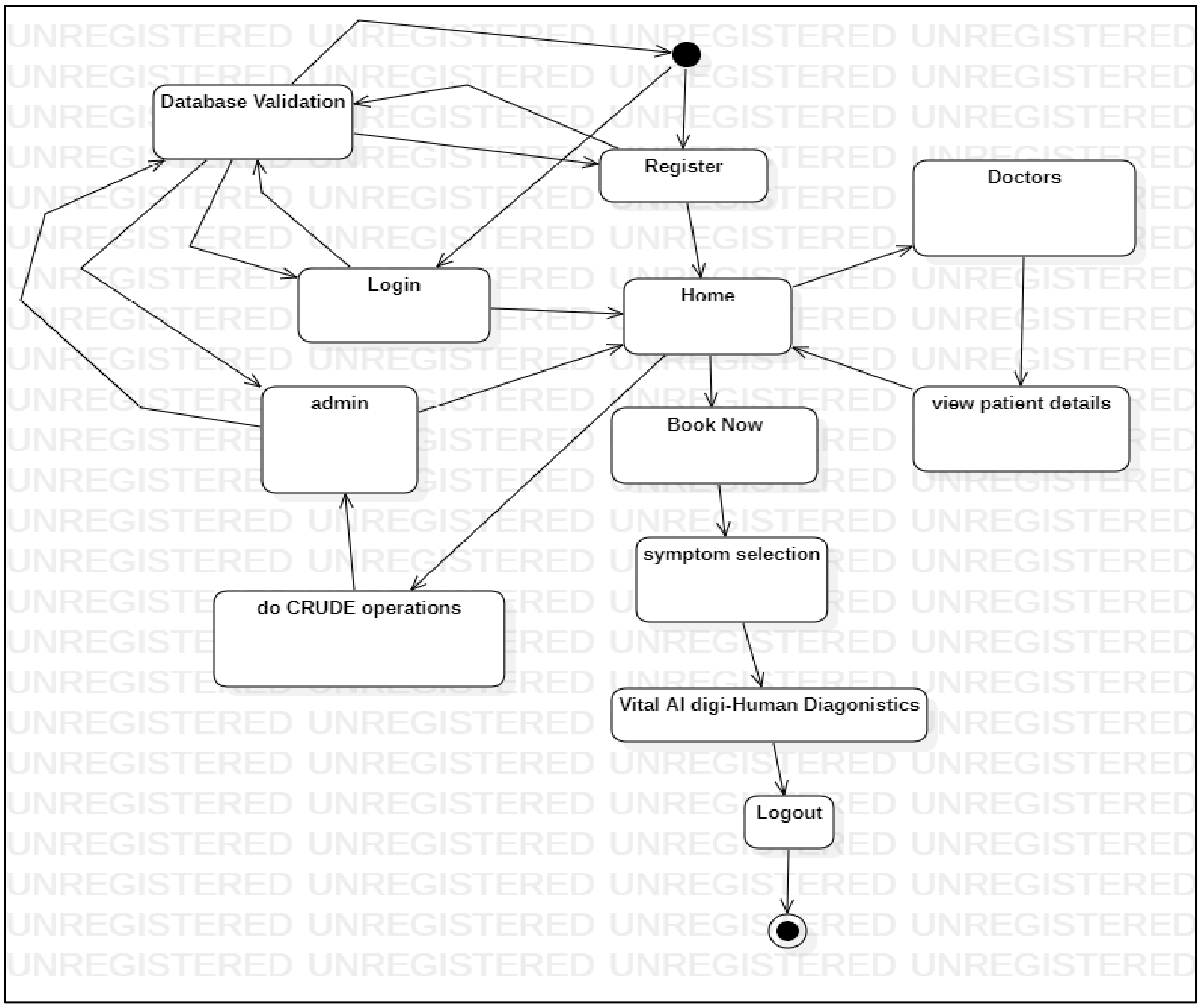


Figure: State Diagram

4.6 Sequence diagram

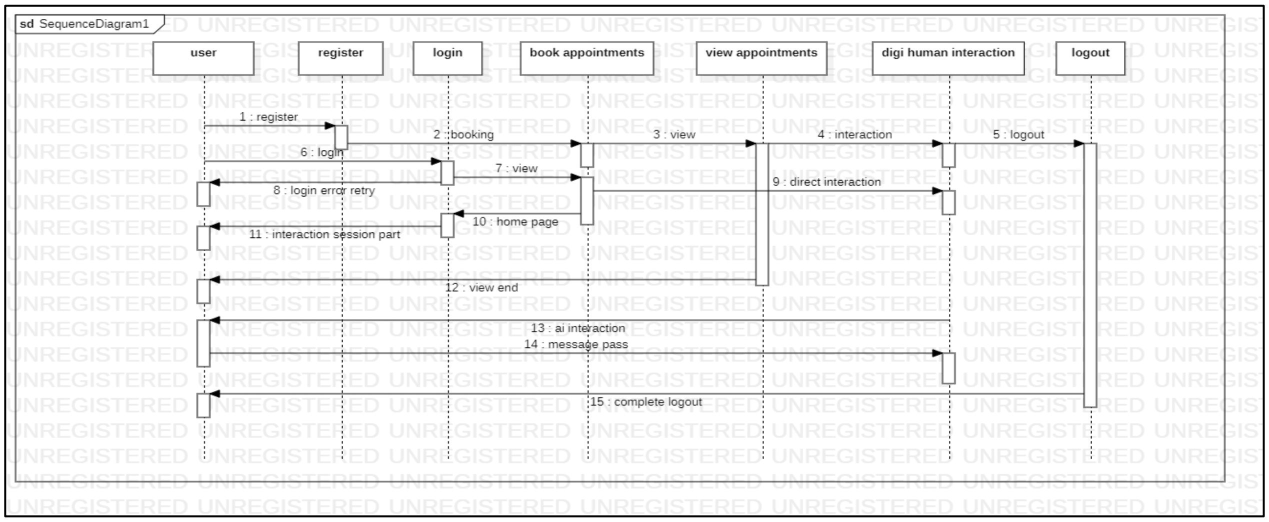


Figure: Sequence Diagram

4.7 Data Flow Diagram

4.7.1 DFD lvl-0

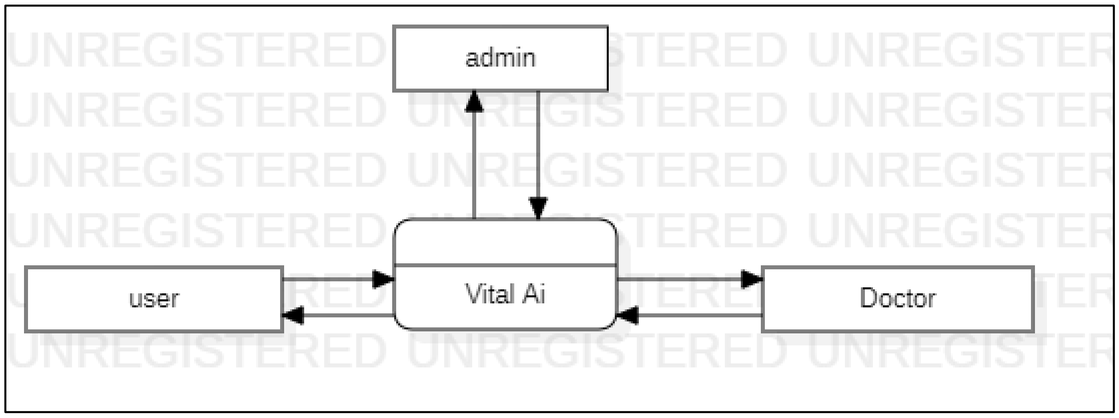


Figure: DFD lvl-0

4.7.2 DFD lvl-1

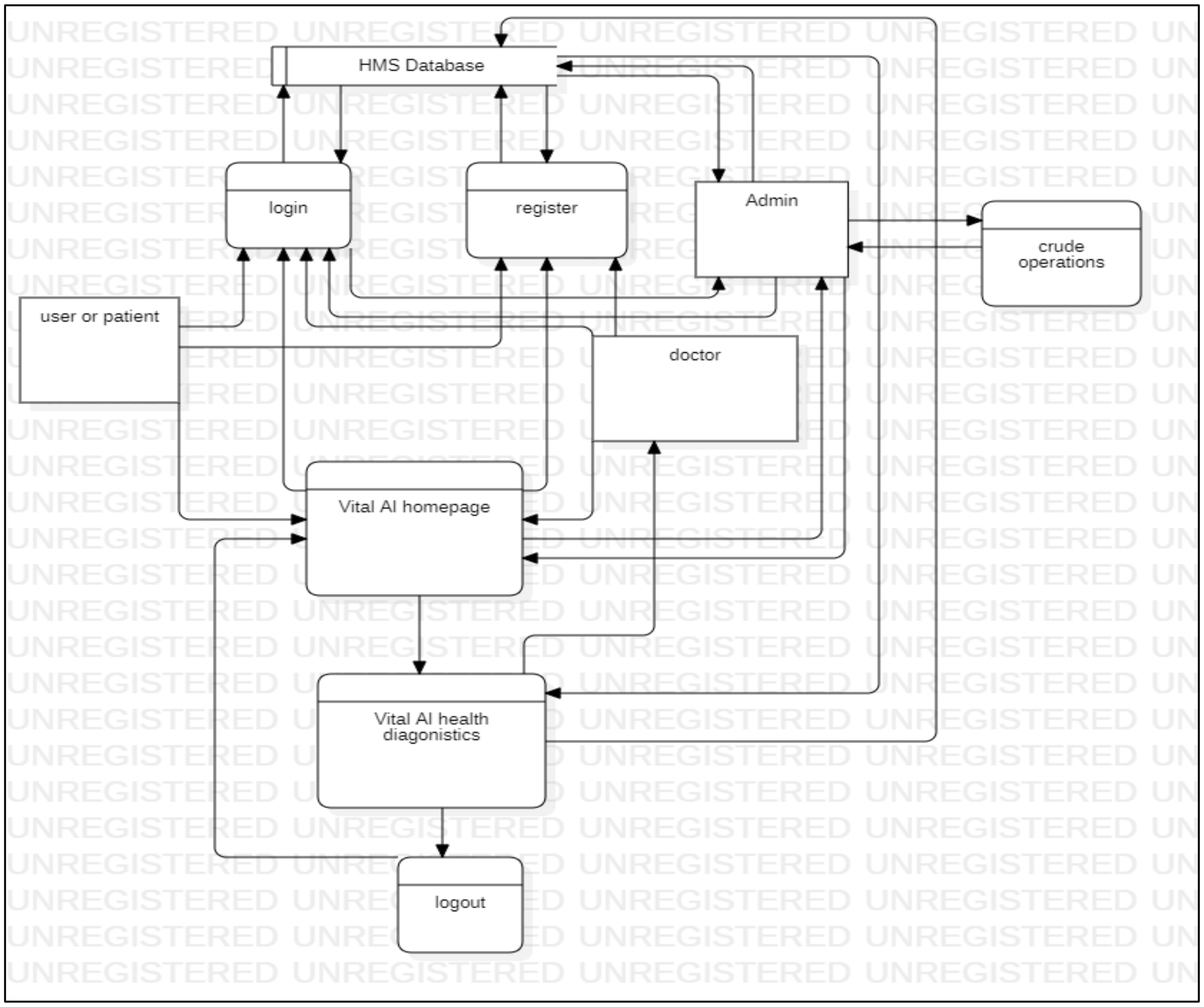


Figure: DFD lvl -1

4.7.3 DFD lvl-2

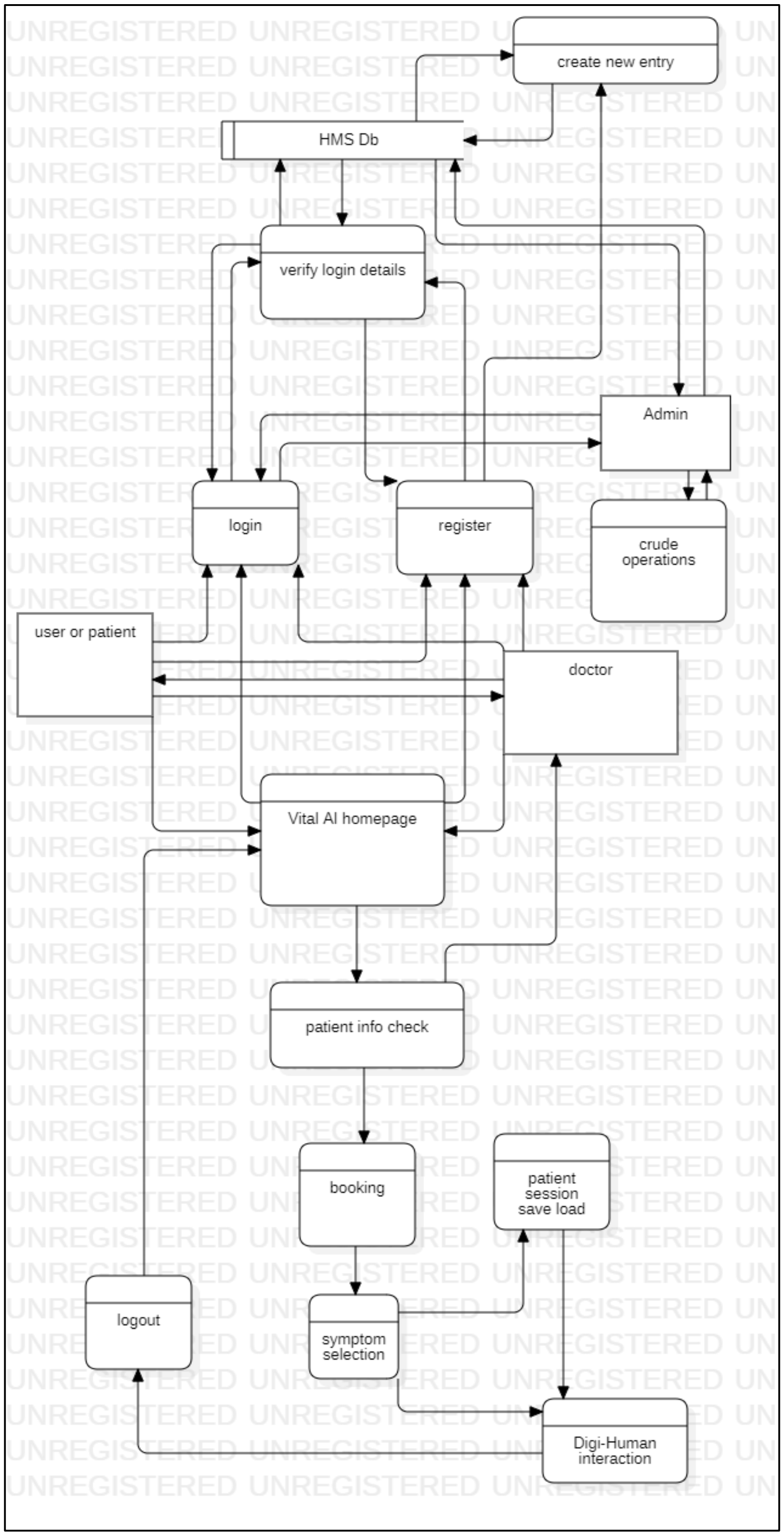


Figure: DFD lvl-2

5. PROCESS MODEL AND APPLICATION PROCESS

5.1 Process Model to Get register in Vital AI and Book Appointment.

1. Users need to create an Account or if he/she is an existing user they can login.
2. After Successful Login, User is redirected towards the Home Page.
3. Users can click on the Patient button on the navigation bar to go to the Patient Page.
4. Users can click on the book now button on the Patient Page or Welcome page to start the Booking Process.
5. First the user needs to fill the symptoms form by choosing the symptoms they are suffering from the drop-down menu and then click on the submit button.
6. Then User is redirected towards the digital human page where the user will interact with doctors virtual Assistant Elixir which will ask questions to the patient regarding their past medical history.
7. These Virtual humans will keep recording the data and storing it to the database so it will be visible to the doctor's Page.

5.2 Backend Process of Web Application

1. The Vital AI is running using flask and hosted using ngrok.
2. Navigation of web pages, Database creation and addition of rows and other operations are carried out using main.py
3. Using @app.route each webpage is accessible to the user as it is passed as Render Templates.
4. SQLALCHEMY is used to connect With MYSQL Database 'hms'.
5. Each Table in database is access in main.py by creating class of that table and declaring all the attributes in the class of it in main.py
6. Different Functions are created for the carried-out operations on the Web Page like Disease Prediction, Login, Register, etc.
7. Jinja Templates alongside HTML, CSS are used to create webpages.

5.3 Disease Prediction Process

1. Disease Prediction Module is created as "DT.py" and it is imported in Main.py
2. This File Consist of Function called "dt_pred" which takes 5 Symptoms that are selected by the patient as parameters.
3. We Used the dataset of Disease prediction from Columbia State University.
4. All the symptoms of these dataset are stored in the list called l1.
5. We have also created another list which consists of disease.
6. Training.csv and Testing.csv are read and stored as Data frame.
7. The Data inside this Data frame is replaced with prognosis i.e., the disease is assigned value serial wise.
8. Classification models are applied and disease is predicted.
9. If the disease is found then these modules will return the disease name else it will return Fit.

5.4 Digital Human and Dialog Flow Integration

1. Digital Human API is integrated with Dialog Flow Agent Thus every intent we set for the agent in Dialog Flow like taking records is visually performed by digital human.
2. Few fallback intents are created to take records from the patient.

6. CODE IMPLEMENTATION

6.1 Main.py

```
from flask import Flask,render_template, url_for, make_response
from flask_sqlalchemy import SQLAlchemy
from flask import request,redirect,session,flash
from flask_login import UserMixin
from werkzeug.security import generate_password_hash,check_password_hash
from flask_login import login_user,logout_user,login_manager,LoginManager
from flask_login import login_required,current_user
import json, os
import DT
from flask_cors import cross_origin
#import mail

# My Db Connection
local_server = True
app = Flask(__name__)
app.secret_key = 'vitalai'

#this is for getting unique user access
login_manager=LoginManager(app)
login_manager.login_view='login'

@login_manager.user_loader
def load_user(user_id):
    return User.query.get(int(user_id))

app.config['SQLALCHEMY_DATABASE_URI']='mysql://root:@localhost/hms'
db = SQLAlchemy(app)

class Test(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    name = db.Column(db.String(100))
    email = db.Column(db.String(100))

class User(UserMixin,db.Model):
    id = db.Column(db.Integer, primary_key=True)
    fname = db.Column(db.String(50))
    Lname = db.Column(db.String(50))
    email = db.Column(db.String(50), unique=True)
    password = db.Column(db.String(1000))

#try:
#       Test.query.all()
#       return 'My Database is connected'
#    except:
#      return 'My Database is not connected'
@app.route("/")
```

```
return render_template('index.html')

@app.route("/doctors")
def doctors():
    return render_template('doctors.html')

@app.route("/bot")
def bot():
    return render_template('bot.html')

@app.route("/disease",methods=["POST","GET"])
def disease():
    if request.method == "POST":
        global sym1,sym2,sym3,sym4,sym5,dis
        sym1 = request.form.get('sym1')
        sym2 = request.form.get('sym2')
        sym3 = request.form.get('sym3')
        sym4 = request.form.get('sym4')
        sym5 = request.form.get('sym5')
        print(sym1,sym2,sym3,sym4,sym5)

        try:
            dis = DT.dt_pred(sym1,sym2,sym3,sym4,sym5)
            return redirect(url_for('bot'))

        except ValueError:
            print("Invalid Credentials")
            return render_template('disease.html')

    return render_template('disease.html')

@app.route('/webhook', methods=['GET','POST'])
@cross_origin()
def webhook():

    req = request.get_json(silent=True, force=True)

    print("req")


    res = processRequest(req)

    res = json.dumps(res, indent=4)
    #print(res)
    r = make_response(res)
    r.headers['Content-Type'] = 'application/json'
```

```
#sessionID=req.get('responseId')
    result = req.get("queryResult")
    #print(result)

    parameters = result.get("parameters")
    diag1 = parameters.get("diag1")
    print(diag1)
    diag2 = parameters.get("diag2")
    diag3 = parameters.get("diag3")
    print(dis)
    intent = result.get("intent").get('displayName')
    if (intent=='Default Welcome Intent - yes'):
        if(dis == '0'):
            status = 'Not Found'
        else:
            status = "You are Found with this"+dis+"Disease."


        fulfillmentText= status
        print(fulfillmentText)
        return {

            "fulfillmentText": fulfillmentText
        }

# return render_template('bd.html', fname=current_user.fname, Lname=current_user.Lname)
#if not User.is_authenticated:
@app.route("/bd")
def bd():

    return render_template('bd.html')

@app.route("/login", methods=["POST","GET"])
def login():
    if request.method == "POST":


        email = request.form.get('email')
        password = request.form.get('password')
        user = User.query.filter_by(email=email).first()

        if user and check_password_hash(user.password,password):
            login_user(user)
            return redirect(url_for('index'))

        else:
```

```
@login_required
def logout():
    logout_user()
    flash("Logout SuccessFul","warning")
    return redirect(url_for('login'))

@app.route("/patients")
@login_required
def patients():
    return render_template('patients.html',fname=current_user.fname, Lname=current_user.Lname)

@app.route("/signup", methods=["POST","GET"])
def signup():
    if request.method == "POST":
        fname = request.form.get('fname')
        Lname = request.form.get('Lname')
        email = request.form.get('email')
        password = request.form.get('password')
        user = User.query.filter_by(email=email).first()
        if user:
            flash("Email Already Registered","warning")
            return render_template('signup.html')

        encpassword = generate_password_hash(password)
        new_user = db.engine.execute(f"INSERT INTO `user` (`fname`, `Lname`, `email`, `password`) VALUES ('{fname}','{Lname}', '{email}', '{encpassword}')")

        flash("Successfully Signup, Please Login","success")
        return render_template('login.html')


    return render_template('signup.html')

app.run(debug=True)
```

6.2 DT.py

```
import numpy as np
import pandas as pd
from sklearn import tree
from sklearn.metrics import classification_report,confusion_matrix,accuracy_score

def dt_pred(symp1,symp2,symp3,symp4,symp5):

    l1=['back_pain','constipation','abdominal_pain','diarrhoea','mild_fever','yellow_urine',
        'yellowing_of_eyes','acute_liver_failure','fluid_overload','swelling_of_stomach',
        'swelled_lymph_nodes','malaise','blurred_and_distorted_vision','phlegm','throat_irritation',
         'redness_of_eyes','sinus_pressure','runny_nose','congestion','chest_pain','weakness_in_limbs',
        'fast_heart_rate','pain_during_bowel_movements','pain_in_anal_region','bloody_stool',
        'irritation_in_anus','neck_pain','dizziness','cramps','bruising','obesity','swollen_legs',
        'swollen_blood_vessels','puffy_face_and_eyes','enlarged_thyroid','brittle_nails',
        'swollen_extremeties','excessive_hunger','extra_marital_contacts','drying_and_tingling_lips',
        'slurred_speech','knee_pain','hip_joint_pain','muscle_weakness','stiff_neck','swelling_joints',
        'movement_stiffness','spinning_movements','loss_of_balance','unsteadiness',
        'weakness_of_one_body_side','loss_of_smell','bladder_discomfort','foul_smell_of urine',
        'continuous_feel_of_urine','passage_of_gases','internal_itching','toxic_look_(typhos)',
       'depression','irritability','muscle_pain','altered_sensorium','red_spots_over_body','belly_pain',
                                                           'abnormal_menstruation','dischromic
_patches','watering_from_eyes','increased_appetite','polyuria','family_history','mucoid_sputum',
        'rusty_sputum','lack_of_concentration','visual_disturbances','receiving_blood_transfusion',
        'receiving_unsterile_injections','coma','stomach_bleeding','distention_of_abdomen',
         'history_of_alcohol_consumption','fluid_overload','blood_in_sputum','prominent_veins_on_
calf',
        'palpitations','painful_walking','pus_filled_pimples','blackheads','scurring','skin_peeling',
         'silver_like_dusting','small_dents_in_nails','inflammatory_nails','blister','red_sore_around_n
ose',
        'yellow_crust_ooze']

    disease=['Fungal infection', 'Allergy', 'GERD', 'Chronic cholestasis',
       'Drug Reaction', 'Peptic ulcer diseae', 'AIDS', 'Diabetes ',
       'Gastroenteritis', 'Bronchial Asthma', 'Hypertension ', 'Migraine',
       'Cervical spondylosis', 'Paralysis (brain hemorrhage)', 'Jaundice',
       'Malaria', 'Chicken pox', 'Dengue', 'Typhoid', 'hepatitis A',
       'Hepatitis B', 'Hepatitis C', 'Hepatitis D', 'Hepatitis E',
       'Alcoholic hepatitis', 'Tuberculosis', 'Common Cold', 'Pneumonia',
       'Dimorphic hemmorhoids(piles)', 'Heart attack', 'Varicose veins',
       'Hypothyroidism', 'Hyperthyroidism', 'Hypoglycemia',
       'Osteoarthristis', 'Arthritis',
       '(vertigo) Paroymsal  Positional Vertigo', 'Acne',
       'Urinary tract infection', 'Psoriasis', 'Impetigo']

    l2=[]
    for i in range(0,len(l1)):
        l2.append(0)

    df=pd.read_csv("C:\\Users\\Asus\\Desktop\\bot\\training.csv")
    DF= pd.read_csv('C:\\Users\\Asus\\Desktop\\bot\\training.csv', index_col='prognosis')
    #Replace the values in the imported file by pandas by the inbuilt function replace in pandas.

     df.replace({'prognosis':{'Fungal infection':0,'Allergy':1,'GERD':2,'Chronic cholestasis':3,'Drug
Reaction':4,
                    'Peptic ulcer diseae':5,'AIDS':6,'Diabetes ':7,'Gastroenteritis':8,'Bronchial
Asthma':9,'Hypertension ':10,
        'Migraine':11,'Cervical spondylosis':12,
                          'Paralysis (brain hemorrhage)':13,'Jaundice':14,'Malaria':15,'Chicken
pox':16,'Dengue':17,'Typhoid':18,'hepatitis A':19,
    X= df[l1]
    y = df[["prognosis"]]
    np.ravel(y)

    tr=pd.read_csv("C:\\Users\\Asus\\Desktop\\bot\\testing.csv")

    #Using inbuilt function replace in pandas for replacing the values

     tr.replace({'prognosis':{'Fungal infection':0,'Allergy':1,'GERD':2,'Chronic cholestasis':3,'Drug Reaction':4,
                    'Peptic ulcer diseae':5,'AIDS':6,'Diabetes ':7,'Gastroenteritis':8,'Bronchial Asthma':9,'Hypertension ':10,
        'Migraine':11,'Cervical spondylosis':12,
                          'Paralysis (brain hemorrhage)':13,'Jaundice':14,'Malaria':15,'Chicken pox':16,'Dengue':17,'Typhoid':18,'hepatitis A':19,
                  'Hepatitis B':20,'Hepatitis C':21,'Hepatitis D':22,'Hepatitis E':23,'Alcoholic hepatitis':24,'Tuberculosis':25,
                   'Common Cold':26,'Pneumonia':27,'Dimorphic hemmorhoids(piles)':28,'Heart attack':29,'Varicose veins':30,'Hypothyroidism':31,
        'Hyperthyroidism':32,'Hypoglycemia':33,'Osteoarthristis':34,'Arthritis':35,
                     '(vertigo) Paroymsal Positional Vertigo':36,'Acne':37,'Urinary tract infection':38,'Psoriasis':39,
        'Impetigo':40}},inplace=True)

    X_test= tr[l1]
    y_test = tr[["prognosis"]]
    np.ravel(y_test)

    clf3 = tree.DecisionTreeClassifier()
    clf3 = clf3.fit(X,y)
    y_pred=clf3.predict(X_test)

    psymptoms = [symp1,symp2,symp3,symp4,symp5]

    for k in range(0,len(l1)):
        for z in psymptoms:
            if(z==l1[k]):
                l2[k]=1

    inputtest = [l2]
    predict = clf3.predict(inputtest)

    predicted=predict[0]
```

```
h='no'
   for a in range(0,len(disease)):
       if(predicted == a):
           h='yes'
           break
   if (h=='yes'):
       return disease[a]
   else:
       return '0'
```

6.3 Screenshots

6.3.1 Homepage

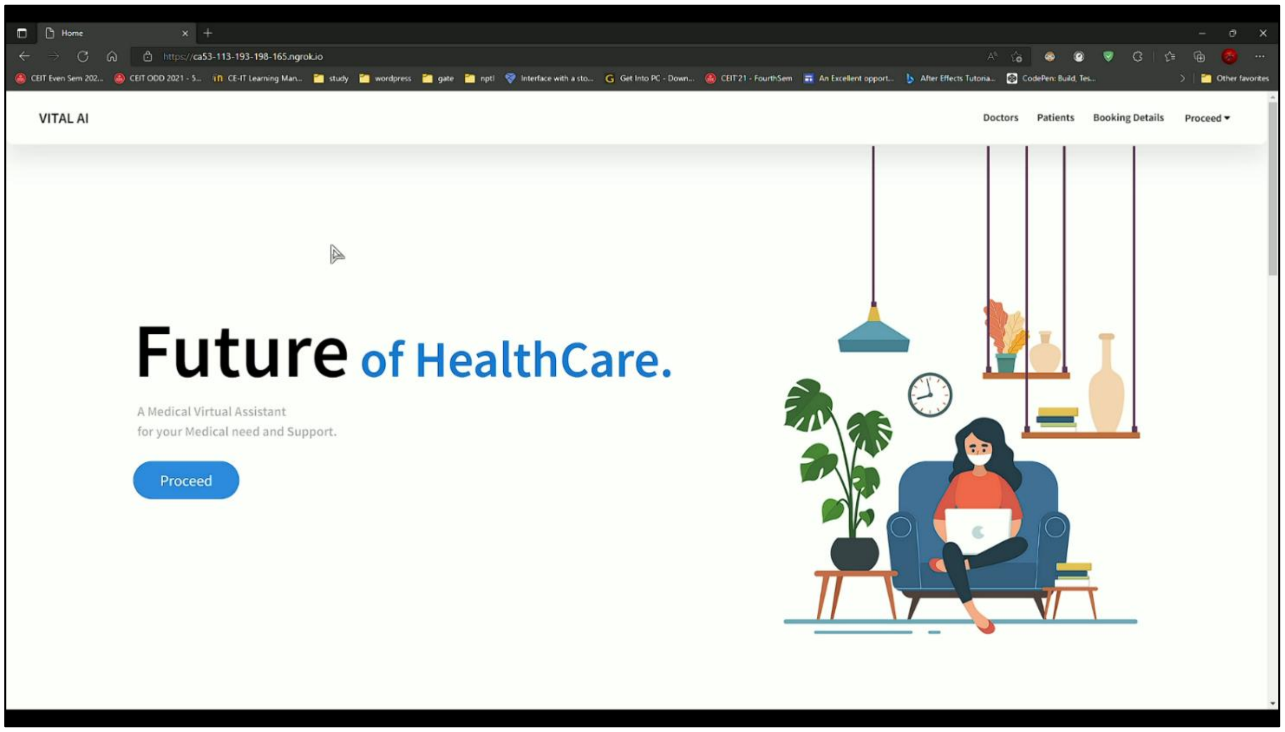


Figure: Home Page

6.3.2 Login Page

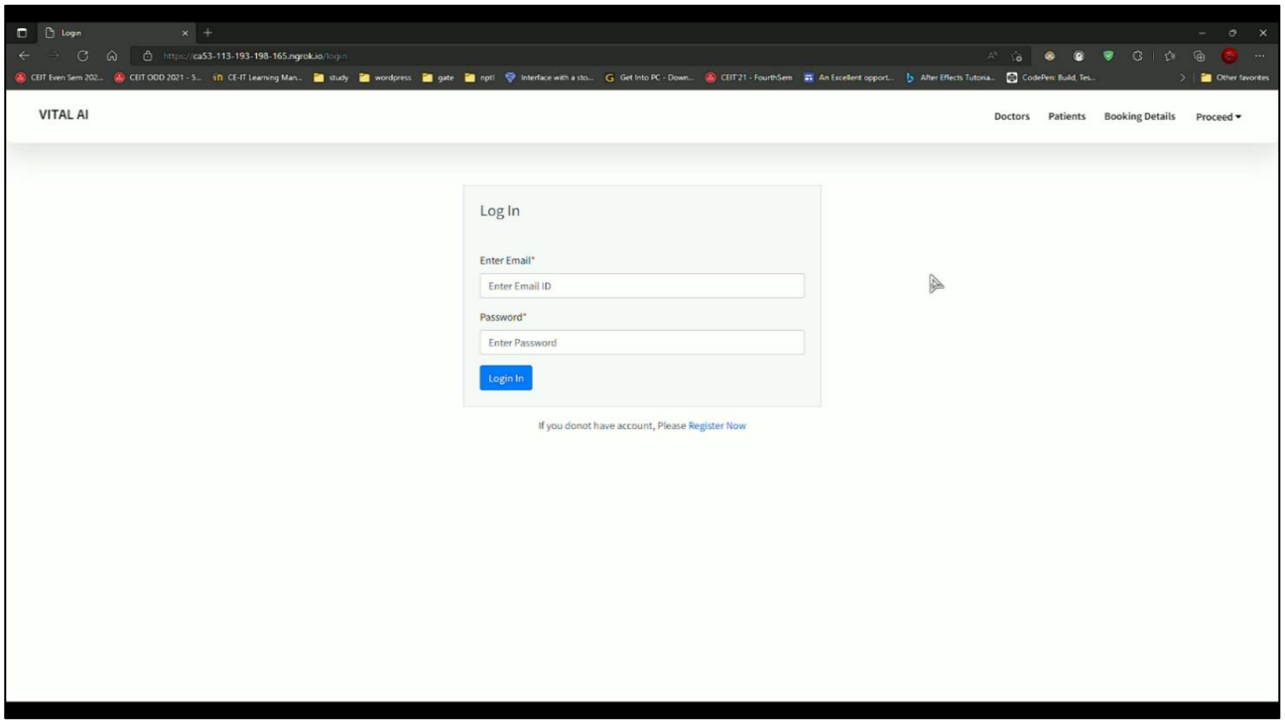


Figure: Login Page

6.3.3 Patient Dashboard

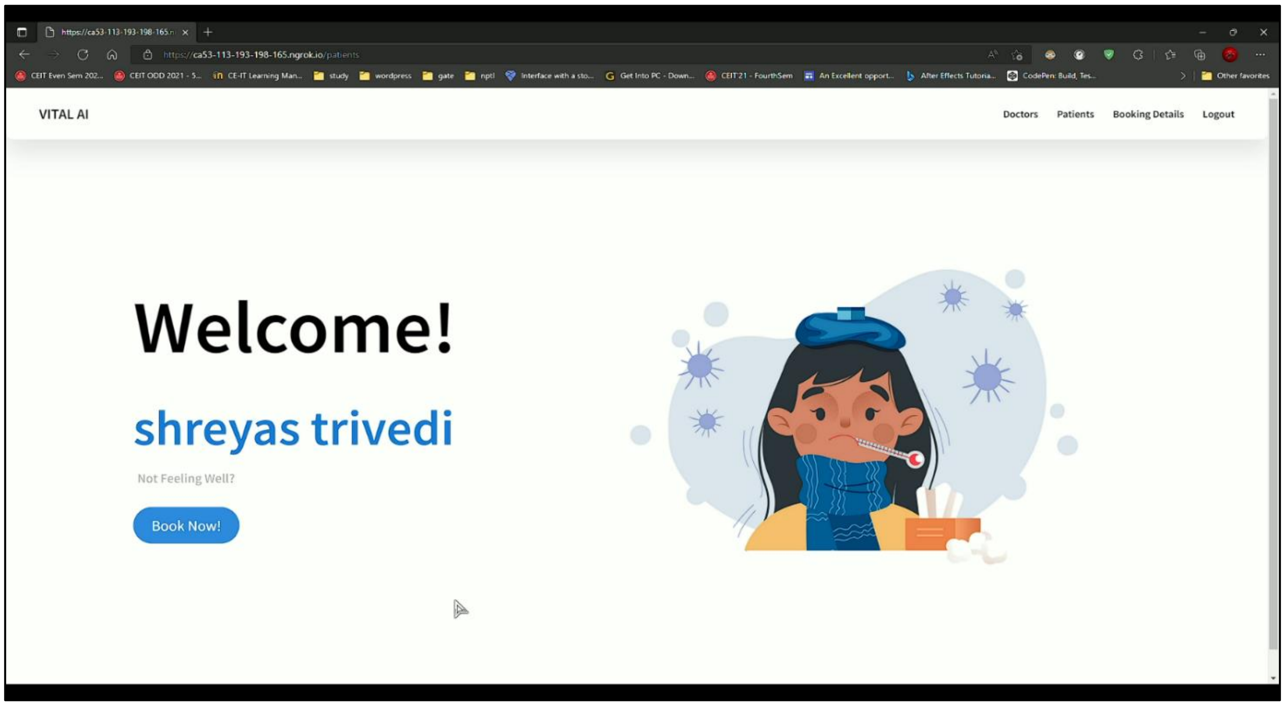
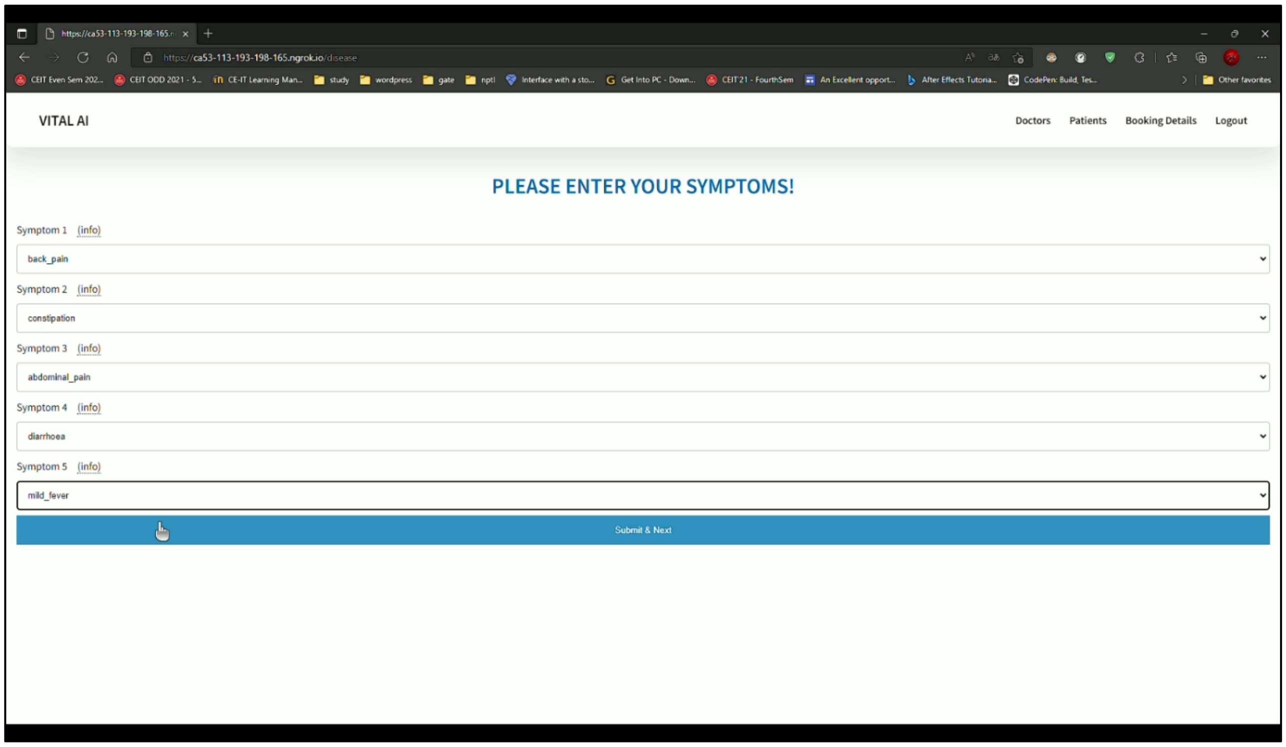


Figure: Patient Dashboard

6.3.4 Symptom Selection



PLEASE ENTER YOUR SYMPTOMS!

Symptom 1 (info)
back_pain

Symptom 2 (info)
constipation

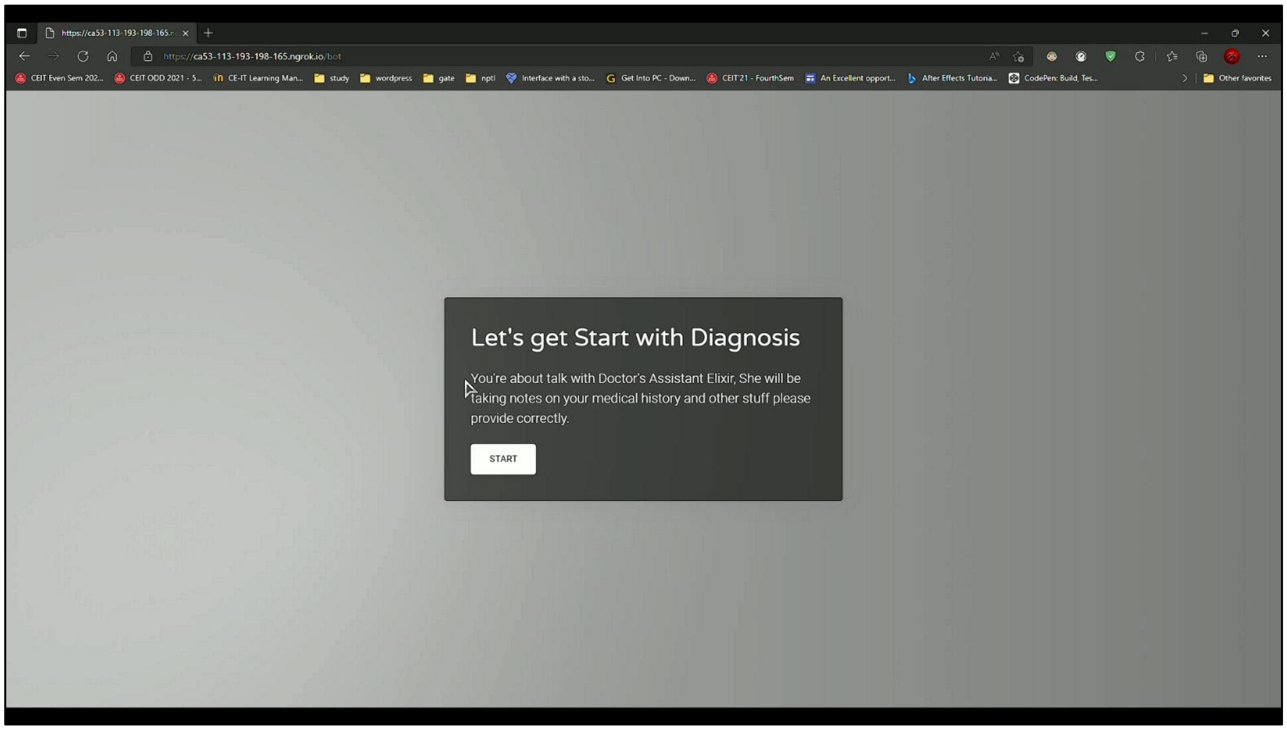
Symptom 3 (info)
abdominal_pain

Symptom 4 (info)
diarrhoea

Symptom 5 (info)
mild_fever

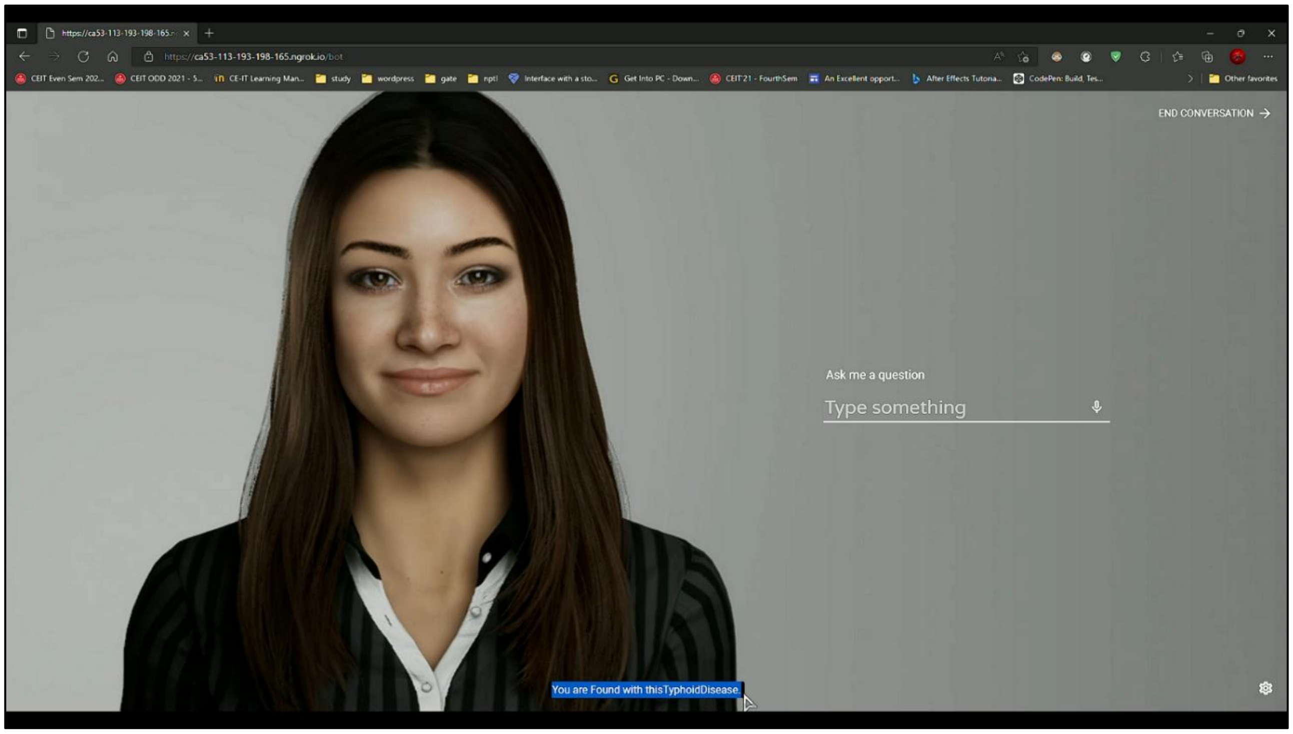
Figure: Symptom Selection

6.3.5 Digital Human


Let's get Start with Diagnosis

You're about talk with Doctor's Assistant Elixir, She will be taking notes on your medical history and other stuff please provide correctly.

Figure: Digital Human -01



You are Found with thisTyphoidDisease.

Figure: Digital Human -02

7. DATA DICTIONARY

7.1 User Table

Field Name	**Data Type**	**Field Length**	**Constrains**	**Description**
Id	Integer	11	Primary key	To give numbering
f_name	Varchar	50	Not null	For First Name of User
Lname	Varchar	50	Not null	For Last Name of User
email	Varchar	50	Unique	Email ID of the user
password	Varchar	1000	Not null	Password by the user

7.2 Patient Details

Field Name	**Data Type**	**Field Length**	**Constrains**	**Description**
Pid	Integer		Primary key	To give numbering
f_name	Varchar	50	Not null	For First Name of User

Lname	Varchar	50	Not null	For Last Name of User
email	Varchar	50	Unique	Email ID of the user
gender	Varchar	50	Not null	gender of the user
Age	Int	20	Not null	Age of the user
disease	Varchar	50	Not null	Disease Predicted
history	Varchar	100	Not null	Medical History

CONCLUSION

From these Capstone Stone Project We Came to learn about various Technologies and Framework like Machine learning, Flask, MYSQL to complete these projects. The Problem of staying updated with the technology is still hard hitting the nerve of these Small-Scale Healthcare Firm. By Performing these projects, we firmly believe this model can help these Firm to be in upfront lead in their Specific Domain.

ABOUT COLLEGE

U. V. Patel College of Engineering (GUNI-UVPCE) is situated in Ganpat Vidyanagar campus. It was established in September 1997 with the aim of providing educational opportunities to students. It was armed with the vision of educating and training young talented students of Gujarat in the field of Engineering and Technology so that they could meet the demands of Industries in Gujarat and across the globe.

The College is named after Shri Ugarchandbhai Varanasi Bhai Patel, a leading industrialist of Gujarat, for his generous support. It is a self-financed institute approved by All India Council for Technical Education (AICTE), New Delhi and the Commissionerate of Technical Education, Government of Gujarat. The College is spread over 25 acres of land and is a part of Ganpat Vidyanagar Campus. It has six ultra-modern buildings of architectural splendor, class rooms, tutorial rooms, seminar halls, offices, drawing hall, workshop, library, well equipped departmental laboratories and several computer laboratories with internet connectivity through 1 Gbps Fiber link, satellite link education center with two-way audio and one-way video link.

Figure: GUNI

The superior infrastructure of the Institute is conducive for learning, research, and training. The Institute offers various undergraduate programs, postgraduate programs, and Ph.D. programs. Placement plays a key role in shaping the future of the students, and keeping this in mind; the institute has forged healthy relations with the prominent industries. These tie-ups are mutually beneficial.

As part of this initiative, Incubation Centre/Start-up activities have also been developed. Running various Centers of Excellence (CoE) in collaboration with Industries: Bosch-Rexroth Centre for Automation Technology (Hydraulics & Pneumatics system, PLC & Mechatronics) Bosch Artisan Centre for Power tools in Electrical, Micro Focus Software University center for Internet of Things (IoT), Artificial Intelligence (AI) and Machine Learning, EC-Council University Center for Cyber Security Technologies.