Computer Science and Applications / Computer Information Systems / Software Engineering

Faculty of Prince Al-Hussein Bin Abdallah II For Information Technology

The Hashemite University, Zarqa-Jordan

HAYAT System

By

Mahmoud Ibrahim Mahmoud Abu Aqel (1936150)

Yazan Hasan Ahmad Qaswal (1932370)

Ala'a Saed Fathi Moh'd Younis (1933981)

Abdallah Najeh Faleh Thabet (1933987)

Maen Al Mutasem Mohammed Matalqah (1931978)

Supervised by

Dr. Khaled AL Makadmeh

Submitted in partial fulfillment of the requirements of B.Sc. Degree in Computer science

May / 2023

CERTIFICATE

It is hereby certified that the project titled < **HAYAT System** > submitted by the undersigned Students in partial fulfillment of the award of the Degree of Bachelor in Software Engineering embodies original work done by them under my supervision.

Project Supervisor: **Dr. Khaled AL Makadmeh**

Department of Software Engineering

Mahmoud Ibrahim Mahmoud Abu Aqel (1936150)

Yazan Hasan Ahmad Qaswal (1932370)

Ala'a Saed Fathi Moh'd Younis (1933981)

Abdallah Najeh Faleh Thabet (1933987)

Maen Al Mutasem Mohammed Matalqah (1931978)

ABSTRACT

Thyroid disease is a medical problem that prevents one's thyroid from producing enough hormones or producing too much. It can affect everyone: men, women, children, youths, and the elderly. The most common types of abnormal Thyroid functions are Hyperthyroidism or overproduction of hormones, and hypothyroidism or underproduction of hormones which are results of the malfunctioning of thyroid hormones. A lot of people here in Jordan are suffering from Thyroid related diseases, but many of them are unaware of having these diseases, and from here we came up with our project idea to empower users with information, raise awareness about thyroid and its related diseases, and the most important is to allow users to get an instant guidance on their thyroid status.

Our project < **Hayat** > is a Thyroid diseases Detector, Healthcare AI Application helps patients to get information regarding their Thyroid condition based on their provided symptoms. The system processes the symptoms provided by the user as input and gives the output as the probability of having Thyroid related diseases or not.

The major goal of our project < **Hayat** > is to make Thyroid diagnosis easier and categorize thyroid diseases based on users' blood test results into three categories: hyperthyroidism, hypothyroidism, and normal, by applying artificial intelligence machine learning prediction algorithms and data mining techniques.

TABLE OF CONTENTS

Page

TABLE OF CONTENTS

Page

• • • •

LIST OF FIGURES

Page

LIST OF FIGURES

Page

• • • •

LIST OF TABLES

Page

• • • •

CHAPTER 1: INTRODUCTION

This Chapter gives a scope description and an overview of Hayat Project, identifying the purposes and the aims of the project and the problem we're trying to solve and defining our project objectives and goals we are hoping to achieve.

1.1 Background

The Thyroid gland is a butterfly-shaped endocrine gland that is normally located in the lower front of the neck as shown below (1). It's one of the most important organs in our body. The thyroid's job is making thyroid hormones, which are secreted into the blood and then carried to every tissue in the body. Thyroid hormones speed up or slow down the metabolism of the body. Help the body using energy, stay warm and keep the brain, heart, muscles, and other organs working as they should. Two of the most prevalent disorders caused by thyroid gland irregularities are hyperthyroidism and hypothyroidism (2).

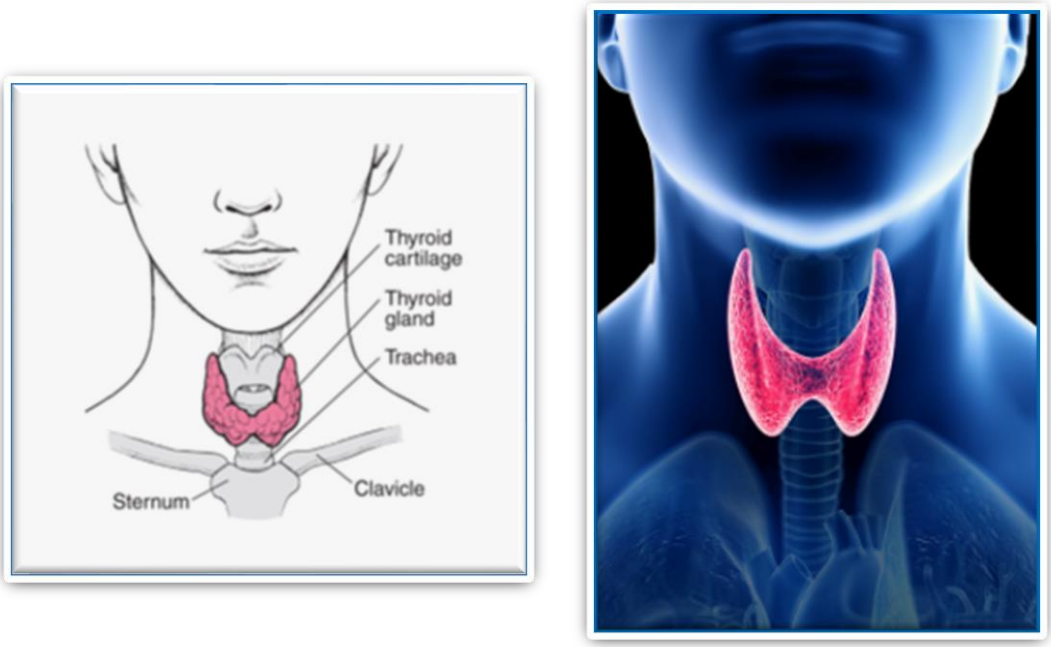


Figure No. 1 Thyroid Gland

Thyroid diseases have increased recently and patients are not aware of it because the symptoms of this disease are usually not easily detected at early stages, because it varies depending on the type, and hence we try to implement prediction of thyroid disease system as not much work has been done on thyroid here in Jordan. This project builds a system which helps predict a normal person about his/her possibility of having thyroid and the type of this thyroid if he/she has.

This system is completely done with help of machine learning algorithms and Code is written in Python programing language. The algorithm is first trained using the data set available from the repository and then tested on the dataset. Then the user enters his/her details and the algorithm starts running, according to the values entered by the user the algorithm predicts that the user is having thyroid or will be in future or not. This system will help doctors as well as individuals to have a possible disease diagnosed. And once a person predicts whether or not he/she can be diagnosed with thyroid disease, our system will provide the users with a lab service whom they can visit and do their thyroid tests there, and the results will be uploaded into the system as soon they are ready, and this will save users searching, effort and time. Moreover, each user will have his/her own information record which can be checked up and viewed when ever wanted.

1.2 Proposed Methodology

The algorithm is first trained with the help of the data base values. The algorithm learns that what inputs could give a positive output and what inputs would result in a negative output. Then the system will allow the user to enter their details - symptoms and the lab to inter the blood test results, then algorithm runs and predicts the result.

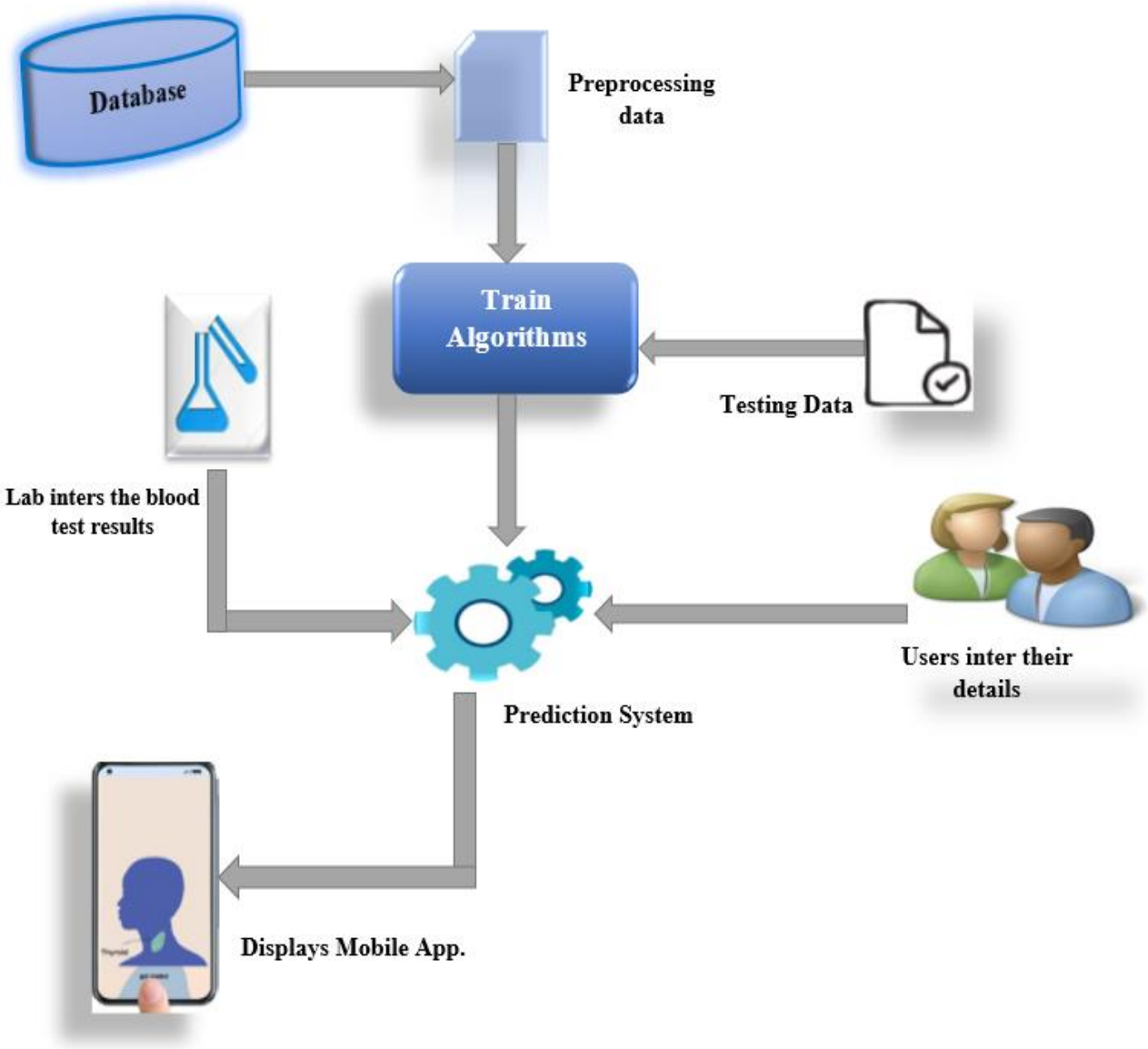


Figure No. 2 System Architecture

1.3 Problem Statement

Nowadays a lot of people suffering from thyroid diseases and some of them are even not aware of that. These diseases have many side effects and symptoms such as gain or loss weight, stress, muscle weakness, blood pressure and so on. If these diseases are detected in earlier stage, then patients can get an instant guidance about their thyroid situation and doctors can give them the proper treatment. So thyroid disease diagnosis involves analyzing these symptoms and detecting whether a disease persists in a body, and this is exactly the purpose of our system predicting the risk of obtaining thyroid disease at early stages with higher accuracy to protect and avoid the worst health condition of the patients, and in this regard, the machine learning plays a crucial role to detect the disease accurately. So, the problem is that we want to build a machine learning model to identify people affected by thyroid diseases, and to solve this problem we will have to analyze the data, do any required transformation and applying a machine learning algorithm to make the proper disease diagnosis. Further, if thyroid disease is present then algorithms are applied to classify the type of thyroid whether hyperthyroidism or hypothyroidism based on the patients' blood test results.

1.4 The Objectives of the Project

Thyroid disease is one of the major diseases that are spreading widely nowadays including Jordan as well, especially hypothyroidism and hyperthyroidism. The normal and traditional methods of thyroid diagnosis are thorough inspection and also various blood tests. Therefore, the major goals for our project Hayat are as follows:

1. The main goal is to recognize the disease at the early stages with a very high correctness by applying Machine learning techniques which play a major role in medical field for making a correct decision, proper disease diagnosis and also saves cost and time for the patient.
2. Making the Thyroid diagnosis easier and reduce the risk of misdiagnosis, which happens too often.

3. Prediction of Thyroid diseases at early stage can identify patients at risk of disease, and this will help doctors to take the appropriate measures to minimize the risk, provide medication at the right time, and avoid the worst health condition of the patients.

4. The major goal of our project < **Hayat** > is categorizing thyroid diseases based on users' symptoms and blood test results into three categories: hyperthyroidism, hypothyroidism, and normal.

5. Disease diagnosis plays a vital role and it is a necessary, so a lot of Healthcare issues can be solved efficiently by applying Machine Learning Technology and data mining techniques.

6. In our system we try to provide the users with a lab service whom they can visit and do their thyroid tests there, and the results will be uploaded into the system as soon they are ready, and this will save them searching, effort and time.

7. Moreover, the system provides a list of information records about its users.

8. This system gives a user-friendly environment and easy to use. As the system is based on a Mobile application, the user can use this system from anywhere and at any time.

1.5 Report Organization

The rest of the report is organized as follows. Chapter 2 introduces literature review. Chapter 3 lists the requirement engineering and analysis. The architecture and design components are described in Chapter 4. Chapter 5 presents the implementation plan and testing plan are described in Chapter 6. Chapter 7 presents the future work and concludes the report.

• • • •

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

This chapter gives a brief description of similar systems to our Project (**Hayat**) built on the same idea which is applying Machine Learning and data mining techniques to predict diseases.

2.2.1 Hypo Thyroid Prediction ML Project

The first system is **Hypo Thyroid Prediction ML**. This project helps patients to get information regarding their Thyroid condition based on their provided symptoms. In this chapter, we will discuss the overall features of this system, the similarities and differences between the two projects which will show the features that distinguish our project and give a brief evaluation of the another.

Hypo Thyroid Prediction ML is a Machine Learning Web Application using Data Mining Techniques to predict Hypothyroidism (underactive Thyroid) one of the main types of Thyroid related diseases and the most common one. It is a condition in which the body doesn't produce enough of important thyroid hormones, can cause gaining weight, trembling hands, muscle weakness, joint pain, dry skin and problems in vision. Hypo Thyroid Prediction ML has four User Interfaces as shown below:

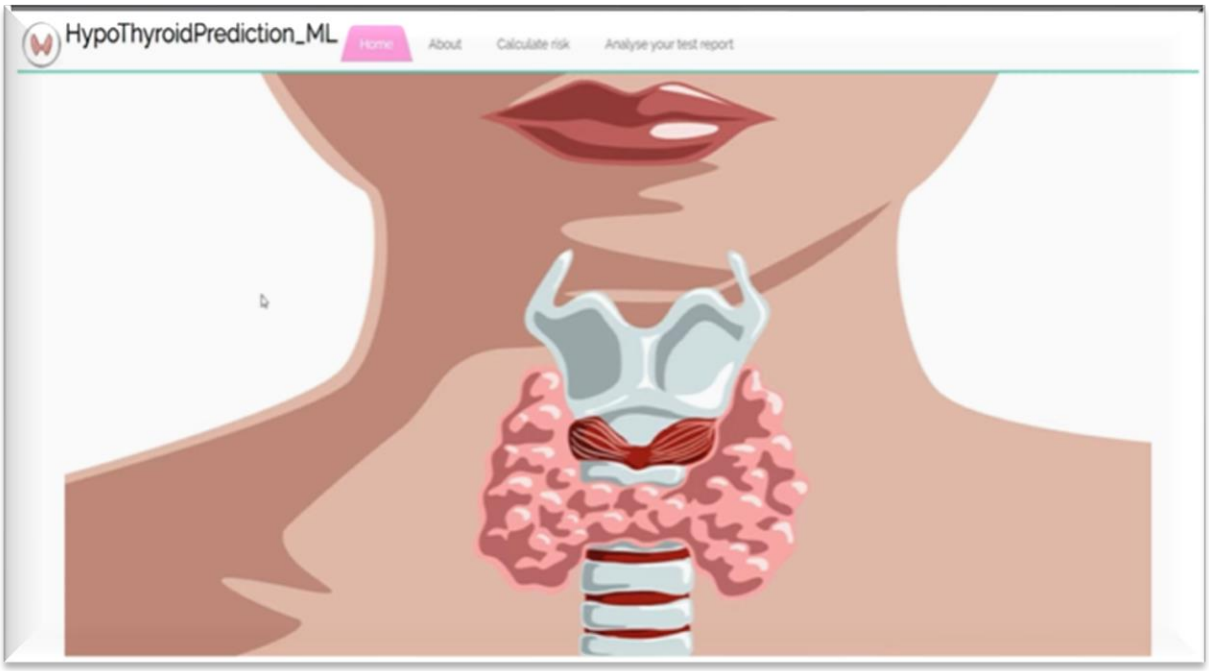


Figure No. 3 Hypo Thyroid Prediction ML System

The system allows users to get instant guidance on their thyroid status through an intelligent system online. Users can give in their various Thyroid related symptoms by answering a Yes or No questions Quiz when they click on calculate risk, then the system will predict and show up if the user doesn't have any risk or he/she at risk.

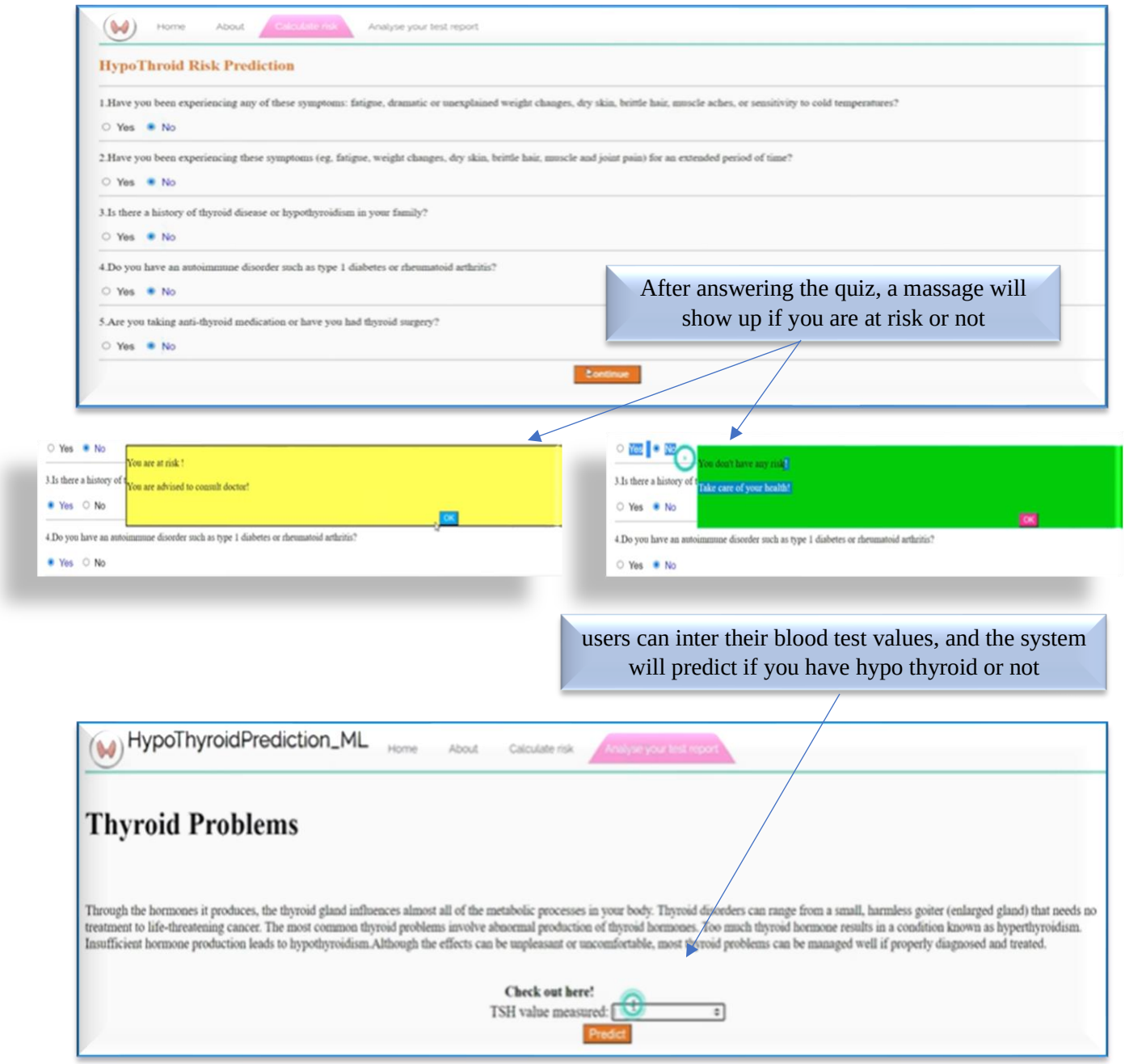


Figure No. 4 Some User Interfaces of Hypo Thyroid Prediction ML System

Additionally, users can inter their blood test results values when they click on Analyze your test reports, then the user inters his/her TSH value measured, and the system will predict and show up the analysis result whether Hypothyroid is negative or positive. Moreover, the system provides the users with Some information about Thyroid and its related diseases and the causes of Hypothyroid.

Ref.: **https://projectworlds.in/hypo-thyroid-disease-prediction-machine-learning-project/**

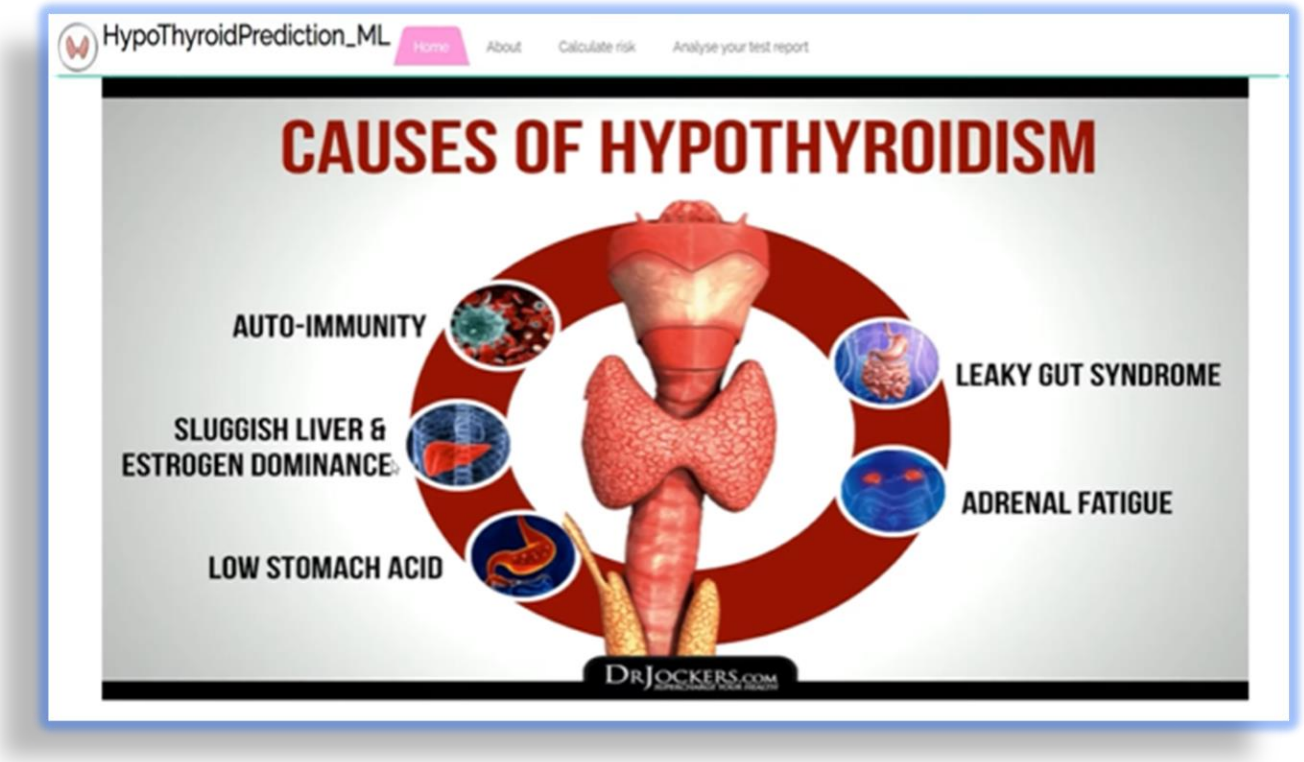


Figure No. 5 Hypo Thyroid Prediction ML System

❖ The following table shows the **similarities and differences** between our project Hayat and Hypo Thyroid Prediction ML Project:

Similarities	Differences
Both of the projects are built based on the same idea which is applying Machine Learning algorithms and data mining techniques to get the predication of Thyroid diseases.	However, our project is more comprehensive, as it aims to predict the two main types of Thyroid related diseases hyperthyroidism and hypothyroidism, unlike the Hypo Thyroid Project which focuses only on predicting hypothyroidism although a lot of people are suffering from hyperthyroidism.
Both of the systems depend on Thyroid Symptoms and blood tests of the users to predict the probability of having the disease and to classify the type of thyroid.	Our system also provides the users with a lab service whom they can visit and do their thyroid tests there, and the results will be uploaded into the system as soon they are ready, and this will save them searching, effort and time, unlike the hypo thyroid project which doesn't offer such a service.
The Code is written in Python in Both of systems.	—
Both of the systems are simple and easy to use.	—
Both of the systems have a great benefit to their users.	—

Table No. 1 The similarities and differences between Hayat Project and Hypo Thyroid Project

From the above, you can notice that our project provides more services to its intended users comparing to Hypo Thyroid Prediction project, and that's because we have gained a lot of knowledge from this project as it helps us to think of new ideas to help our users and enhance our project services.

• • • •

2.2.2 Multi Disease Predictor

The second system is **Multiple Disease Predictor**. it's a web application using Machine Learning and deep learning models to predict several diseases. This Web App was developed using Python Flask Web Framework. The models were trained on large Datasets to predict the diseases. The Multi Disease Predictor Web App can predict the following Diseases:

- Diabetes
- Breast Cancer
- Heart Disease
- Kidney Disease
- Liver Disease
- Malaria
- Pneumonia

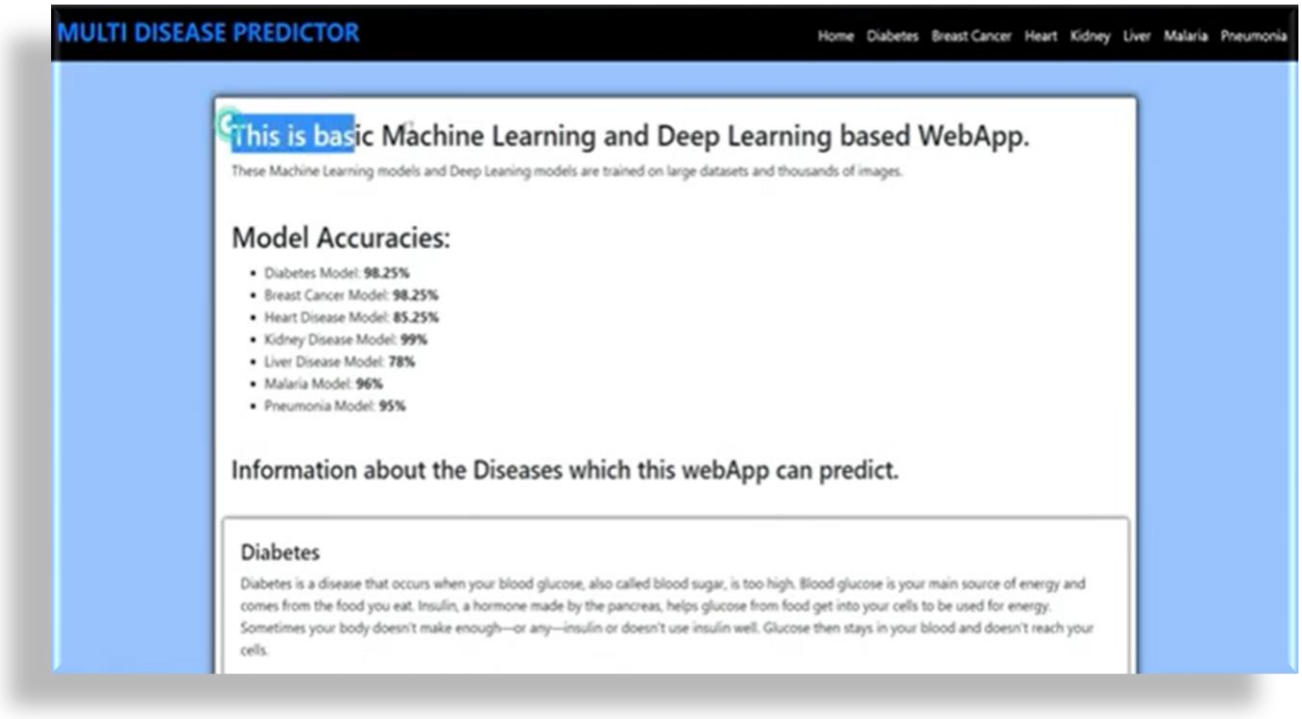


Figure No. 6 Multiple Disease Predictor Web App

The system allows the users to choose which disease they want to predict among the 7th diseases, then filling the disease predictor form with the related details and click on predict, after that, the system will predict and show up if the user has the disease or not as shows below:

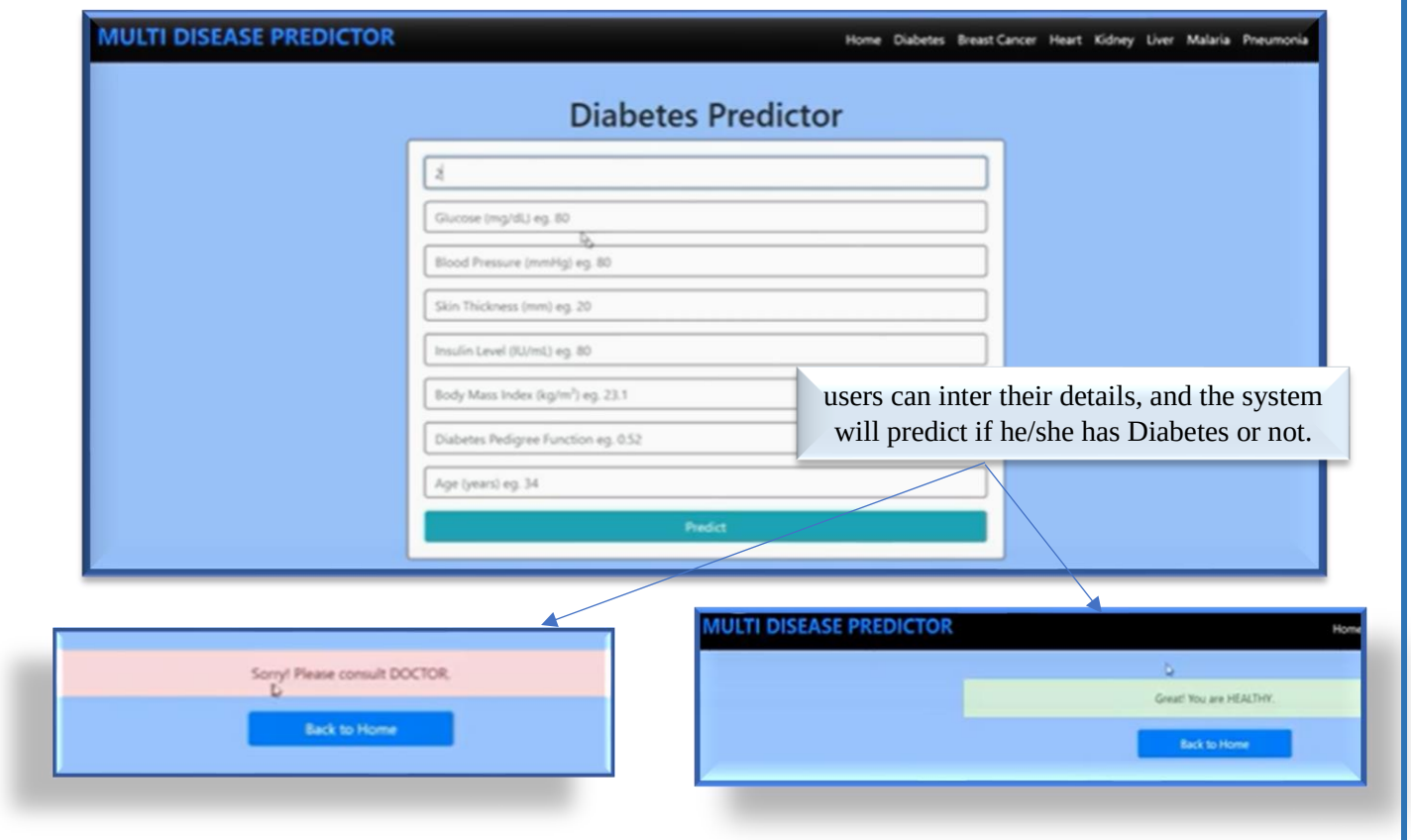


Figure No. 7 Diabetes Predictor User Interface

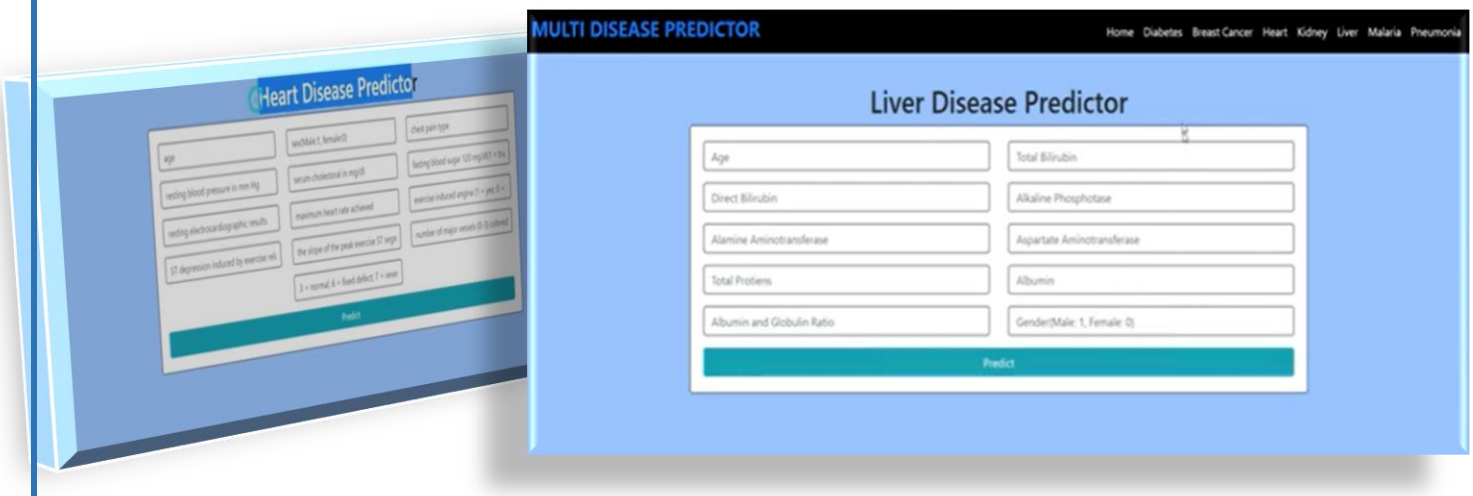


Figure No. 8 Liver and Heart Disease Predictor User Interfaces

Ref.: https://projectworlds.in/multiple-disease-prediction-using-machine-learning/

❖ The following table shows the **similarities and differences** between our project Hayat and Multiple Disease Predictor Web App:

Similarities	Differences
Both of the projects are built based on the same idea which is applying Machine Learning algorithms and data mining techniques to predict diseases.	However, our project aims to predict the two main types of Thyroid related diseases hyperthyroidism and hypothyroidism, unlike the Multiple Disease Predictor Web App which aims to predict several diseases: Diabetes, Breast Cancer, Heart Disease, Kidney Disease, Liver Disease, Malaria and Pneumonia
Both of the systems are depending on training the algorithms on Dataset to predict the diseases.	Multiple Disease Predictor system applying both Machine Learning and deep learning models to predict several diseases, However, our system depending on Deep learning models only.
The Code is written in Python in Both of systems.	Our system is aimed to be a Mobile Application, on the other hand, Multiple Predictor is developed to be a Web Application.
Both of the systems are simple and easy to use.	—
Both of the systems have a great benefit to their users.	—

Table No. 2 The similarities and differences between Hayat Project and Multiple Disease Predictor Web App.

From the above, you can notice that Multiple Disease Predictor Web App. provides more services to its intended users comparing to our project, as it predicting 7 kinds of diseases which will be very helpful to patients and to healthcare providers as well, and in the future, we are hoping to expand our project and use more ML Algorithms to predict more diseases than Thyroid diseases to help and benefit more users.

• • • •

CHAPTER 3: SOFTWARE REQUIREMENTS

3.1 Requirements Elicitation

This chapter specifies the software requirement specifications (SRS), both functional and nonfunctional for the **Hayat** Application. This document will illustrate and provide the overall description, purpose, usage, and detail the development concepts and features of this system. Additionally, this document will serve as a tool for the development team and software testers involved in this project, to better understand the full scope of this application. The purpose of this software application is to provide a simple, easy to use system which helps to predict and classify Thyroid related diseases.

3.2 Functional User Requirements

To present this system in an easy clear mode, we decided to divide the work based on the functions of the system according to actors' roles. Therefore, the system is composed of three users as we will see below. We used in this project the **Draw IO UML Tool** in modeling most of the requested diagrams.

The Stakeholders (Actors) of the system and their use case modeling are as follows:

Patient: Can login and give in his/her symptoms with the help of which the application would predict the kind of thyroid disease he/she has, and give him/her instructions for doing the necessary blood tests and advice to consult a doctor if needed.

Laboratory: Providing the blood test services, and generates the Lab report conclusion based on the tests' results.

The System Administrator: will be managing every activity of the system for its intended purposes, and he/she is the highest privileged user who can access the system.

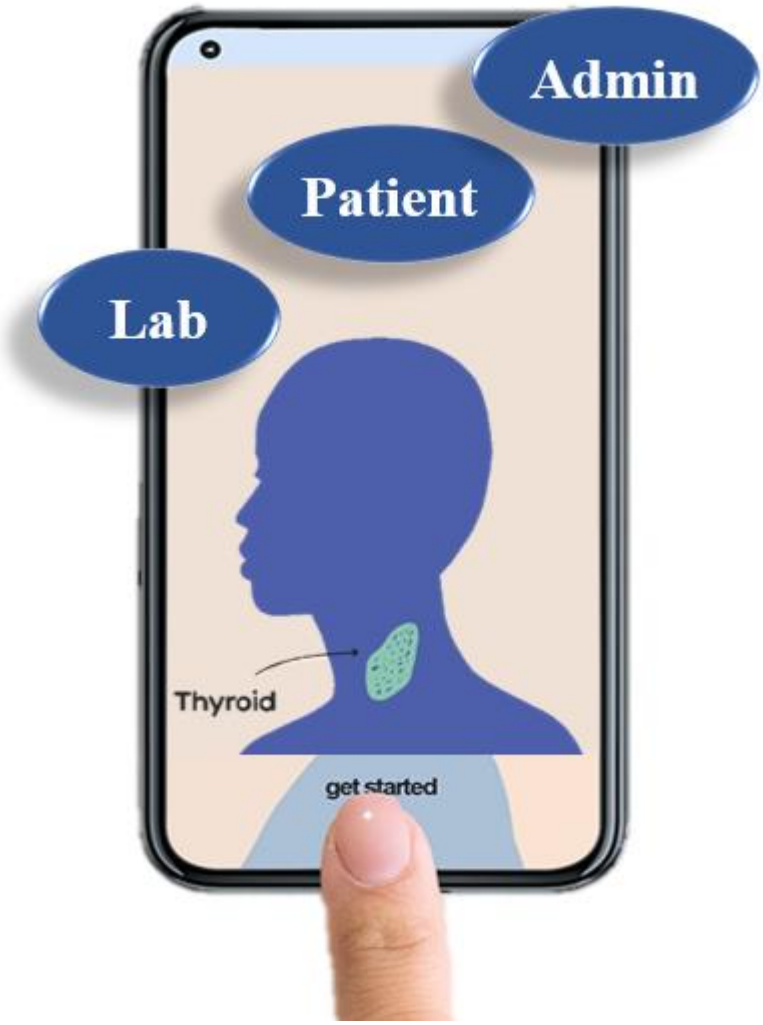


Figure No. 9 Stakeholders (Actors)

3.2.1 The Use Case Model (Script & Diagram)

In this section we will describe in details the use case diagram of Hayat Application system which represents the functionality of the system from the user's point of view, and we will divide the functions of the system according to actors' roles. First, we will start with the Patient's functions, then Admin's responsibilities and finally with lab's function.

3.2.1.1 The Patient

Use case name	**Login**
Actors	**Patient, Laboratory, Admin**
Pre-Condition	System must be connected to the network
Basic flow	1- Inter username which will be the National Id No. 2- Validate National Id No. 3- Home Page is displayed if username is valid.
Alternative flow	[Invalid National Id No.] At step 3 if the user name (National Id No.) is invalid, Error Message is displayed. The actor can choose to either return to the begginig of the basic flow and inter the correct National Id No. or cancel the login .
Special requirments	None
Post-Conditio	If the use case was successfull, the actor is now logged into the system, if not the system state is unchanged.

Table No. 3 The Patient Login Use case

Use case name	**Logout**
Actors	**Patient, Laboratory, Admin**
Pre-Condition	System must be connected to the network
Basic flow	1. Patient will choose to logout of the system 2. Click on Logout tab 3. The actor will be no longer connectected to the system
Alternative flow	None
Special requirments	None
Post-Condition	If the use case was successfull, the actor will be disconnected to the system, if not the system state is unchanged.

Table No. 4 The Logout Use case

Use case name	**Check Ansewrs**
Actors	**Patient**
Pre-Condition	Actor must be connected to the system, and answer the questionnare.
Basic flow	1- Click on Check Ansewrs tab 2- Once the actor click at the tab, a new page will be displayed, and the system will predict and show up the result and the probability of having Thyroid releated diseases or not. 3- If the actor has the probability of having Thyroid diseases he/she must do some blood tests to make sure if he/she has the disease. 4- The address of the labortory will be mentioned for helping the users, with the available timings for doing the blood tests.
Alternative flow	None
Special requirments	None
Post-Condition	If the use case was succesful, the actor viewed his/her result and knew if he/she has the disease, and what to do next, else the actor dosen't.

Table No. 5 Check Ansewrs Use case

Use case name	**Thyroid Questionnare**
Actors	**Patient**
Pre-Condition	Actor must be connected to the network
Basic flow	1- Click on Thyroid Questionnare tab 2- System shows a page with a Thyroid questionnare; consist of Yes or No questions questions releated to Thyroid symptoms & Family history if any, the actor shall be able to view and choose the symptoms from the questionnare. 3- Once the actor provides the requested answers, He/she shall click on check answers.
Alternative flow	None
Special requirments	None
Post-Condition	If the use case was succesful, the actor shall complete answering the questionnare.

Table No. 6 The Questionnare Use case

Use case name	**View Test Results**
Actors	**Patient**
Pre-Condition	Actor must be connected to the network, and did the blood tests at the Laboratory
Basic flow	1- Include (Login) 2- Actor clicks at View Patient's Record tab 3- System shows new page with the blood tests result and the type of thyroid diseases he/she has, classified either Hyperthyroidism or hypothyroidism else the result is normal.
Alternative flow	None
Special requirments	None
Post-Condition	If the use case was succesful, the actor shall view his/her result whether it's Hyperthyroidism, hypothyroidism or normal.

Table No. 7 View Test Results Use case

3.2.1.2 The System Administrator

Use case name	**Manage Patients**
Actors	**Admin**
Pre-Condition	Actor must be connected to the network and logged into the system
Basic flow	1- Include (Login) 2- Click on Manage Patients tab 3- System will show up the Patient's list 4- View Patient's list 5- Filter and Search Patients [Add Patient]
Alternative flow	At step 4, Admin can add a new patient with his/her information to the system. [Edit Patient] At step 5, Admin can change Patient's information. [Delete Patient] At step 5, Admin can delete a patient from the system if needed .
Special Requirments	None
Post-Condition	If the use case was succesful, The Actor viewed and managed the patient's info.

Table No. 8 Manage Patients Use case

Use case name	**Manage Tests**
Actors	**Admin**
Pre-Condition	Actor must be connected to the network, and logged into the system
Basic flow	1- Include (Login) 2- Click on Manage Tests tab 3- System Shows up the list of tests 4- Admin can view details of the tests
Alternative flow	[Add Test] At step 3, Admin can add a test into the system. [Delete Test] At step 3, Admin can delete a test from the system.
Special Requirments	None
Post-Condition	If the use case was succesful, The Actor viewed and managed the tests.

Table No. 9 Manage Tests Use case

Use case name	**Generate Patients Records**
Actors	**Admin**
Pre-Condition	Actor must be connected to the network and logged into the system
Basic flow	1- Include (Login) 2- Click on generate Patients Records tab 3- Admin shall be able to generate a new patients record consist of patient details (name. National Id, gender, age, questionnare result, blood test date, blood test result and Thyroid type if any) 4- System will show up the list of Patients with thier Records 5- Filter and Search Patients
Alternative flow	[Edit Patient record] At step 5, Admin can change Patient's information [Delete Patient record] At step 5, Admin can delete a patient's record from the system if needed.
Special Requirments	None
Post-Condition	If the use case was succesful, The Actor generated the patient record managed the her/his info.

Table No. 10 Generate Patients Records Use case

Use case name	**Register**
Actors	**System Administrator**
Pre-Condition	System must be connected to the network
Basic flow	1. Click on Register tab 2. Inter name 3. Inter E-mail & Phone No. 4. Inter username 5. Inter password 6. Confirm password 7. The system will verify and validate the actor's personal data 8. A message of vertification code will be sent to the actor 9. Inter the vertification code 10. Your account has been created
Alternative flow	[Wrong Vertification Code] At step 9 if the actor intered incorrect Vertification Code, an error Message is displayed with Wrong Vertification Code The actor can either click on resend Code or cancel the Register.
Special requirments	None
Post-Condition	If the use case was successfull, the actor's acccount has been created in the system, if not the system state is unchanged.

Table No. 11 Register Use case

3.2.1.3 The Laboratory

Use case name	**Fill out Lab Report**
Actors	**Lab**
Pre-Condition	Actor must be connected to the network and logged into the system and lab test results are available
Basic flow	1- Include (Login) 2- Actor selects lab report form 3- System prompts patient National Id. 4- Actor enters patient National Id. 5- System prompts to select lab test category 6- Actor selects category and enters test results 7- Actor selects add lab test record 8- System display successfully added message
Alternative flow	None
Special Requirments	None
Post-Condition	If the use case was succesful, The Actor enters the test results.

Table No. 12 Fill out Lab Report Use case

The Use Case Diagram of Hayat System is as shown below:

Figure No. 10 The Use Case Diagram of Hayat System

3.3 Non-Functional User Requirements

The Non-Functional attributes of the project are illustrated under various sections below:

3.3.1 Performance

- User-interface: The user-interface screen shall respond within 5 seconds.
- The system should be reliable and scalable.

3.3.2 Maintainability

The system is developed in such a manner that its functionality can be enhanced to support further development in the system.

3.3.3 Safety & Security

- The system is secure as the user has to enter each time his/her login (National Id No.) to prevent unauthorized access.
- The database should be secured from SQL injection to prevent leak or loss information.

3.3.4 Usability

The system offers an easy-to-use interface and the new users shall get used too fast as possible.

3.3.5 Availability

The system is available for use at 24 hours a day, 7 days a week.

• • • •

CHAPTER 4: SOFTWARE ARCHITECTURE & DESIGN

4.1 Overview

In This Chapter We Well Explain All Components of Our App and How They Are Related To Each Other Through Some of The Diagrams And A Full Description Of The Interfaces of the App.

4.2 Software Architecture

A system architecture is a conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system

4.2.1 Sequence Diagram

Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration.

These are the sequence diagrams for "Hayat" project application:

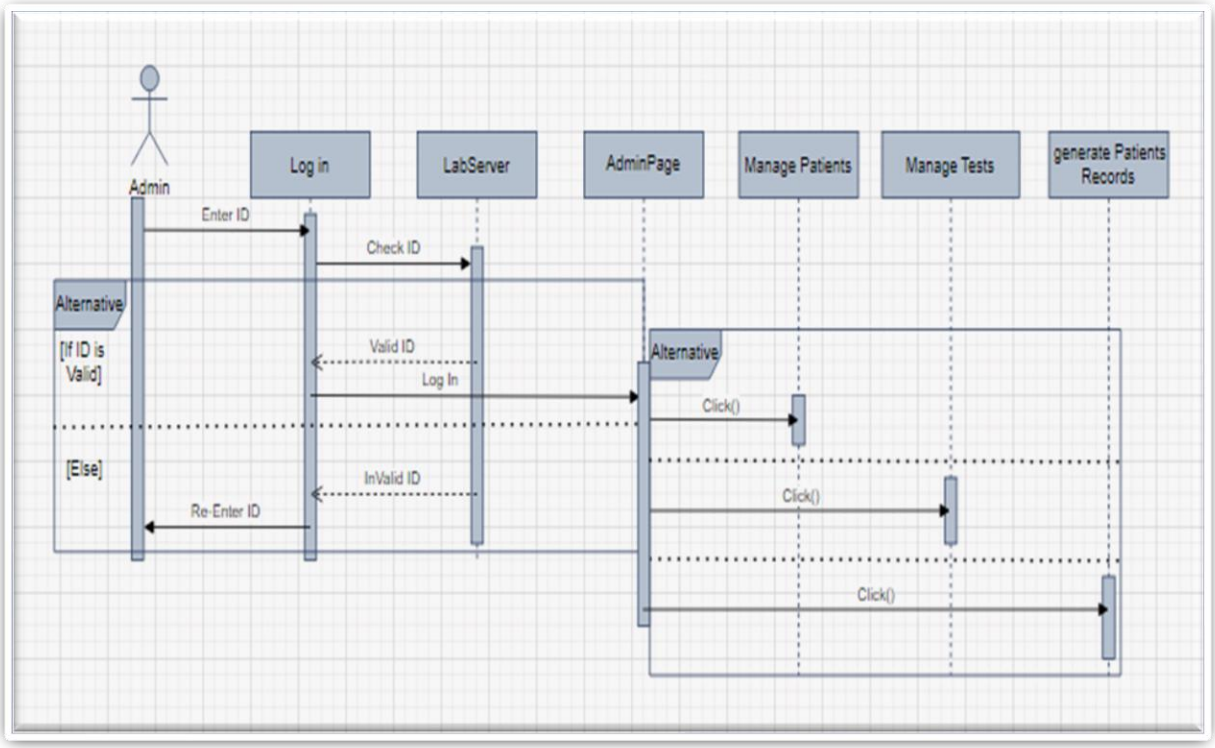


Figure No. 11 The Sequence Diagram for The Admin

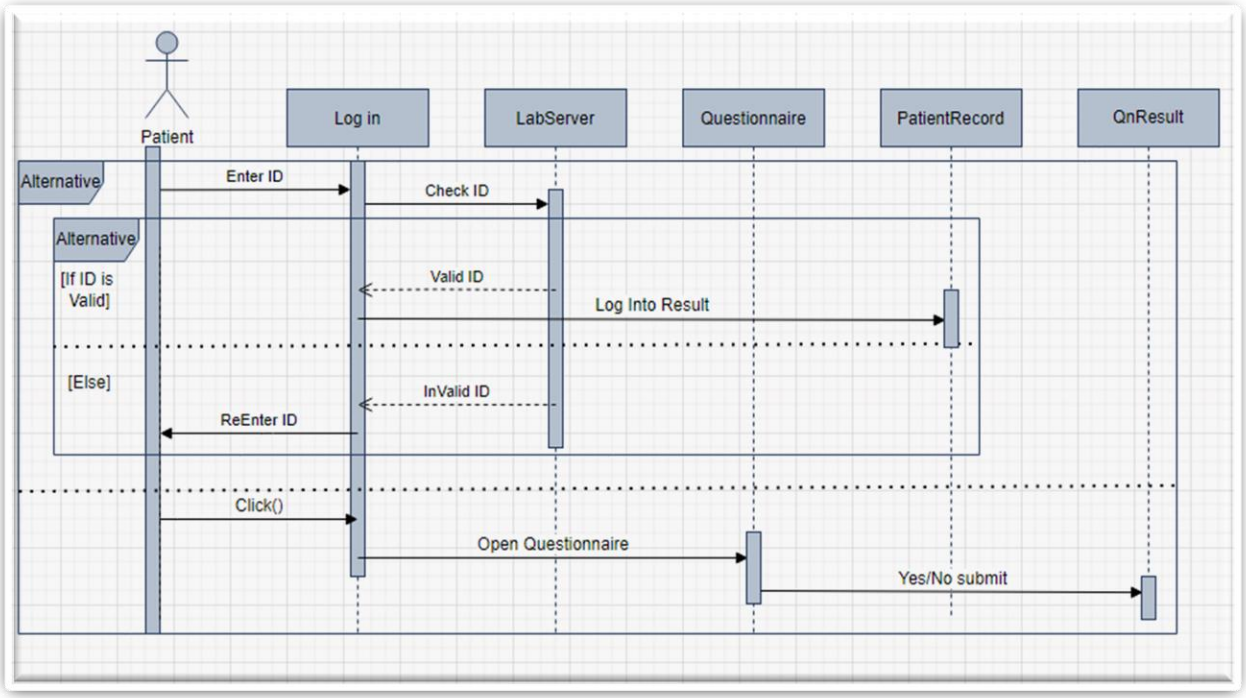


Figure No. 12 The Sequence Diagram for The Patient

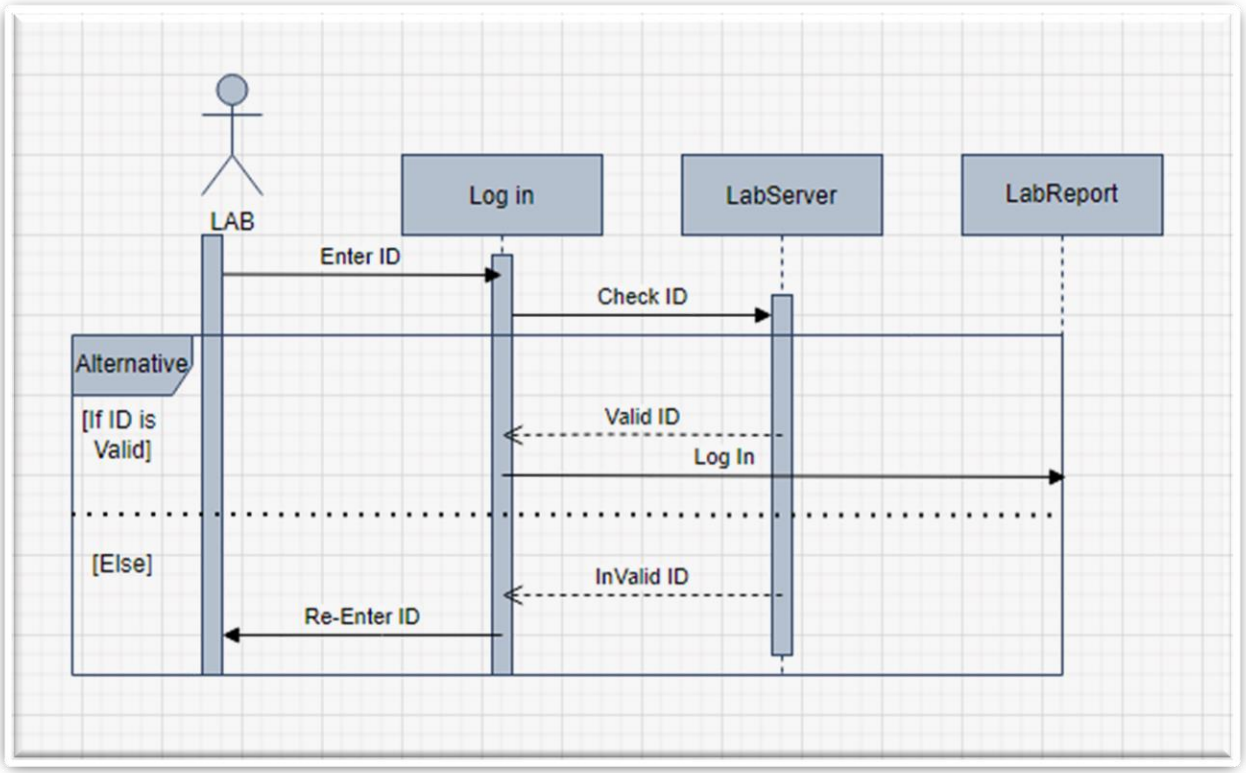


Figure No.13 The Sequence Diagram for The Lab

4.2.2 Activity Diagram

An activity diagram visually presents a series of actions or flow of control in a system similar to a flowchart or a data flow diagram. Activity diagrams are often used in business process modeling. They can also describe the steps in a use case diagram. Activities modeled can be sequential and concurrent.

This is the activity diagram for the "Hayat" project application.

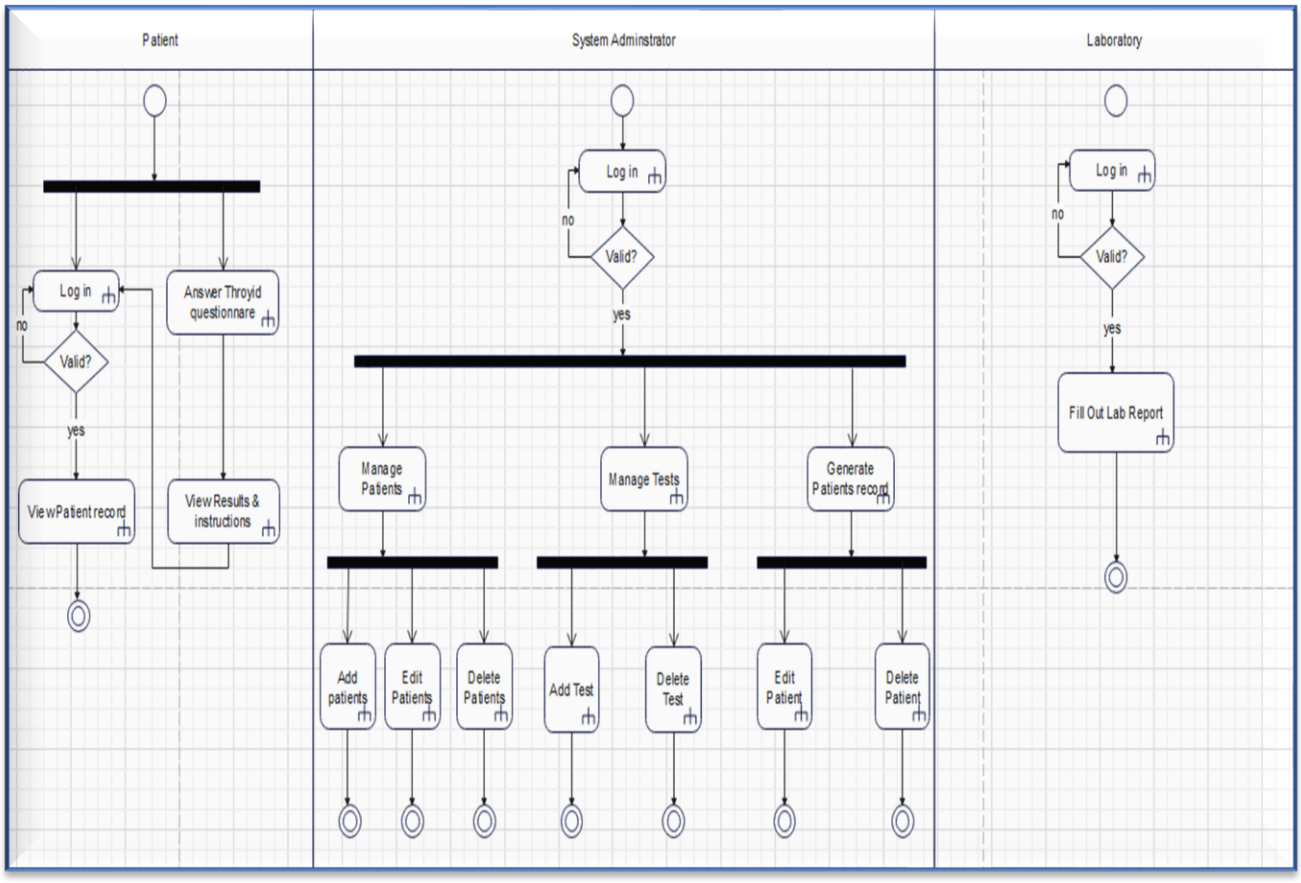


Figure No. 14 The Activity Diagram of Hayat System

4.2.3 Class Diagram

The class diagram is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

the class diagram provides a basic notation for other structure diagrams prescribed by UML.

This is the class diagram for "Hayat" project application:

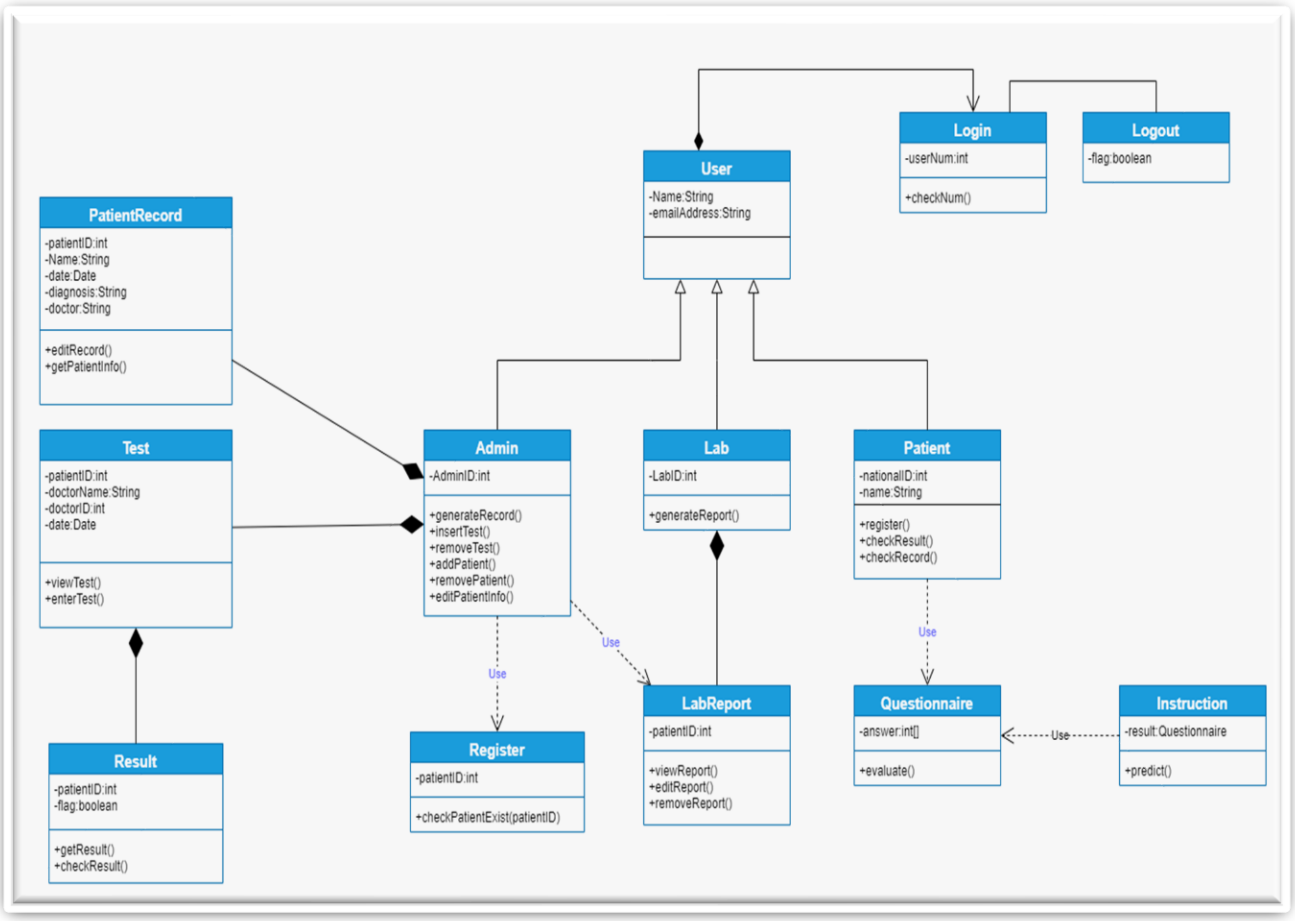


Figure No. 15 The Class Diagram of Hayat System

• • • •

4.2.5 Interface Specification

User must inter his/her username which will be the National Id No.the home page will be displayed if the username is valid, if not Error Message is displayed. Then he/she can press the questionnaire tab.

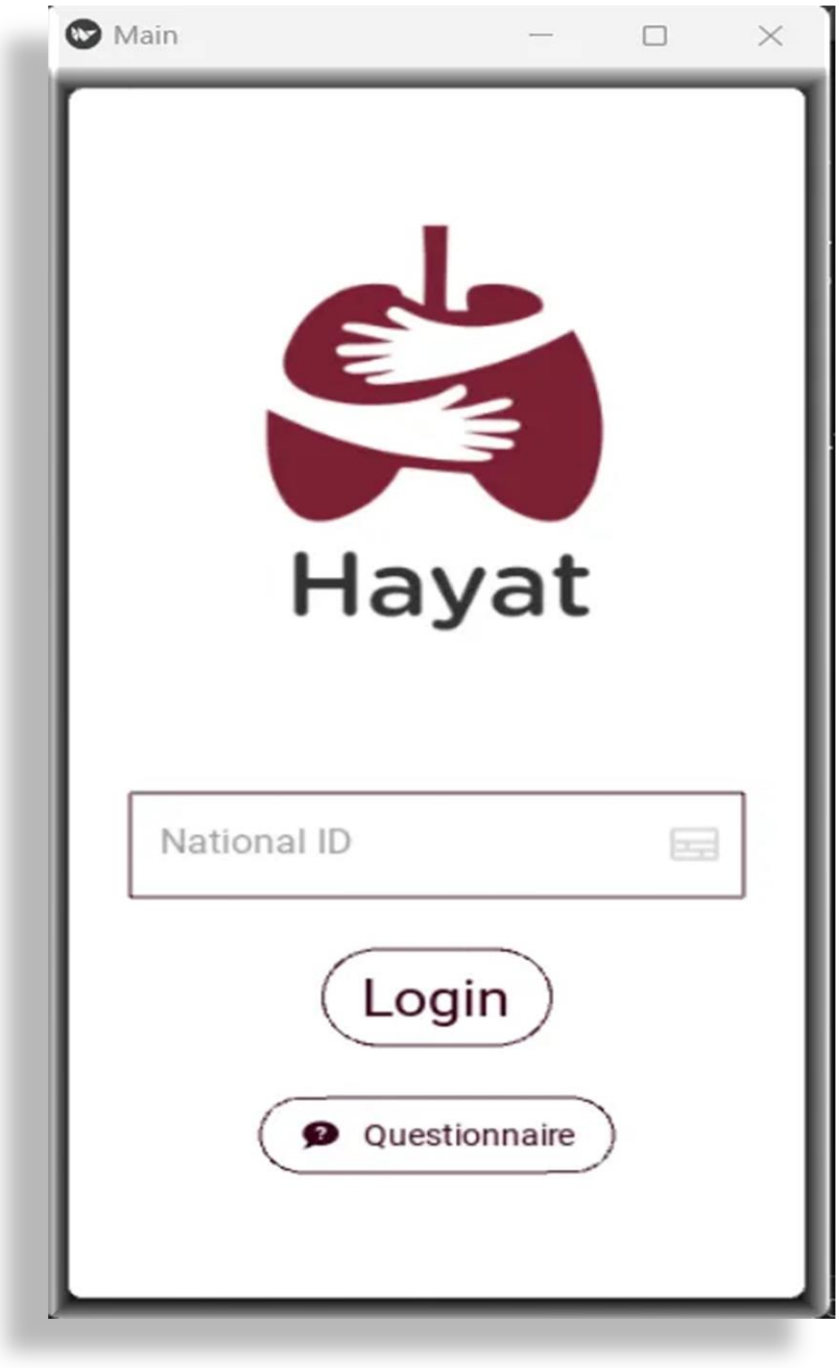


Figure No. 17 Login Interface for Hayat Application

The questionnaire page consists of Yes or No questions related to Thyroid symptoms, the user must answer the questions and press check answers, If the actor has the probability of having Thyroid diseases he/she must do some blood tests to make sure if he/she has the disease.and a page will show the address of the laboratory and its available timings for doing the blood test.

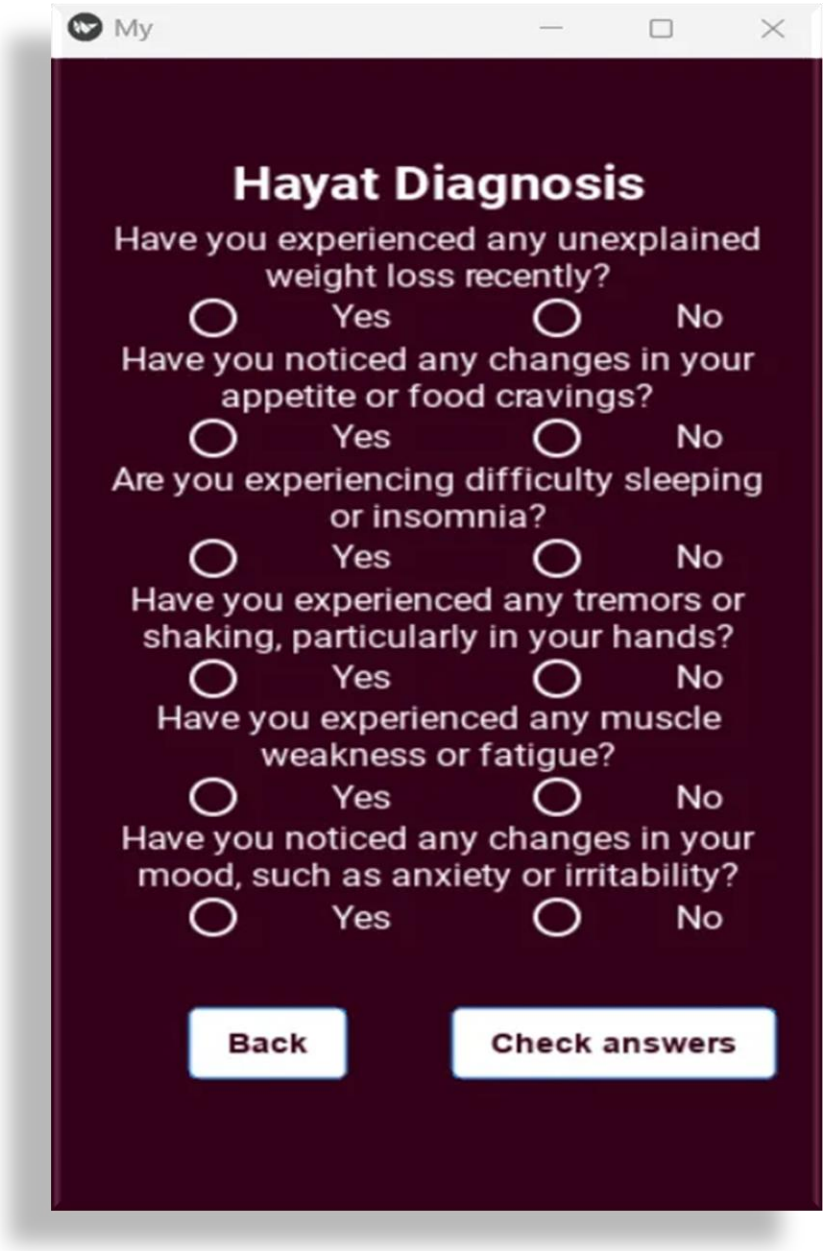


Figure No. 18 Questionnaire Interface for Hayat Application

The user can check his/her test results after doing the blood tests, the system shows the page with the blood tests result and the type of thyroid diseases he/she has classified either Hyperthyroidism or hypothyroidism else the result is normal.

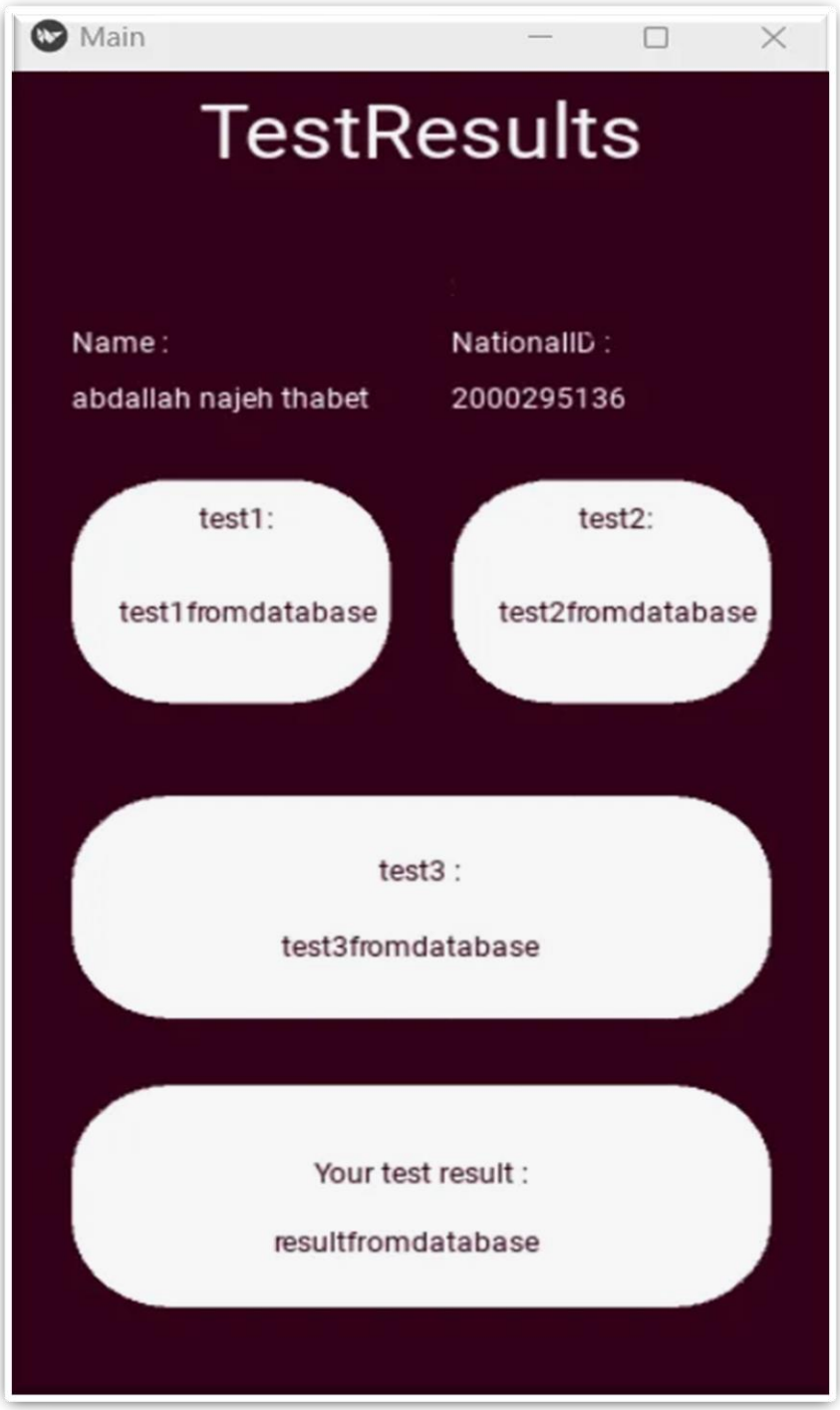


Figure No. 19 Test Results Interface for Hayat Application

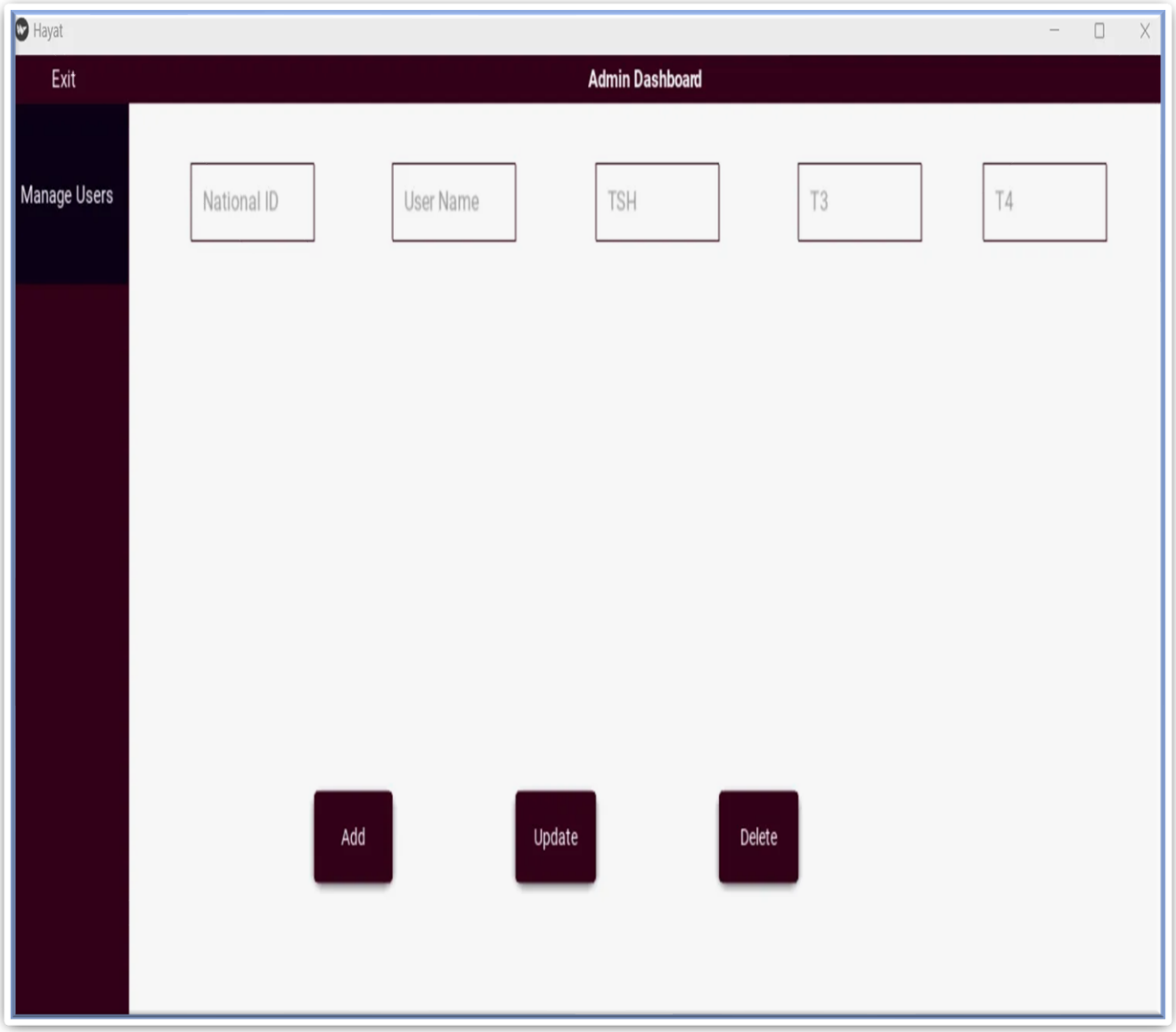


Figure No. 20 Admin Dashboard

• • • •

CHAPTER 5: IMPLEMENTATION PLAN

5.1 Methodology

Our project < **Hayat** > is a Thyroid diseases Detector, Healthcare AI Mobile Application helps patients to get information regarding their Thyroid condition based on their provided symptoms. The system processes the symptoms provided by the user as input and gives the output as the probability of having Thyroid related diseases and categorize the diseases based on users' blood test results into three categories: hyperthyroidism, hypothyroidism, and normal, by applying **artificial intelligence machine learning prediction algorithms and data mining techniques,** and Code is written in Python programing language. The algorithm is first trained using the data set available from the repository "Thyroid Disease Data Set Kaggle", The algorithm learns what inputs could give a positive output and what inputs would result in a negative output. Then the system will allow the user to enter their details -symptoms and the lab to inter the blood test results, then algorithms start running, according to the values entered by the user and predict if the user is having thyroid or will be in future or not.

This App was developed using **Kivy the Open Source Python App Framework**. The models were trained on large Datasets to predict the diseases from **Kaggle Dataset.**

The initial stage of our project was selecting data samples suit our project aims for experiments as shown in Figure 21. Experiments are performed with using several machines learning and deep learning models.

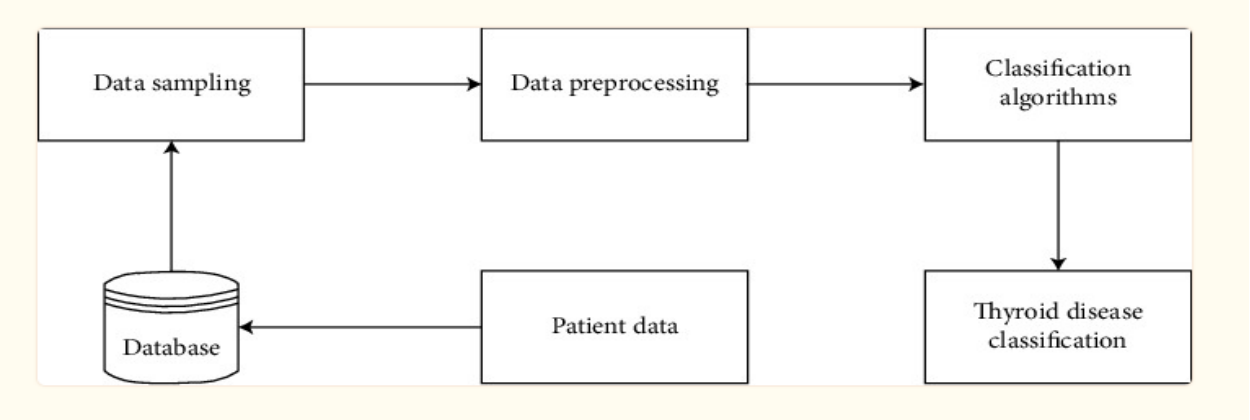


Figure 21 System Diagram

The dataset contains the attributes of age, six, and some thyroid markers like TSH, T3, and T4U to categorize the disease as we can see in table 13.

Attribute	Description
age	age of the patient
sex	sex patient identifies
on thyroxine	whether patient is on thyroxine
query on thyroxine	whether patient is on thyroxine
on antithyroid meds	whether the patient is on antithyroid meds
sick	whether patient is sick
pregnant	whether patient is pregnant
thyroid_surgery	whether patient has undergone thyroid surgery
query_hypothyroid	whether the patient believes they have hypothyroid
query hyperthyroid	whether the patient believes they have hyperthyroid
lithium	whether patient * lithium
goiter	whether patient has goitre
tumor	whether patient has tumor
hypopituitary	whether patient * hyperpituitary gland
psych	whether patient * psych
TSH_measured	whether TSH was measured in the blood
T3_measured	whether T3 was measured in the blood
TT4_measured	whether TT4 was measured in the blood
T4U_measured	whether T4U was measured in the blood
FTI_measured	whether FTI was measured in the blood
referral source	

Table No. 13 Dataset attributes

Then starting data preprocessing which is a data mining approach to transform the raw data acquired from the dataset into cleaner information that's more suited for our work. Raw data might contain missing or inconsistent values and provide a lot of duplicate information. In this data preprocessing stage, we verified whether there is any data missing or contains erroneous data and outliers that we can identify in the data set or check the absence of data limitations.

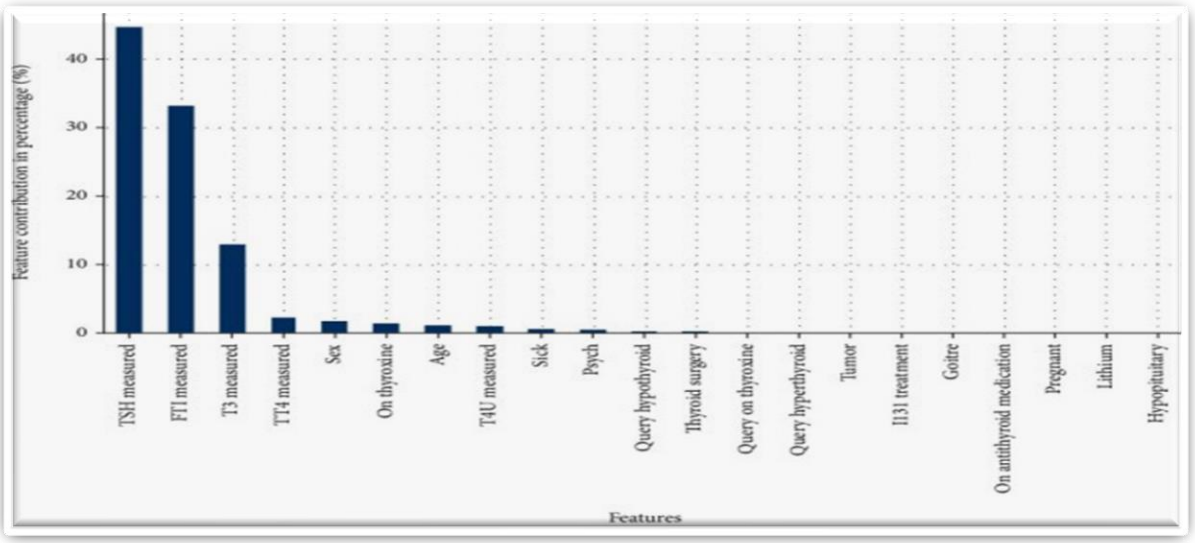


Figure No. 22 Features contributing to classification

Figure No. 23 Dataset distribution 1

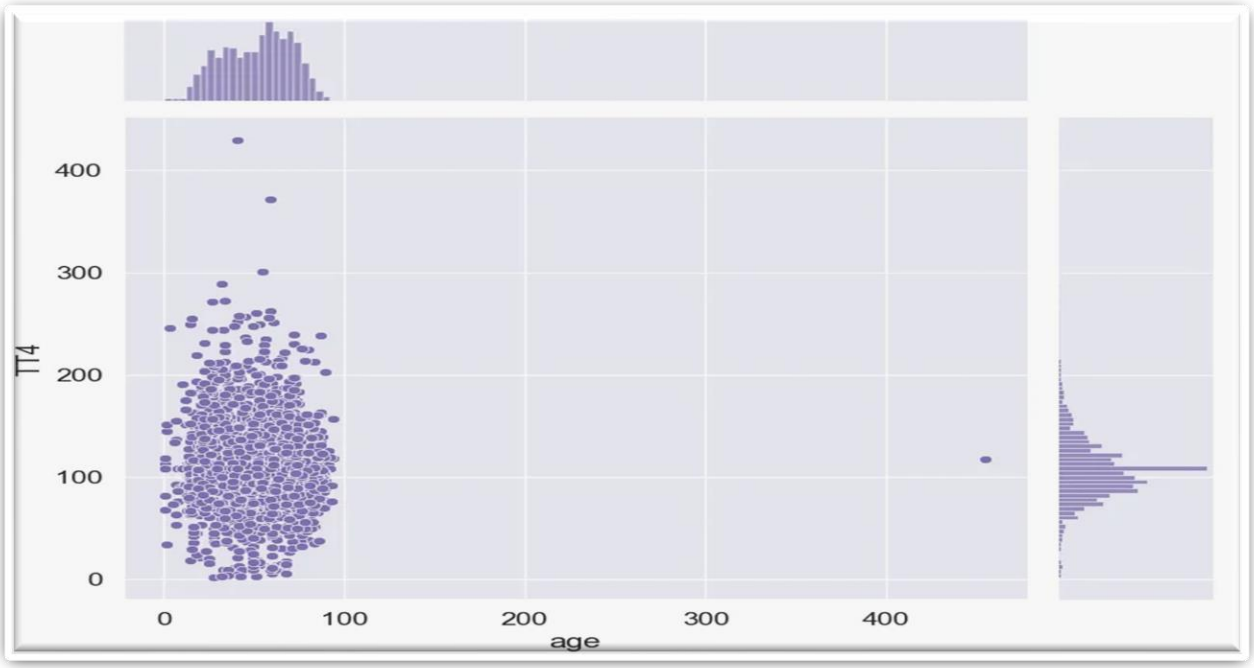


Figure No. 24 Dataset distribution 2

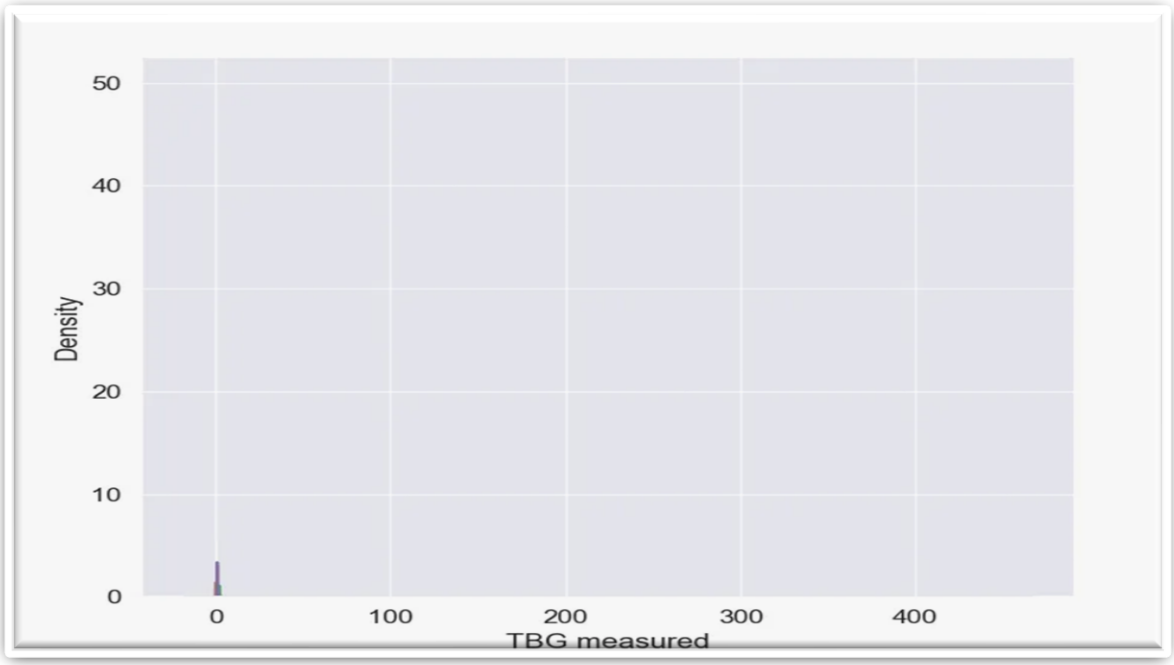
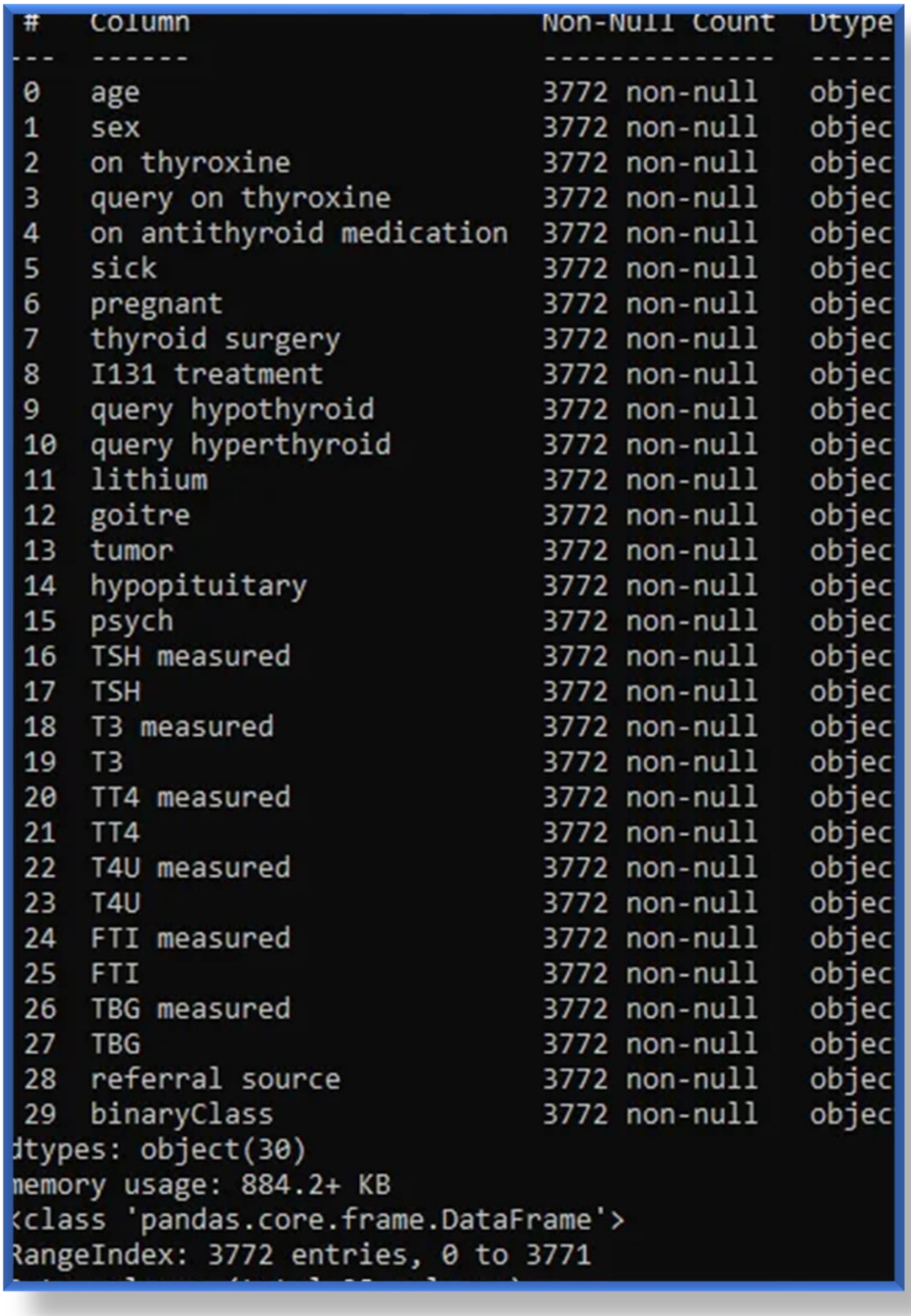
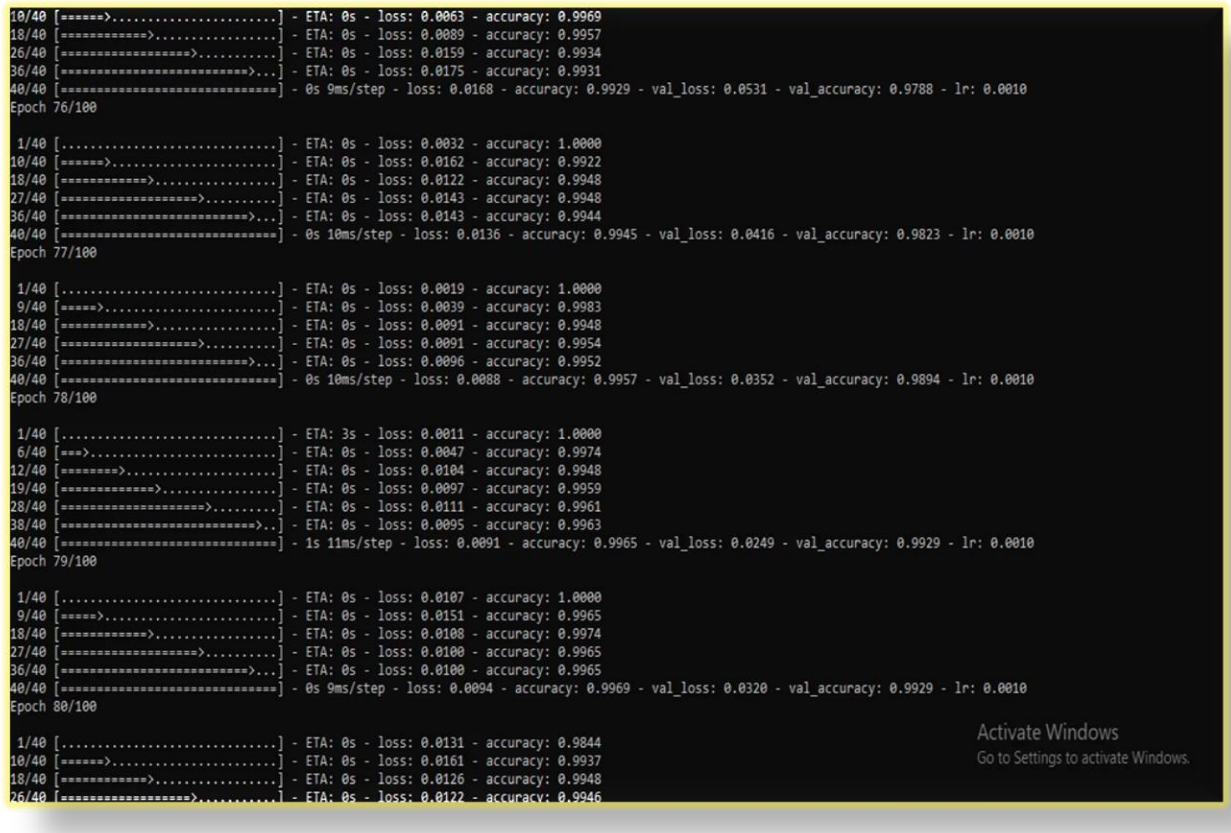


Figure No. 25 Dataset distribution 3



```
#   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
0   age                        3772 non-null   objec
1   sex                        3772 non-null   objec
2   on thyroxine               3772 non-null   objec
3   query on thyroxine         3772 non-null   objec
4   on antithyroid medication  3772 non-null   objec
5   sick                       3772 non-null   objec
6   pregnant                   3772 non-null   objec
7   thyroid surgery            3772 non-null   objec
8   I131 treatment             3772 non-null   objec
9   query hypothyroid          3772 non-null   objec
10  query hyperthyroid         3772 non-null   objec
11  lithium                    3772 non-null   objec
12  goitre                     3772 non-null   objec
13  tumor                      3772 non-null   objec
14  hypopituitary              3772 non-null   objec
15  psych                      3772 non-null   objec
16  TSH measured               3772 non-null   objec
17  TSH                        3772 non-null   objec
18  T3 measured                3772 non-null   objec
19  T3                         3772 non-null   objec
20  TT4 measured               3772 non-null   objec
21  TT4                        3772 non-null   objec
22  T4U measured               3772 non-null   objec
23  T4U                        3772 non-null   objec
24  FTI measured               3772 non-null   objec
25  FTI                        3772 non-null   objec
26  TBG measured               3772 non-null   objec
27  TBG                        3772 non-null   objec
28  referral source            3772 non-null   objec
29  binaryClass                3772 non-null   objec
dtypes: object(30)
memory usage: 884.2+ KB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3772 entries, 0 to 3771
```

Figure No. 26 Part of Running the Algorithms 1



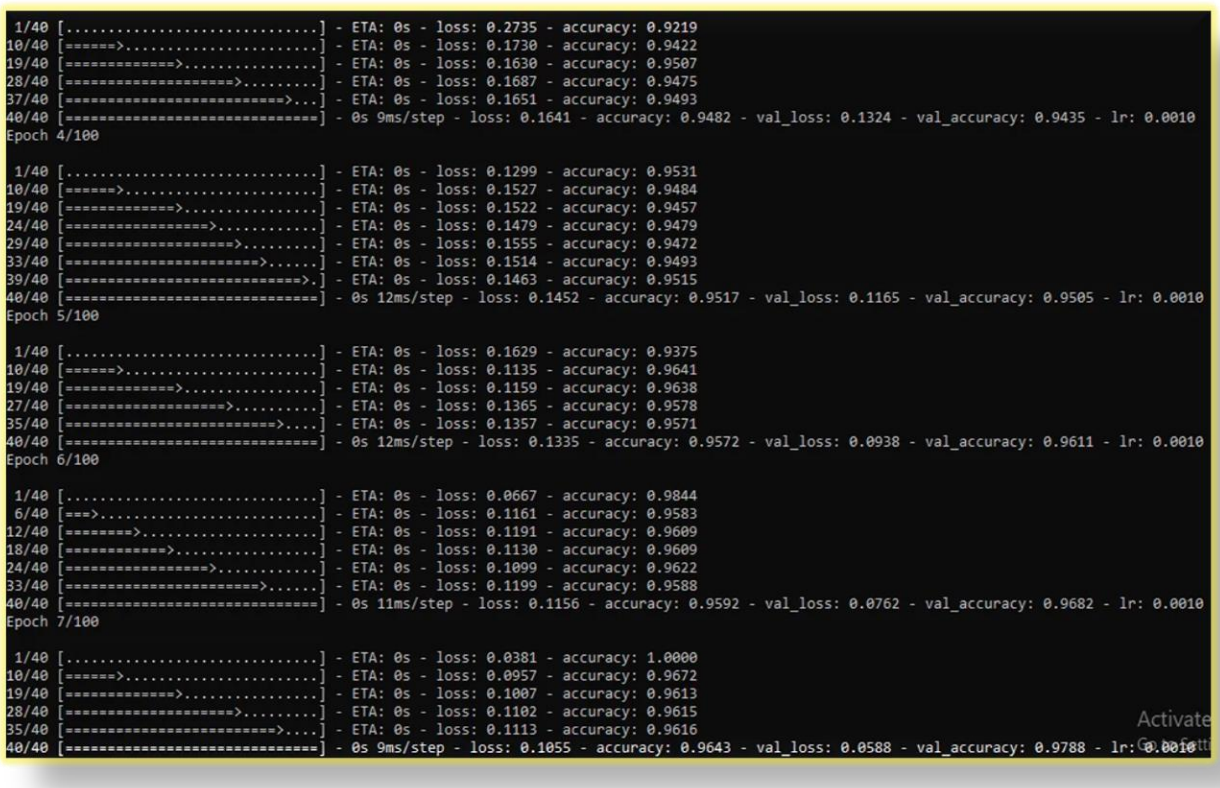


Figure No. 27 Part of running the Algorithms 2

After that we divided our dataset into training and testing dataset, then we import the deep learning classification algorithm an apply a sequential modeling approach using an artificial neural network (ANN) to develop two models - Artificial Neural Network (ANN) - and passed our training and testing dataset into it and then we print the accuracy of the model. Our machine learning model achieved an accuracy of 97.88%

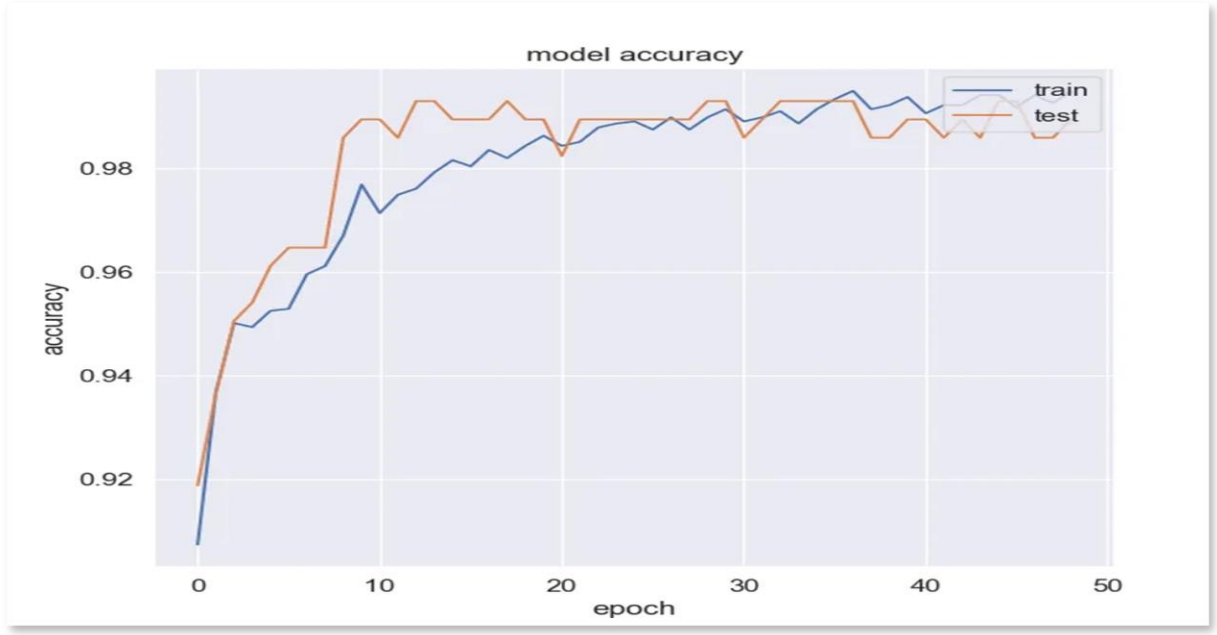
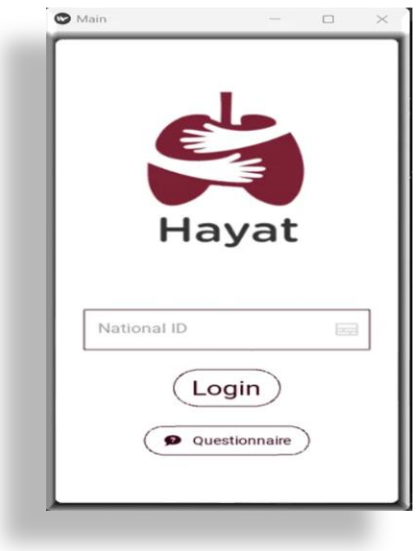


Figure No. 28 Model Accuracy

Once the model was trained and tested, we saved it and deployed on Android Mobile Application with the help of KIVY Framework, to Enable the users to enter their details and runs the algorithms to check if the patient has a thyroid disease or not.



5.2 Tools &Technologies Overview

We have developed this project using the below Technologies and Tools:

- Deep Learning Techniques : Deep learning is a subfield of artificial intelligence that enables machines to learn from data, without being explicitly programmed. In this project, we used deep learning classification algorithms to predict the Thyroid related diseases.

- Algorithm Model: An Artificial Neural Network (ANN) A deep learning model We applied a sequential modeling approach using an artificial neural network (ANN) to develop two models which were able to predict the thyroid diseases.

- Python: All the business logic has been implemented in Python.

- Kivy : Project has been developed over the Kivy framework, which is an Open Source Python App Framework, that enables the development of Mobile applications in Python. Kivy runs on Android, iOS, Linux, macOS and Windows. In our project, we used Kivy to develop a mobile application that allows users to input their symptoms and The Python code interacts with the database to retrieve user data and then the user receives the predictions from the trained machine learning model.

- Kaggle Dataset : We used the Kaggle dataset for training and testing our machine learning model. The Database consists of thyroid patient records. The Patients record is having different attributes described in the dataset.

- MySQL Database : https://www.mysql.com/ MySQL is a relational database management system based on SQL – Structured Query Language. The application is used for a wide range of purposes, including data warehousing, e-commerce, and logging applications.

- Android Operating System : We configured this project on Android operating system.

- Kivy MD
- VISUALSTUDIO Code
- VISUALSTUDIO Microsoft
- Draw.io

5.5 Part of Implementation

```
MDScreen:
    MDCard:
        size_hint: None, None
        size: 300,580
        pos_hint: {'center_x': 0.5,'center_y': 0.5}
        elevation: 6
        padding: 25
        spacing: 25
        orientation: 'vertical'

        FitImage:
            source: "m.png"
            size_hint_y: .25
            pos_hint: {"top": 1}
            radius: 30, 30, 0, 0

        MDTextField:
            id: tf
            hint_text: "National ID"
            mode: "rectangle"
            theme_text_color: "Hint"
            line_color_normal: (0.2, 0, 0.1, 1)
```

Figure No. 29 Part of Login 1 Code

```
    line_color_focus:(0.2, 0, 0.1, 1)
    hint_text_color_focus:(0.2, 0, 0.1, 1)
    icon_right: "id-card"
    icon_right_color_focus: (0.2, 0, 0.1, 1)
    text_color_focus:(0.2, 0, 0.1, 1)
    max_text_length: 10

MDRoundFlatButton:
    text:"Login"
    font_size:25
    pos_hint:{"center_x":.5}
    text_color: (0.2, 0, 0.1, 1)
    line_color: (0.2, 0, 0.1, 1)
    on_release: app.callback()

MDRoundFlatIconButton:
    text: "Questionnaire"
    icon: "chat-question"
    text_color: (0.2, 0, 0.1, 1)
    line_color: (0.2, 0, 0.1, 1)
    pos_hint:{"center_x":.5}
    theme_icon_color: "Custom"
```

Figure No. 30 Part of Login 2 Code

```
from json import load
from turtle import bgcolor
from kivy.app import App
from kivymd.app import MDApp
from kivy.uix.boxlayout import BoxLayout
from kivy.uix.label import Label
from kivymd.uix.selectioncontrol import MDCheckbox
from kivymd.uix.button import MDFillRoundFlatButton
from kivymd.uix.button import MDFlatButton
from kivy.uix.popup import Popup
from kivy.metrics import dp
from kivy.core.window import Window
from kivymd.uix.label import MDLabel
from kivymd.uix.gridlayout import MDGridLayout
from kivymd.uix.boxlayout import MDBoxLayout
from kivymd.uix.button import MDRectangleFlatButton
from kivy.lang import Builder
from kivy.graphics import Color, Rectangle
class YesNoLayout(MDBoxLayout):
    def __init__(self, **kwargs):
        super().__init__(**kwargs)
        self.orientation = 'vertical'
```

Figure No. 31 Part of Questionnaire 1 Code

```
    self.orientation = 'vertical'
    self.elelvation=6
    self.padding = [dp(20), dp(50)]
    self.create_title('Hayat Diagnosis','tahomabd',35,10)
    self.create_question('Have you experienced any unexplained weight loss
    self.create_question('Have you noticed any changes in your appetite or
    self.create_question('Are you experiencing difficulty sleeping or insom
    self.create_question('Have you experienced any tremors or shaking, part
    self.create_question('Have you experienced any muscle weakness or fatig
    self.create_question('Have you noticed any changes in your mood, such a
    self.create_buttons()
    with self.canvas.before:
        # Set the background color of the widget
        Color(0.2, 0, 0.1, 1)
        self.rect = Rectangle(size=self.size, pos=self.pos)
    self.bind(size=self._update_rect, pos=self._update_rect)

def _update_rect(self, instance, value):
    self.rect.pos = instance.pos
    self.rect.size = instance.size

def check_answers(self, instance):
```

Figure No. 32 Part of Questionnaire 2 Code

```
def check_answers(self, instance):
    yes_count = 0
    for child in self.children:
        if isinstance(child, MDBoxLayout):
            if len(child.children) > 0:
                checkbox_layout = child.children[1]
                if len(checkbox_layout.children) > 0:
                    yes_checkbox = checkbox_layout.children[1]
                    if isinstance(yes_checkbox, MDCheckbox) and yes_checkbox.active:
                        yes_count += 1
    if yes_count >= 4:
        message = "\n you need to visit hayat lab \n located in zarqa \n street No.
    else:
        message = "Your health is good."

    popup_layout = MDBoxLayout(orientation='vertical', padding=dp(10))
    popup_label = MDLabel(text=message, halign='center',theme_text_color="Custom",
    popup_layout.add_widget(popup_label)

    button_layout = MDBoxLayout(orientation='horizontal', padding=dp(10))
    button1 = MDRectangleFlatButton(text="OK", font_name="arialbd", md_bg_color=(1,
```

Figure No. 33 Part of Questionnaire 3 Code

```
def create_question(self, text, group, font_path, font_size):
    question_label = MDLabel(text=text, font_name=font_path,font_size=dp(font_size
    self.add_widget(question_label)

    checkbox_layout = MDBoxLayout(size_hint_y=0.15)

    yes_layout = MDBoxLayout(size_hint_x=2, padding=(0.3, 0.3, 0, 0.3))
    yes_label = MDLabel(text='Yes', size_hint_x=3, font_size=dp(18),theme_text_col
    yes_checkbox = MDCheckbox(size_hint_x=None,group=group)
    yes_checkbox.color=(1,1,1,1)
    yes_layout.add_widget(yes_checkbox)
    yes_layout.add_widget(yes_label)
    checkbox_layout.add_widget(yes_layout)

    no_layout = MDBoxLayout(size_hint=(2, 1), padding=(0, 0.3, 0.3, 0.3))
    no_label = MDLabel(text='No', size_hint_x=None, font_size=dp(18),theme_text_co
    no_checkbox = MDCheckbox(size_hint_x=None,group=group)
    no_checkbox.color=(1,1,1,1)
    no_layout.add_widget(no_checkbox)
    no_layout.add_widget(no_label)
    checkbox_layout.add_widget(no_layout)
```

Figure No. 34 Part of Questionnaire 4 Code

```
MDScreen:
    md_bg_color:0.2, 0, 0.1, 1
    MDBoxLayout:
        orientation:"vertical"
        size:root.width,root.height
        MDBoxLayout:
            size_hint_y:0.1
            padding:dp(5)
            size:root.width,root.height
            MDLabel:
                text:"TestResults"
                halign:"center"
                font_style:"H4"
                pos_hint: {"center_x": .1, "center_y": .1}
            MDBoxLayout:
                adaptive_size:True
                spacing:dp(70)
        MDGridLayout:
            size_hint_y:.8
            cols:2
            padding:dp(25)
            spacing:dp(25)
```

Figure No. 35 Part of Test Results 1 Code

```
    ElementCard1:
        text:"Name :"
        subtext:"abdallah najeh thabet"
    ElementCard2:
        text:"NationalID :"
        subtext:"2000295136"
    ElementCard3:
        text:"test1:"
        subtext:"test1fromdatabase"
    ElementCard4:
        text:"test2:"
        subtext:"test2fromdatabase"

MDGridLayout:
    size_hint_y:.9
    cols:1
    padding:dp(25)
    spacing:dp(30)
    ElementCard8:
        text:"test3 :"
        subtext:"test3fromdatabase"
    ElementCard9:
```

Figure No. 36 Part of Test Results 2 Code

```
<FlatButton@ButtonBehavior+Label>:
    text: 'default'
<AdminWindow>:
    id: main_win
    orientation: "vertical"
    canvas.before:
        Color:
            rgba: (1,1,1,1)
        Rectangle:
            size: self.size
            pos: self.pos
    BoxLayout:
        id: top_nav
        size_hint_y: None
        height: 30
        canvas.before:
            Color:
                rgba: (0.2, 0, 0.1, 1)
            Rectangle:
                size: self.size
                pos: self.pos
```

```
MDTextField:
    id: user
    hint_text: "User Name"
    mode: "rectangle"
    theme_text_color: "Hint"
    size_hint: (.4, .1)
    pos_hint: {'center_x' : 0,'center_y' : 0.9 }
    line_color_normal: (0.2, 0, 0.1, 1)
    line_color_focus:(0.2, 0, 0.1, 1)
    hint_text_color_focus:(0.2, 0, 0.1, 1)
    icon_right_color_focus: (0.2, 0, 0.1, 1)
    text_color_focus:(0.2, 0, 0.1, 1)
MDRaisedButton:
    id: UPD
    text: " Update "
    pos_hint: {'center_x' : 0.5,'center_y' : 0.2 }
    md_bg_color: [0.2,0,0.1,1]
    size_hint: (.1, .1)
    height: 30
    on_release: app.Update_data(Nalid.text , user.text, Tsh.text , t3.text , t4.text)
```

Figure No. 37 Part of Admin Code

```
        Rectangle:
            size: self.size
            pos: self.pos
    ToggleButton:
        id: user_toggle
        text: "Manage Users"
        size_hint_y: .1
        state: 'down'
        background_color: (0.2, 0, 0.1, 1)
        background_normal: ""
    Label:
        id: sp
        text: ""
        size_hint_y: .4
MDFlatButton:
    id: fake
    pos_hint: {'center_x' : 0.5,'center_y' : 0 }
    size_hint: (.1, None)
    height: 10
MDTextField:
    id: Nalid
    hint_text: "National ID"
```

```
    FlatButton:
        id: Exit
        text: "Exit"
        on_press: exit()
        size_hint: (.1, None)
        height: 30
    Label:
        text: "Admin Dashboard"
        bold: True
        size_hint: (.9, None)
        height: 30
BoxLayout:
    id: content_nav
    BoxLayout:
        id: nav_tabs
        orientation: 'vertical'
        size_hint_x: .4
        spacing: 3
        canvas.before:
            Color:
                rgba: (0.2, 0, 0.1, 1)
```

Figure No. 38 Part of Admin 2 Code

• • • •

CHAPTER 6 : CONCLUSION

6.1 Conclusion

Thyroid disease is one of the major diseases that are spreading widely nowadays including Jordan as well. The normal and traditional methods of thyroid diagnosis are thorough inspection and also various blood tests. Therefore, we came up with our project idea to raise awareness about thyroid and its related diseases, and the most important is to allow users to get an instant guidance on their thyroid status.

Our project Hayat is done with the help of deep learning algorithms and the Code is written in Python programing language. The algorithm first trained using the data set available from the repository and then tested on the dataset. Then the system will allow the user to enter their details -symptoms and the lab to inter the blood test results, then algorithm runs and predicts the result of having the thyroid disease or not. The main goal for our project is to recognize the disease with a very high correctness into three categories: hyperthyroidism, hypothyroidism, or normal.

Our project will help doctors as well as individuals to have a possible disease diagnosed. And once a person predicts whether or not he/she can be diagnosed with thyroid disease, our system will provide the users with a lab service whom they can visit and do their thyroid tests there, and the results will be uploaded into the system as soon they are ready, and this will save users searching, effort and time. Moreover, each user will have his/her own information record which can be checked up and viewed when ever wanted.

Moreover, Hayat system gives a user-friendly environment and easy to use. As the system is based on a Mobile application, the user can use this system from anywhere and at any time.

To sum up, Disease diagnosis plays a vital role and it is a necessary, so a lot of Healthcare issues can be solved efficiently by applying Machine Learning and deep learning Technologies beside data mining techniques.

6.2 Future Work

- In the future we are thinking to expand the work area of our project and provide more services to our users by using machines learning and deep learning models to diagnose and predict other diseases like Diabetes, Heart Disease, Kidney Disease, Liver Disease and others.

- The system will be configured on several operating systems beside Android.

- Moreover, we will enhance the design of the interfaces and will add more interfaces, as we will provide more services to our users and expand the work area of our project.

- In the future, it is necessary to expand the set of data and attributes considered. and train more algorithms on the data to increase the accuracy and the performance of our system.

- Furthermore, more users will be able to use our system, not only Jordanians who have National No., but also other nationalities, as the username will be changed to Passport No. with password instead of the National No. to help more people.

- The system will be in both Arabic and English.

• • • •

REFERENCES

1- https://projectworlds.in/hypo-thyroid-disease-prediction-machine-learning-project/

2- https://projectworlds.in/multiple-disease-prediction-using-machine-learning/

3- https://www.mayoclinic.org/diseases-conditions/hypothyroidism/symptoms-causes/syc-20350284

4- https://www.drawio.com/

5- https://visualstudio.microsoft.com/

6- https://code.visualstudio.microsoft.com/

7- https://www3schools.com/python/

8- https://www.python.org/shell/

9- https://www.geeksforgeeks.org/kivy-tutorial/

10- https://kivy.org/doc/stable/guide/basic.htm/

11- https://kivymd.readthedocs.io/en/1.1.1/

12- https://www.ncbi.nlm.nih.gov/pmc/articles/PMC

13- Thyroid Disease Prediction Using Machine Learning Approaches | Gyanendra Chaubey - Academia.edu

14- https://www.verywellhealth.com/thyroid-disease-symptoms

15- https://www.kaggle.com/datasets

16- The Sequential model (keras.io)

• • • •