R. V. COLLEGE OF ENGINEERING, BENGALURU-560059

(Autonomous Institution Affiliated to VTU, Belagavi)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Health Diagnostic System

Mini - Project Report

Submitted by

Keshav Bharat **1RV16CS070**
Kshitij Tiwari **1RV16CS075**

in partial fulfillment for the requirement of 5th Semester

DBMS Laboratory Mini Project (16CS54)

Under the Guidance of

Academic Year 2018 - 2019

R.V. COLLEGE OF ENGINEERING, BENGALURU - 560059
(Autonomous Institution Affiliated to VTU, Belagavi)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the project work titled **'Health Diagnostic System'** is carried out by **Keshav Bharat (1RV16CS070), Kshitij Tiwari (1RV16CS075),** who are bonafide students of R. V. College of Engineering, Bengaluru, in partial fulfillment of the curriculum requirement of 5th Semester Database Design Laboratory Mini Project during the academic year **2018-2019**. It is certified that all corrections/suggestions indicated for the internal Assessment have been incorporated in the report deposited in the departmental library. The report has been approved as it satisfies the academic requirements in all respect laboratory mini-project work prescribed by the institution.

Signature of Faculty In-charge

Head of the Department
Dept. of CSE, RVCE

External Examination

Name of Examiners	Signature with date
1	
2	

Table of Contents

1. Introduction

Medical diagnosis is the process of determining which disease or condition explains a person's symptoms and signs. It is most often referred to as diagnosis with the medical context being implicit. The information required for diagnosis is typically collected from a history and physical examination of the person seeking medical care.
Often, one or more diagnostic procedures, such as diagnostic tests, are also done during the process. Sometimes posthumous diagnosis is considered a kind of medical diagnosis.
Our Aim is to create a health diagnosis system where accurate diagnosis is made on the basis of symptoms.

Clinical decision support systems are interactive computer programs designed to assist health professionals with decision-making tasks. The clinician interacts with the software utilizing both the clinician's knowledge and the software to make a better analysis of the patients' data than either human or software could make on their own. Typically the system makes suggestions for the clinician to look through and the clinician picks useful information and removes erroneous suggestions.

Once a diagnostic opinion has been reached, the provider is able to propose a management plan, which will include treatment as well as plans for follow-up. From this point on, in addition to treating the patient's condition, the provider can educate the patient about the etiology, progression, prognosis, other outcomes, and possible treatments of her or his ailments, as well as providing advice for maintaining health.

A treatment plan is proposed which may include therapy and follow-up consultations and tests to monitor the condition and the progress of the treatment, if needed, usually according to the medical guidelines provided by the medical field on the treatment of the particular illness.

1.1 Objective

Our objective is to create a system which can efficiently search for and draw inferences from a vast database of symptoms causing various diseases. This will be beneficial in the healthcare and medical sectors as it provides the means for easy validation and self-diagnosis in areas with lesser development.

The diagnosis is arrived at as a result of the output given by a neural network. The neural network will be trained on a dataset of symptoms and diagnosis.

On the whole, we aim to realize a self-correcting system involving generation of database, framing the model, training and using this model. As the model is dynamic it learns as the users grow, predicating with better accuracies. Moreover the model has the ability to learn any new disease found or new diagnosis brought into market, which is been fed into the database, hence acquainted with current trend.

1.2Scope

India has one doctor for every 1,667 patients. Furthermore, the medical sector is plagued by several problems such as incompetence of practitioners, prohibitive pricing of drugs, and lack of diagnostic facilities, especially in the rural areas.

A low cost, reliable and accurate solution is the need of the hour to address the growing health sector imbalance. A medical diagnostic system is a good solution to all of the aforementioned issues, as it is relatively cheaper to physical diagnostic systems, servicing can be done remotely, and results will be reliable with a high probability.

Reduce readmissions Machine learning can reduce readmissions in a targeted, efficient, and patient-centred manner. Clinicians can receive daily guidance as to which patients are most likely to be readmitted and how they might be able to reduce that risk.

Prevent hospital acquired infections (HAIs) Health systems can reduce HAIs, such as central-line associated bloodstream infections (CLABSIs)-40 percent of CLABSI patients die-by predicting which patients with a central line will develop a CLABSI. Clinicians can monitor high-risk patients and intervene to reduce that risk by focusing on patient-specific risk factors.

Reduce hospital Length-of-Stay (LOS) Health systems can reduce LOS and improve other outcomes like patient satisfaction by identifying patients that are likely to have an increased LOS and then ensure that best practices are followed.

Predict chronic disease Machine learning can help hospital systems identify patients with undiagnosed or misdiagnosed chronic disease, predict the likelihood that patients will develop chronic disease, and present patient-specific prevention interventions.

2. Specific Requirements

Software requirements specification establishes the basis for an agreement between customers and contractors or suppliers on how the software product should function (in a market-driven project, these roles may be played by the marketing and development divisions). Software requirements specification is a rigorous assessment of requirements before the more specific system design stages, and its goal is to reduce later redesign.

2.1 Software Requirements

The software requirements are detailed below:

- Design and planning : Microsoft Visio 2016
- Language Used : SQL, Python 3, JAVA 8
- Database : MySQL 5.5
- Deep Learning Libraries : TensorFlow 18, Keras 2.2.2

2.2 Hardware Requirements

The hardware specifications are listed below:

- CPU : Intel, AMD Ryzen 5, 2.00GHz or higher
- Operating System : Ubuntu 16 or greater, Windows 7 or greater
- RAM : 4GB DDR4, 2500MHz
- Hard Drive : 5GB free space
- Display : Standard VGA display

2.3 Functional Requirements

This section includes the requirements that specify all the fundamental actions of the software system for the user classes.

2.3.1 Patient

2.3.1.1 Patient Registration

Given that a patient has installed the application, then the patient should be able to register through it. The patient must provide patient-name, password and e-mail address. The patient can choose to provide a regularly used phone number.

2.3.1.2 Patient Login

Given that a patient has registered, the patient should be able to log in to the application using credentials provided at the time of registration. The log-in information will be stored on the device and in the future the patient should be logged in automatically.

2.3.1.3 Reset Password

Given that a patient has registered, then the user should be able to reset his/her

password by email/phone.

2.3.1.4 Search for Doctor

Patient searches for doctor either by the specialization, or by doctor name.
Search results are displayed in a list, with a maximum of 50 results.
The results are sorted by ratings given by other patients as well as by specializations .
If no results are found, then the patient is informed and remains on search screen.

2.3.1.5 Request Appointment

Given that patient has found a doctor, an appointment can be requested with a brief description of the ailment, not exceeding 50 words.

2.3.2 Doctor

2.3.2.1 Create an Account

Doctor needs to specify name, password, email address, clinic name, location, visiting hours, qualifications, specialization(s), and experience for registration. The doctor is then listed in search results, based on various filters applied while searching.

2.3.2.2 Doctor Login

Given that a doctor has registered, he should be able to log in to the application using credentials provided at the time of registration. The log-in information will be stored on the device and in the future the doctor should be logged in automatically.

2.3.2.3 Schedule Appointments

Given a patient requesting an appointment, doctor will schedule an appointment and request the patient to confirm the appointment.

2.3.2.4 Diagnosis

Based on the symptoms fed in by the doctor, a diagnosis is given along with the treatment and cost.

2.3.2.5 Payment

Payment for the treatment is transferred against the doctor's name, with a smaller portion deducted towards the application.

2.3.3 Administrator

2.3.3.1 Administrator Login

The administrator will log in using the credentials of the administrator account,

which are predetermined.

2.3.3.2 Verify Doctor

The administrator should verify the details provided by the doctor and ensure that all are valid and accurate. The doctor then then be verified or rejected and the appropriate email is sent.

2.3.3.3 Manage Doctors

The administrator can add doctors, modify their status as rejected, approved or pending verification, and can remove doctors from the database.

2.3.3.4 Manage Patients

Given the administrator has logged in, he can add, modify or delete any of the user details in the database.
The user's consultation history can be viewed and favourite doctor will be assigned.

2.4 Hardware Interfaces

Since neither the mobile application nor the web portal have any designated hardware, it does not have any direct hardware interfaces. The interfacing with the input panel and the database are handled by the underlying operating systems.

3. ER Diagram

An entity relationship diagram model is a graphical representation of an information system that depicts the relation among various entities, real world objects within the system. It is used as the foundation for a relational database.
There are 3 basic components of an ER diagram:

- Entities, which are objects or concepts that can have data stored about them.
- Attributes, which are properties or characteristics of entities. (Can denote primary key, foreign key etc.)
- The relationships between and among those entities.

In our ER diagram the main entities that take part in database creation are doctor, patient, symptoms, treatment, diagnosis and appointment.

The patient and doctor interact with each other via an appointment. The patient takes an appointment to meet the doctor and doctor can only attend patients with an appointment thus patients must have total participation. All the details of the patient's condition and diagnosis report can be accessed via their appointment number on that particular date (to remove confusion among patients with same appointment number on different dates), using the appointment we can note the diagnosis prescribed by the system/doctor i.e. the diagnosis entity is related to appointment via 'results in' relationship, it is an 1:1 relationship as every appointment must have a single diagnosis. Diagnosis entity is related to the symptoms as it helps in finding and pinpoint the correct diagnosis based on the visible symptoms, there may be N numbers of symptoms resulting in a disease, therefore, the diagnosis-symptoms relation is 1:N. As these diagnostics has to be treated, a treatment entity is related to diagnosis via 'treats' relationship, assuming that all the diagnosis predicted are treatable we can safely assume that the relation is of 1:1 kind. This entity-relationship can be visualized in figure 1.

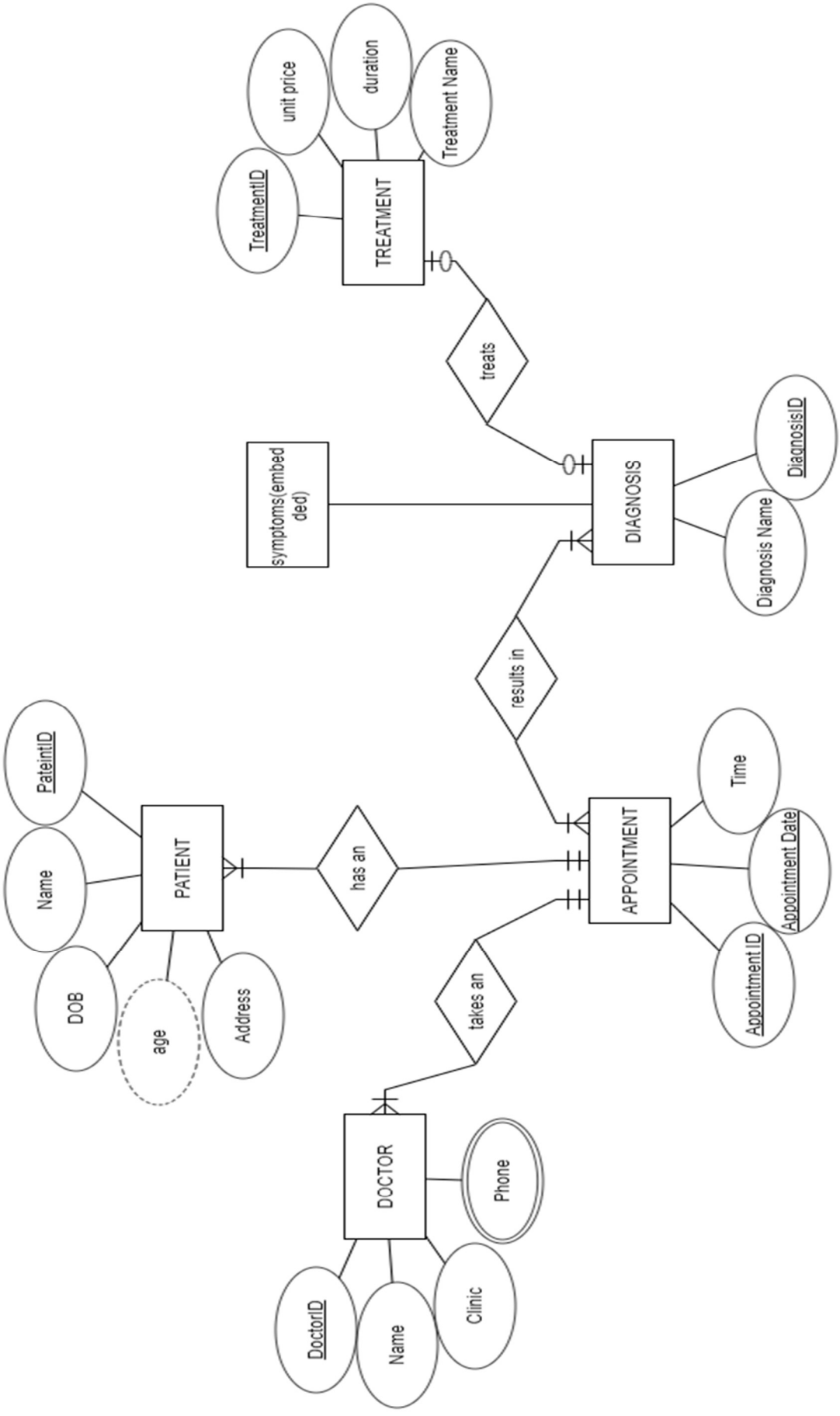

Figure 1 ER Diagram of health diagnosis system

4. Detailed design

In our project the detailed design of the database and the data flow is given by DFD. A data flow diagram (DFD) maps out the flow of information for any process or system. Various symbols and shapes are used to show data inputs, outputs, storage points, processes and the routes between each destination.
Data flow diagrams are used to graphically represent the flow of data in a business information system. DFD describes the processes that are involved in a system to transfer data from the input to the file storage and reports generation.
A data flow diagram can dive into progressively more detail by using levels and layers, zeroing in on a particular piece. DFD levels are numbered 0, 1 or 2, and occasionally go to even Level 3 or beyond.

a. DFD level 0

DFD Level 0 is also called a Context Diagram. It's a basic overview of the whole system or process being analyzed or modeled. It's designed to be an at-a-glance view, showing the system as a single high-level process, with its relationship to external entities. Here patient and doctor entity interact with each other with the help of the health diagnostics system. Fig. 2 illustrates our level 0 DFD.

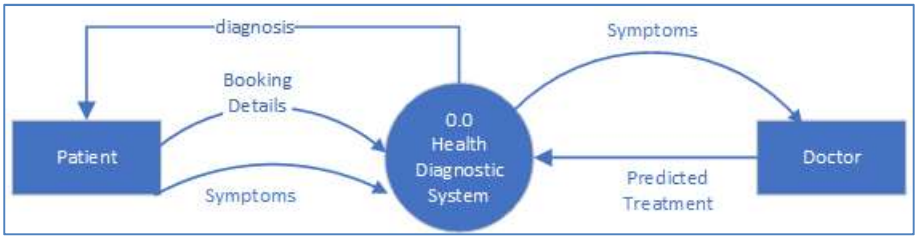

Figure 2 DFD level 0

b. DFD level 1

DFD Level 1 provides a more detailed breakout of pieces of the Context Level Diagram. The patient can set an appointment with the doctor by requesting to the system which fetched the patient data from the database. In the same way doctors can accept/reject the appointment and prescribe as appropriate treatment as shown in fig.3

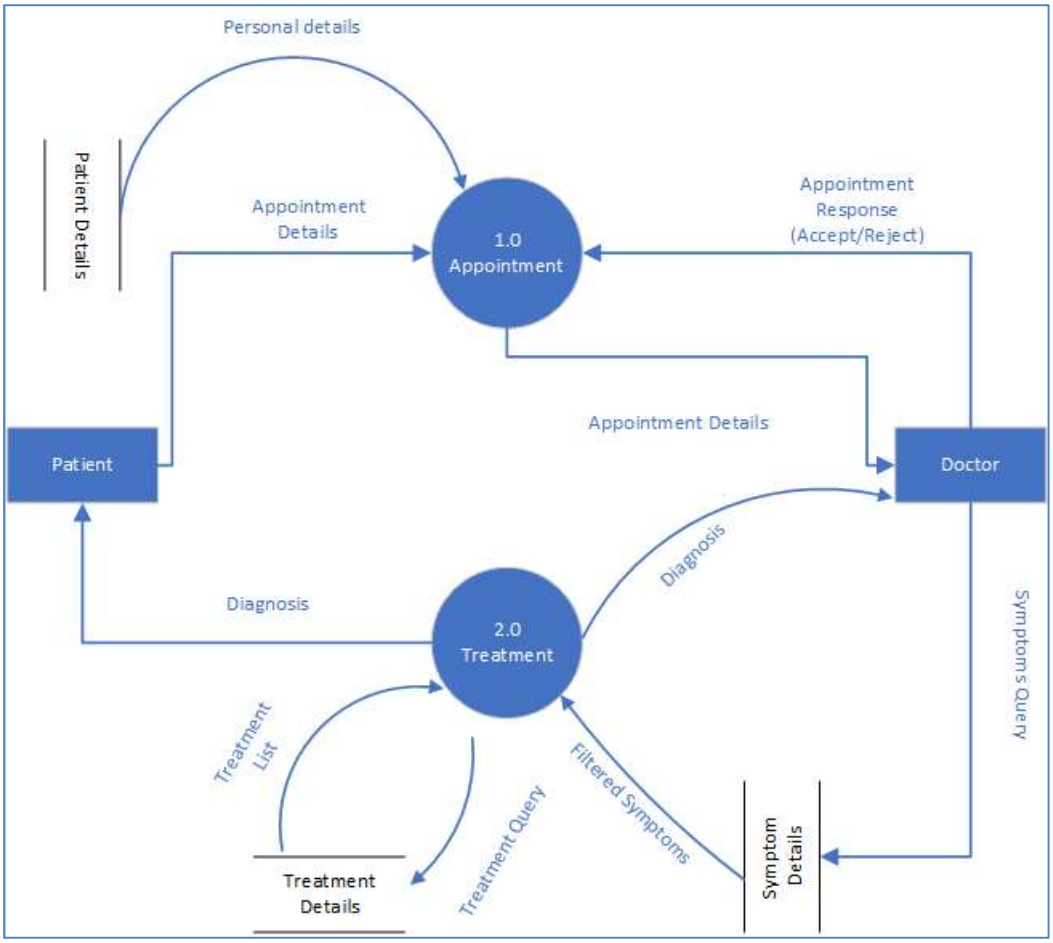

Figure 3 DFD level 1

c. DFD level 2

A level 2 data flow diagram (DFD) offers a more detailed look at the processes that make up an information system than a level 1 DFD does. It can be used to plan or record the specific makeup of a system.

It expands upon the functions mentioned in the level 1 diagram. It also includes all the modules required for a process to work properly.

i. Appointment Process:

The level 2 DFD for the appointment process of the application is given in Fig. 4. Appointment has been exploded into 3 sub processes which need to work together to achieve the goal.

Patient logs in to the system using his credentials. Once logged in, the patient can request an appointment from a doctor of his choice. The request is forwarded to the doctor, who either accepts or rejects it.

Once an appointment is accepted, the details are written into the appointment table and patient is notified.

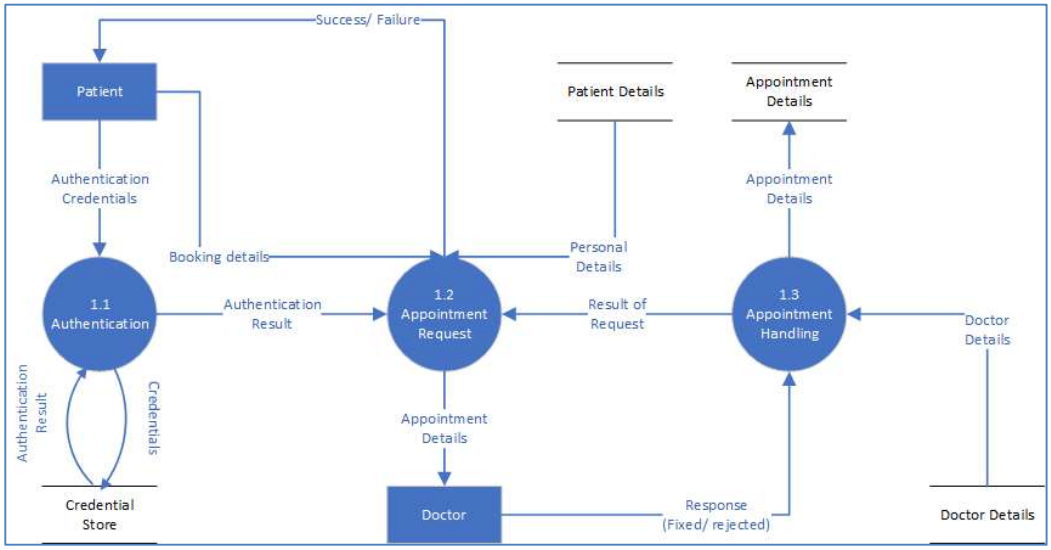

Figure 4 DFD for appointment process

ii. Treatment Process:

The level 2 DFD for the treatment process of the application is given in Fig. 5. Treatment has been exploded into 4 sub processes which need to work together to achieve the goal.

Doctor enters the observed symptoms, and the closest symptoms are taken from the database. A symptoms vector is generated, and diagnosis is generated from the same. Based on this diagnosis, predicted treatment is generated, which is passed to the doctor and the patient.

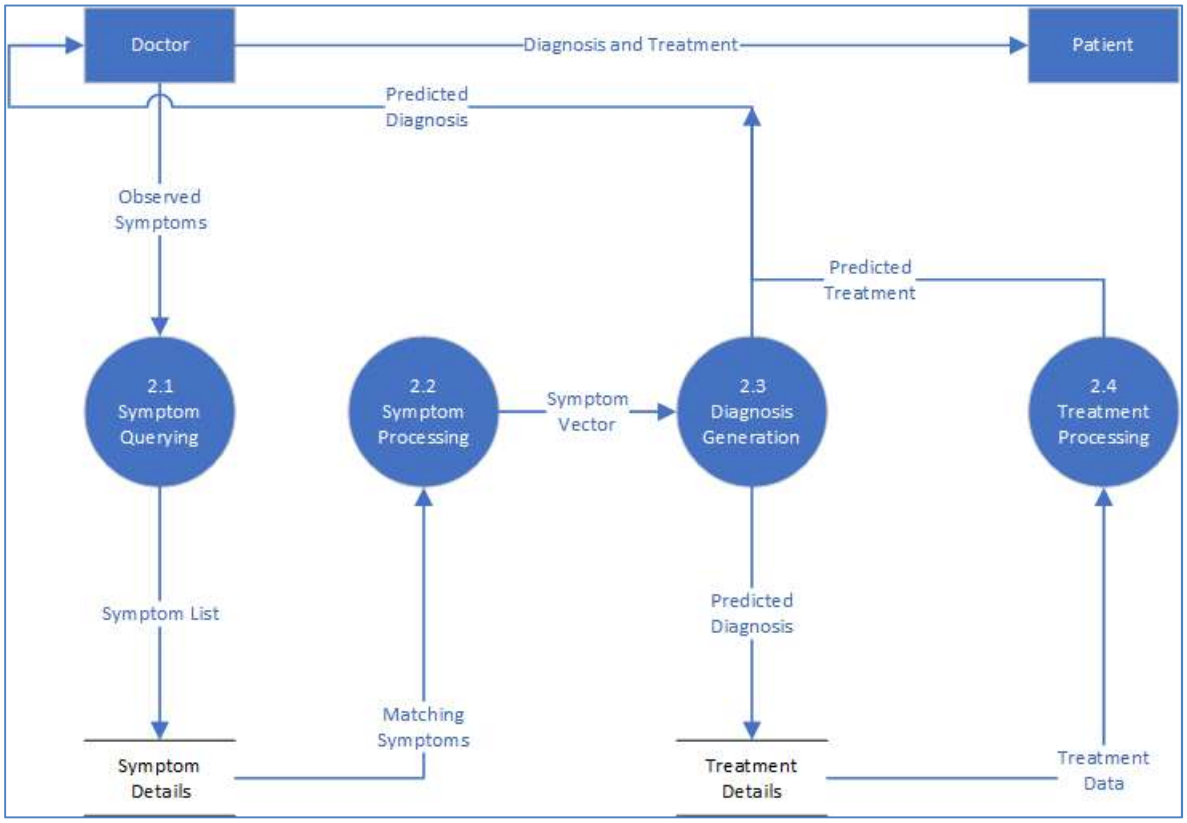

Figure 5 DFD for Treatment Process