

## ABSTRACT

This system supports an end user and online consultation. Here we propose a framework that enables clients to get moment direction on their medical problems through an astute social intelligent health care system online. The framework is bolstered with different symptoms and the disease or illness associated with those systems. Also the system allows the user to share their symptoms and issues about disease. The health care industry generally diagnosis is done by doctor's expertise and experience. Computer Aided Decision Support system plays a major role in medical field. With the growing research on disease prediction system, it has become important to categories the research outcome and provides readers with an overview of the existing disease prediction techniques in each category. Symptoms are one of the main data mining analytical tools that can be utilised to make predictions for medical data. From the study it is observed that Intelligent Algorithm improves the accuracy of the disease prediction system. The commonly used techniques for diseases prediction and their complexities are summerised in this paper. Data mining as a field of research has already well proven capabilities of identifying hidden patterns, analysis towards generating novel and keep insights of these large biomedical data also. Uncovering new biomedical and health care related knowledge to support clinical decision making, is another dimension of data mining. Through massive literature survey, it is found that early disease prediction is the most dominated area of research and health care sector. As health care domain is bit wider domain and having different disease characteristics, different techniques have their own prediction efficiency, which an be enhanced and changed in order to get into most optimize way. The health care industries collect huge amounts of data that contain some hidden information, which is useful for making effective decisions.

For providing appropriate results and making effective decisions on data, some advanced data mining techniques are used. In this study, a Disease Prediction System (DPS) is developed using Naives Bayes and Decision Tree algorithms for predicting the risk level of disease. The system uses 15 medical parameters such as age, sex, blood pressure, cholesterol, and obesity for prediction. The DPS predicts the likelihood of patients getting common disease. It enables significant knowledge. E.g. Relationships between medical factors related to diseases and patterns, to be established. We have employed the multilayer perception neural network with back propagation as the training algorithm. The obtained results have illustrated that the designed diagnostic system can effectively predict the risk level of diseases.

**Chapter-1****INTRODUCTION****Introduction:**

The main objective of this research is to develop a disease prediction system. The system can discover and extract hidden knowledge associated with diseases from a historical data set. Disease prediction system aims to explore machine learning techniques on medical data set to assist in the prediction of the heart diseases.

The 'Diabetes prediction' a Web Application which predicts whether the person is suffering from Diabetes or not according to the given symptoms. Disease remain the biggest cause of deaths for the last two decades. Recently computer technology and machine learning techniques to develop software to assist doctors in making decision of disease in early stage. Disease prediction system can assist medical professionals in predicting disease status based on the clinical data of patients. Machine learning with intelligent algorithms can be used to tackle the said problem of prediction in medical database involving multiple inputs. The health-care industry collects huge amounts of health-care data and that need to be mined to discover hidden information for effective decision making.

Clinicians and patients need reliable information about an individual's risk of developing disease. Ideally, they would have entirely accurate data and would be able to use a perfect model to estimate risk. Such a model would be able to categorize people with disease and others. Indeed, the perfect model would be able to predict the stage of disease's onset.

Here we use some intelligent data mining techniques to guess the most accurate illness that could be associated with patient's details. Based on result, the can contact doctor accordingly for further treatment. The system allows user to view doctor's details too. Among all fatal disease, common diseases are considered as the most prevalent. Medical practitioners conduct different surveys on common diseases and gather information of patients, their symptoms and disease progression. Increasingly are reported about patients with common diseases who have typical symptoms. In this fast moving world people want to live a very luxurious life so they work like a machine in order to earn lot of money and live a comfortable life therefore in this race they forget to take care of themselves, because of this there food habits change their entire lifestyle change, in this type of lifestyle they are more tensed they have blood pressure, sugar at a very young age and they don't give enough rest for themselves and eat what they get and they even don't bother about the quality of the food if sick the go for their own medication as a result of all these small negligence it leads to a major threat that is the heart disease, diabetes, obesity etc.

Data analysis proves to be crucial in the medical field. It provides a meaningful base to critical decisions. It helps to create a complete study proposal. One of the most important uses of data analysis is that it helps in keeping human bias away from medical conclusion with the help of proper statistical treatment. By use of data mining for exploratory analysis because of nontrivial information in large volumes of data. The health care industries collect huge amounts of data that contain some hidden information, which is useful for making effective decisions for providing appropriate results and making effective decisions on data, some data mining techniques are used to better the experience and conclusion that have been given. Disease predictor system will use the data mining knowledge to give a user-

oriented approach to new and hidden patterns in the data. The knowledge which is implemented can be used by the health care experts to get better quality of service and to reduce the extent of adverse medicine effect.

Clinical decisions are often made based on doctor's insight and experience rather than on the knowledge rich data hidden in the data set. This practice leads to unwanted biases, errors and excessive medical costs which affects the quality of service provided to patients. The proposed system will integrate clinical decision support with computer-based patient records (Data Sets).

This will reduce medical errors, enhance patient safety, decrease unwanted practice variation, and improve patient outcome. This suggestion is promising as data modeling and analysis tools, e.g., data mining, have the potential to generate a knowledge rich environment which can help to significantly improve the quality of clinical decisions. There are voluminous records in medical data domain and because of this, it has become necessary to use data mining techniques to help in decision support and prediction in the field of health care. Therefore, medical data mining contributes to business intelligence which is useful for diagnosing of disease.

Machine learning is the process of finding previously unknown patterns and trends in databases and using that information to build predictive models. Data mining combines statistical analysis, machine learning and database technology to extract hidden patterns and relationships from large databases. The World Health Statistics 2012 report enlightens the fact that one in three adults worldwide has raised blood pressure - a condition that causes around half of all deaths from stroke and heart

disease. Heart disease, also known as cardiovascular disease (CVD), encloses a number of conditions that influence the heart – not just heart attacks. Common diseases like heart disease was the major cause of casualties in the different countries including India. These common yet dangerous disease kills one person every 34 seconds in the United States. The term “cardiovascular disease” includes a wide range of conditions that affect the health and the blood vessels and the manner in which blood is pumped and circulated through the body. Diagnosis is complicated and important task that needs to be executed accurately and efficiently. The diagnosis is often made, based on doctor’s experience & knowledge. This leads to unwanted results & excessive medical costs of treatments provided to patients. Therefore, an automatic medical diagnosis system would be exceedingly beneficial.

## Chapter-2

### PROBLEM STATEMENT

To design a Health Prediction System for medical data classification and early disease prediction by using different algorithms. It might have happened so many times that you or someone need doctor's help immediately, but they are not available due to some reason. People cannot identify his or her symptoms and take medicines without consulting doctors. Some medicines are very much harmful to health. So user needs online consultation. Also one main reason of this online health care system is in literature survey, it is found that early disease prediction is the most demanded area of research in health care sector. As health care domain is bit wider domain and having different disease characteristics, different techniques have their own prediction efficiencies, which can be enhanced and changed in order to get into most optimize way.

The term 'common disease' includes the diverse diseases that affect health. The number of people suffering from these common disease is on the rise (health topics, 2010). The report from world health organization shows us a large number of people that die every year due to the these common disease all over the world.

Common diseases is also stated as one of the greatest killers in Africa. Data mining has been used in a variety of applications such as marketing, customer relationship management, engineering, and medicine analysis, expert prediction, web mining and mobile computing. Of late, data mining has been applied successfully in health care fraud and detecting abuse cases.

These disease can be managed effectively with a combination of lifestyle changes, medicine and, in some cases, surgery. With the right treatment, the symptoms of

diseases can be reduced and the functioning of the health improved. The predicted results can be used to prevent and thus reduce cost for surgical treatment and other expensive. The overall objective of my work will be to predict accurately with few tests and attributes the presence of common disease. Attributes considered form the primary basis for tests and give accurate results more or less. Many more input attributes can be taken but our goal is to predict with few attributes and faster efficiency the risk of having common disease. Decisions are often made based on doctors' intuition and experience rather than on the knowledge rich data hidden in the data set and databases. This practice leads to unwanted biases, errors and excessive medical costs which affects the quality of service provided to patients.

Data mining holds great potential for the health care industry to enable health systems to systematically use data and analytics to identify inefficiencies and best practices that improve care and reduce costs. According to (Wurz & Takala, 2006) the opportunities to improve care and reduce costs concurrently could apply to as much as 30% of overall health care spending. The successful application of data mining in highly visible fields like e-business, marketing and retail has led to its application in other industries and sectors. Among these sectors just discovering is Health care. The health care environment is still „information rich“ but „knowledge poor“. There is a wealth of data available within the health care systems. However, there is a lack of effective analysis tools to discover hidden relationships and trends in the data for African genres.



## Chapter- 3

### OBJECTIVE AND SCOPE OF PROJECT

#### **Purpose:**

It might have happened so many times that you or someone yours need doctors help immediately, but they are not available due to some reason. The Disease Prediction application is an end user support and online consultation project. Here, we propose a web application that allows users to get instant guidance on their heart disease through an intelligent system online. The application is fed with various details and the heart disease associated with those details. The application allows user to share their heart related issues. It then processes user specific details to check for various illness that could be associated with it. Here we use some intelligent data mining techniques to guess the most accurate illness that could be associated with patient's details. Based on result, the can contact doctor accordingly for further treatment. The system allows user to view doctor's details too. The system can be used for free heart disease consulting online. The purpose of this document is to outline the operational requirements to the website in order to make it more attractive, efficient & user-friendly.

#### **Scope of Project:**

The project "Disease prediction" has been developed on python and Tensorflow . React is used for the web front-end development and Tensorflow is an open source machine learning library used for research and production.

This project is developed to computerized the process of Diabetes prediction.

The purpose of the project entitled as “Disease prediction” is to predict whether the person is suffering from any Disease or not according to the given symptoms. Here the scope of the project is that integration of clinical decision support with computer-based patient records could reduce medical errors, enhance patient safety, decrease unwanted practice variation, and improve patient outcome. This suggestion is promising as data modeling and analysis tools, e.g., data mining, have the potential to generate a knowledge-rich environment which can help to significantly improve the quality of clinical decisions.

## Chapter-4

### SOFTWARE REQUIREMENTS SPECIFICATION

The app will ask the symptoms according to your Diabetes . For Diabetes the symptoms are following:

- Age
- Insulin
- Blood Pressure
- Height
- Weight
- According to the symptoms the neural network will of web app will predict whether the person is suffering from Diabetes Disease or not.

#### **Intended Audience:**

- **Testers:** The testers would use this document to know the interface website.

#### **Definitions, Acronyms, and Abbreviations:**

- **SRS** – Systems Requirements Software, this document which outlines the requirements that the software must fulfill. Entirely design independent.
- **User** – any person who uses the website and access its content.

- **UML:** Unified Modeling Language
- **ER:** Entity Relationship

**Overview:** The next chapter, the General Description section, of this document gives an overall description of the product. It describes the product functions, user characteristics and constraints that are applied while working with product and this all information is used to establish a context for the technical requirements specification in the next chapter.

The third chapter, Specific Requirements section, of this document is written primarily for the developers and describes in technical terms the details of the functionality of the product.

Both sections of the document describe the same software product in its entirety, but are intended for different audiences and thus use different language.

### **Overall Description:**

- **Product Perspective:** This product is designed in order to predict the risk of suffering from Diabetes issue.
- **Product Functionality:** This product provides the prediction using the neural network and symptoms, whether the user is suffering from Diabetes or not. The app will ask about symptoms according the specific selected Diabetes issue type
- **User Characteristics:** The User will just have to click on the given symptoms

as options.

### **Operating Environment:**

The operating environment of this project used some software and operating system. The project “Diabetes prediction” has been developed on Python and TensorFlow. So, the software, hardware and operating system used are:

- **Operating System:**

1. Windows 7/8/10
2. Linux, Ubuntu
3. MacOS

- **Software Requirement:**

1. Spider editor
2. Visual Studio Code
3. Octae
4. Tensorflow
5. Anaconda

- **Hardware Requirement:**

1. 64-bit architecture
2. Windows or Linux Operating System
3. 4 or more CPUs/cores
4. At least 8 GB of memory/RAM

5. At least 50 GB of disk space
6. Access to the public internet, either directly or via proxy

### **Design and Implementation Constraints:**

For designing and implementing this project there we use different –different programming languages and editors which are:

- 1. TensorFlow :** TensorFlow is an end-to-end open source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML powered applications. Build and train ML models easily using intuitive high-level APIs like Keras with eager execution, which makes for immediate model iteration and easy debugging.

- 2. Python :**

Python is an [interpreted, high-level, general-purpose programming language](#) . Created by [Guido van Rossum](#) and first released in 1991, Python's design philosophy emphasizes [code readability](#) with its notable use of [significant whitespace](#). Its [language constructs](#) and [object-oriented](#) approach aim to help programmers write clear, logical code for small and large-scale projects.

Python is [dynamically typed](#) and [garbage-collected](#). It supports multiple [programming paradigms](#), including [structured](#) (particularly, [procedural](#)),

object-oriented, and [functional programming](#). Python is often described as a "batteries included" language due to its comprehensive [standard library](#).

### **User Documentation:**

The Diabetes prediction is user interactive and easy for use for the users. A user document should be provided at the end of the development.

### **Assumptions and Dependencies:**

This website will response to your queries within the given time limitation and as per the government criteria.

### **External Interface Requirements:**

#### **● User Interfaces:**

Here user interfaces are provided by certain options like:

#### **1. Symptoms filling**

#### **● Hardware Interfaces:**

Hardware Interfaces are provided by:

1. Touch Screen
2. Keyboard
3. Mouse
4. Monitor

## 5. Speakers

### ● **Software Interfaces:**

Software Interfaces are provided by:

1. Tensorflow
2. Visual Studio Code
3. Spider Editor

### ● **Communication Interfaces:**

At the transmission and network layers, the Transmission Control Protocol (TCP) and the Internet Protocol (IPv4) will be utilized in the web-based portion of this product.

### **Environmental Requirements:**

#### **1. Functional Requirements:**

- The User can select the options of Yes or No for the symptoms.
- The User will have to fill some fields manually.

#### **2. Non- Functional Requirements:**



- **Performance:** App performance is quite good but it may work slower depends on internet connectivity speed i.e. 80% of work will be processed in seconds though Express API.

### **Feasibility Study:**

- **Technical Feasibility:**

This study is carried out to check the technical feasibility, project is legally and technically feasible as well as economically justifiable. Other required technical requirements are not at all as much expensive.

- **Operational Feasibility:**

It is Easy to operate a Web Application than any other system. And the application is created with a very easy work flow. And all the useful features are easily locatable in the application.

- **Legal Feasibility:**

This Application will not conflict any law and acts.

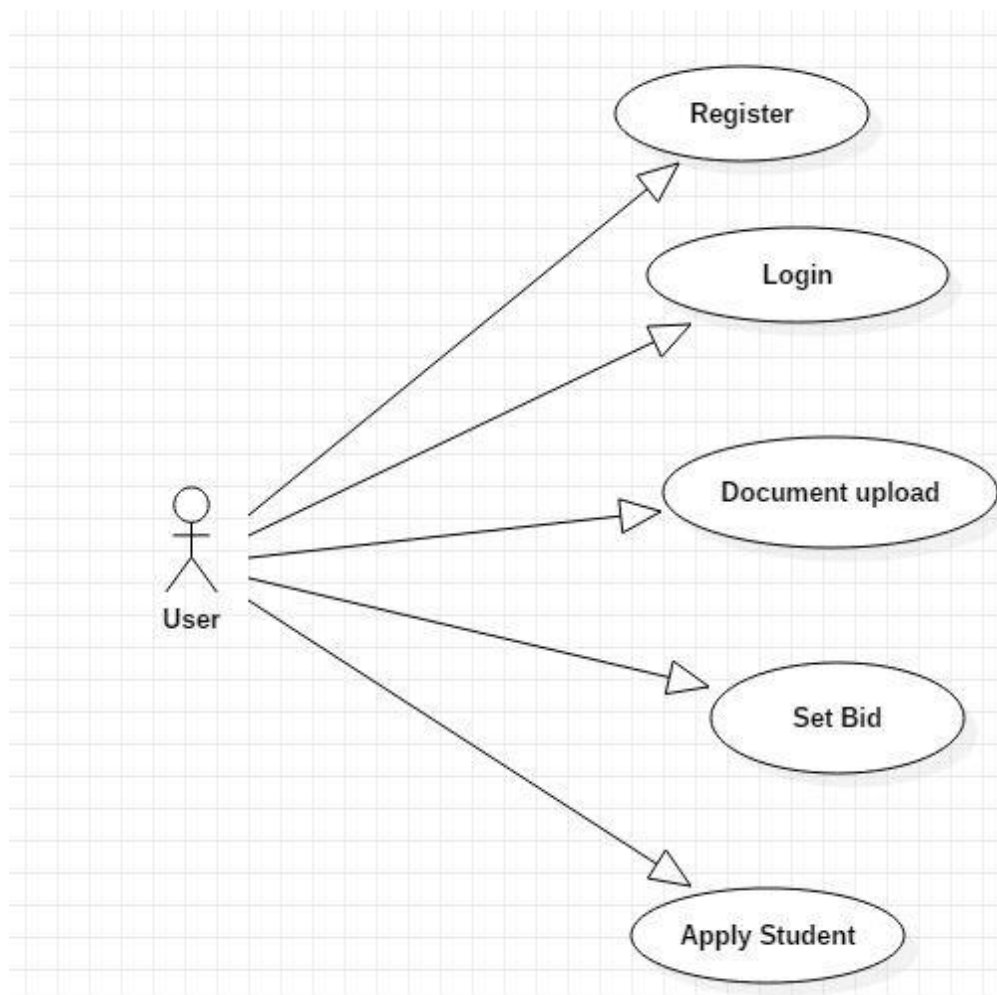
### **Use Case:**

Use case diagrams are usually referred to as behavior diagrams used to describe a set of actions that some system or systems (subject) should or can perform in

collaboration with one or more external users of the system each use case should provide some observable and valuable result to the actors or other stakeholders of the system.

In software and systems engineering, a use case is a list of actions or event steps typically defining the interactions between a role (known in the Unified Modeling Language (UML) as an actor) and a system to achieve a goal. The actor can be a human or other external system. In systems engineering, use cases are used at a higher level than within software engineering, often representing missions or stakeholder goals. The detailed requirements may then be captured in the Systems Modeling Language (SysML) or as contractual statements.

### Use Case Diagram



## **Fig: 4.2**Use Case Diagram

### **Brief Description**

The User can perform operation like setting Bid and Applying Students to other universities by seeing the bid.

## **Chapter-5**

### **DETAILED LIFE CYCLE OF THE PROJECT**

SDLC refers to a methodology for developing systems. It provides a consistent framework of tasks and deliverables needed to develop systems. The SDLC methodology may be condensed to include only those activities appropriate for a particular project, whether the system is automated or manual, whether it is a new system, or an enhancement to existing systems. The

SDLC methodology tracks a project from an idea developed by the user, through a feasibility study, systems analysis and design, programming, pilot testing, implementation, and post-implementation analysis. Documentation developed during the project development is used in the future when the system is reassessed for its continuation, modification, or deletion.

#### **SDLC Phases:**

Phases in SDLC are Planning, Analysis, Design, Implementation, Testing and Maintenance is a guideline for developing systems/software that involves following Phases:

1. Project planning, feasibility study: Establishes a high-level view of the intended project and determines its goals.
2. Systems Analysis, Requirements Definition: Refines project goals into defined functions and operation of the intended application. Analyzes end-user information needs.

3. Systems Design: Describes desired features and operations in detail, including screen layouts, business rules, process diagrams, pseudo



**Fig 5.1**SDLC Phases

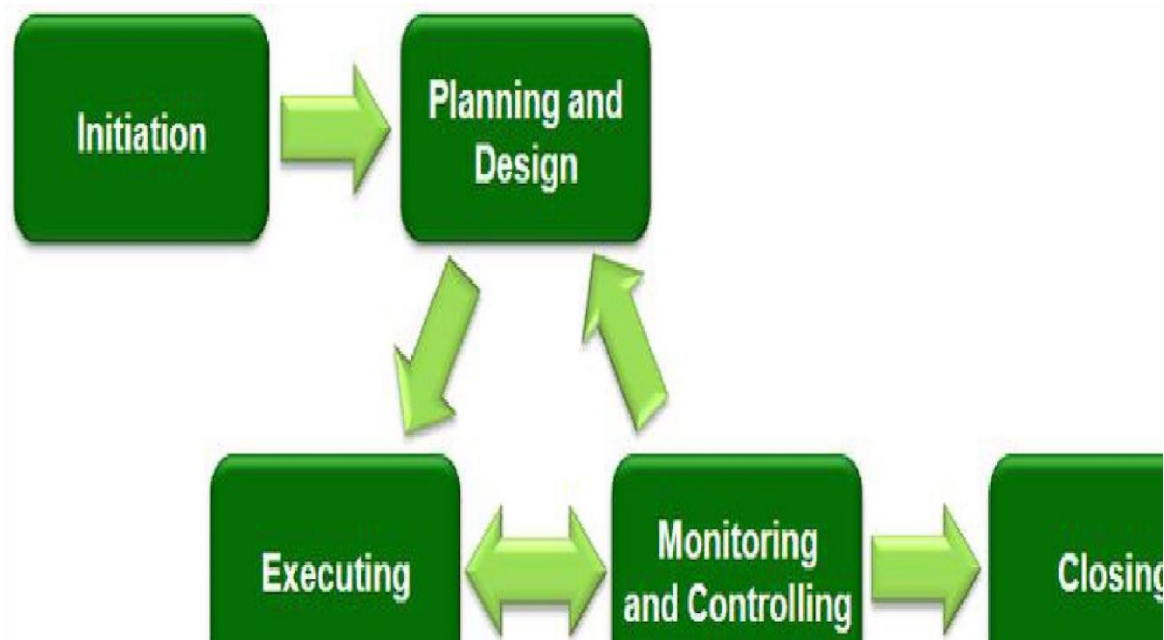
### **SDLC Models:**

There are various software development life cycle models defined and designed which are followed during the software development process. These models are also referred as Software Development Process Models". Each process model follows a Series of steps unique to its type to ensure success in the process of software development.

Following are the most important and popular SDLC models followed in the industry

—

- Waterfall Model
  - Iterative Model
  - Spiral Model
  - V-Model
  - Big Bang Model
1. Code and other documentation. A prototype should be developed during the logical design phase if possible. The detailed design phase modifies the logical design and produces a final detailed design, which includes technology choices, specifies system architecture, meets all system goals for performance, and still has all of the application functionality and behavior specified in the logical design.
  2. Implementation (Development): The real code is written here.
  3. Integration and Testing: Brings all the pieces together into a special testing environment, then checks for errors, bugs and interoperability.
  4. Acceptance, Installation, Deployment: The final stage of initial development, where the software is put into production and runs actual business.
  5. Maintenance: What happens during the rest of the software's life changes, correction, additions, moves to a different computing platform and more.



**Fig 5.2**Software Development Life Cycle

## Chapter-6

### SYSTEM PLANNING

Planning the system requires the user to define what the problem is. The planning may also include how the user would like to solve the problem. Defining the scope of the problem is also important in this stage as well. Defining the scope helps to prevent the project from scope once the problem is determined, and one or more solutions have been selected, planning to implement the solution begins. Multiple scenarios may be enacted to determine the best course of action for implementing the system. Course of action should be well documented and take into consideration a schedule showing anticipated start and completion times of activities (milestones) leading to the objectives, knowing expenditures required to achieve objectives, scheduling regular status reviews (are we on course?), anticipating any organizational restructuring to accommodate the objectives, anticipating and planning for mitigation of risks that may hinder achievements, implementing policies and procedures for decision making, and defining a standard level of performance.

Within the planning according to the John Sazinger "five of the main activities must exist" as he explains in his book the five activities should include:

- Produce the project schedule
- Confirm project feasibility
- Staff the project
- Launch the project



Why do plans fail? Some of the many reasons are:

- Goals/specifications are not understood.
- Objectives are too expensive for the time allotted.
- Budgets were not accurate.
- Project is understaffed or under skilled.
- Status reviews were not scheduled or insufficient.
- Poor morale (no commitment).

One of the most difficult decisions in planning is to know when to pull the plug on a project. This will require an effective control and monitoring system. If you cannot monitor a system you cannot control it. No organization wants to admit failure but there may come a point when a project can no longer be salvaged. This is especially critical with Information Technology projects because of rapidly changing technologies. Most managers are reluctant to prematurely terminate a project as careers and egos are at stake. The fallacy of sunk costs may play a role as well. The result is that projects continue beyond the point of no return. To avoid this problem, monitor and control systems must be put in place early during the planning stage. It is critical to define and enforce milestones where a project will be terminated if necessary. A saving grace is that because a project is terminated it doesn't make it a complete failure. Excessive cost is saved for the organization and management can walk away with lessons learned that can be applied to the next project. In general, there are two types of monitoring "INFORMAL" and "FORMAL". Informal are typically general meetings, email, and observing. The formal includes status reports,

scheduled milestones, audits, reviews, and generally more costly and are used during system development processes. Both systems can be used in combination and involve the questions: "what performance metrics to use" and "how often do reviews occur"? Attention and energy must be focused on identifying and correcting out-of-control processes.

**Chapter-7****SYSTEM DESIGN****7.1 UML MODELING:**

UNIFIED MODELING LANGUAGE (UML) is a standardized general-purpose modeling language in the field of object-oriented software engineering. The Unified Modeling Language includes a set of graphic notation techniques to create visual models of object-oriented software-intensive systems.

Unified Modelling Language is used to specify, visualize, modify, construct and document the artifacts of an object-oriented software-intensive system under development. The metamodeling architecture of Unified Modeling Language (UML) is defined in the Meta object function (MOF).

It is important to distinguish between the UML model and the set of diagrams of a system. A diagram is a partial graphic representation of a system's model. The model also contains documentation that drives the model elements and diagrams (such as written use cases).

**UML diagrams represent two different views of a system model**

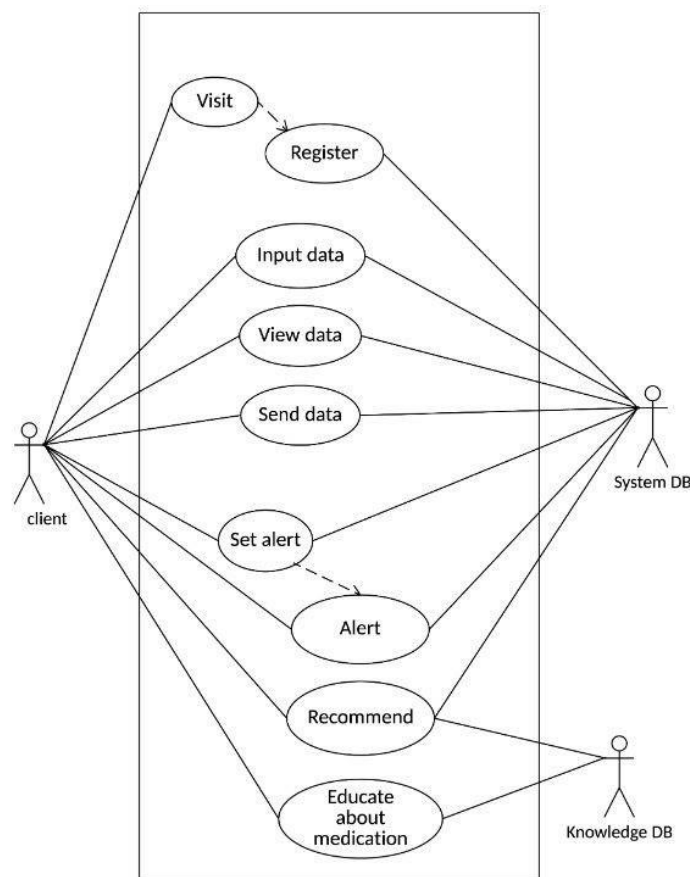
Static (or structural) view: emphasizes the static structure of the system using objects, attributes, operations and relationships. The structural view includes class diagrams and composite structure diagrams.

Dynamic (or behavioral) view: emphasizes the dynamic behavior of the system by showing collaborations among objects and changes to the internal

states of objects. This view includes sequence diagrams, activity diagrams and state machine diagrams.

## 1. USE CASE DIAGRAM:

Use case diagrams are usually referred to as behavior diagrams used to describe a set of actions that some system or systems (subject) should or can perform in collaboration with one or more external users of the system each use case should provide some observable and valuable result to the actors or other stakeholders of the system.

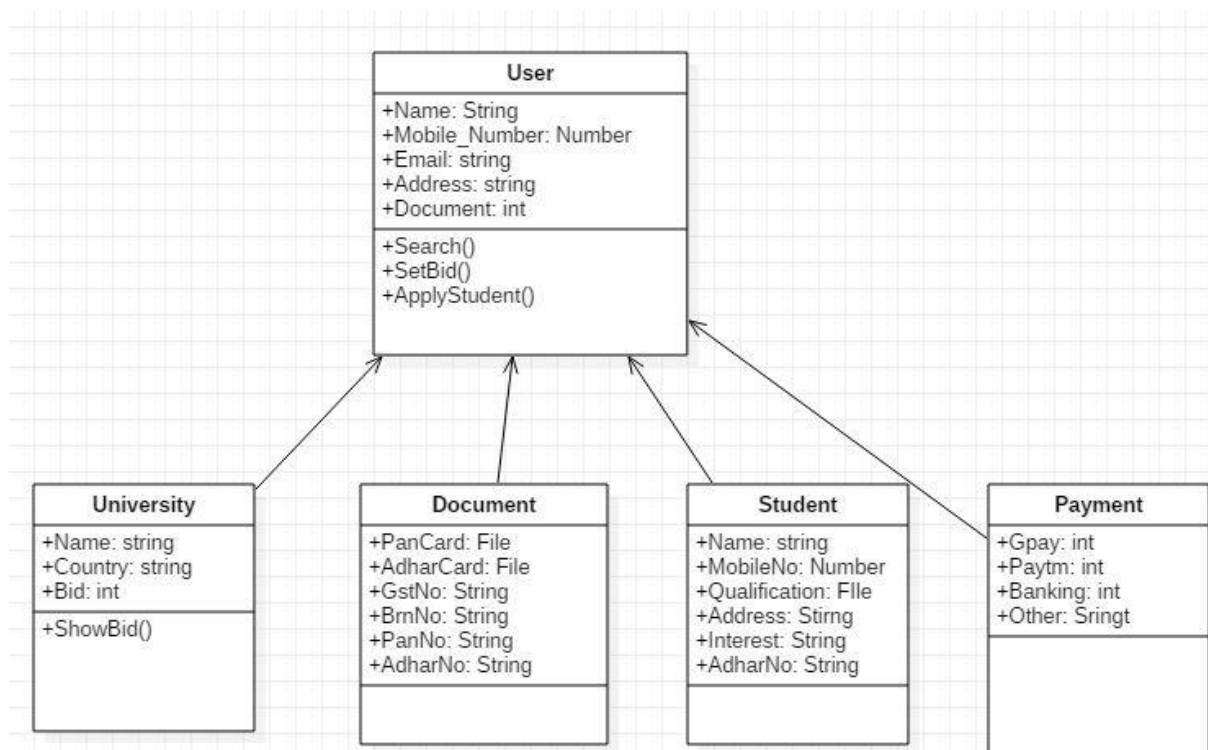


**Fig: 7.1** Use Case Diagram**Brief Description**

The Society Admin can perform Add/Remove of Security Staff, Add/Remove of Society Members, View Report and Manage Visitors.

**CLASS DIAGRAM:**

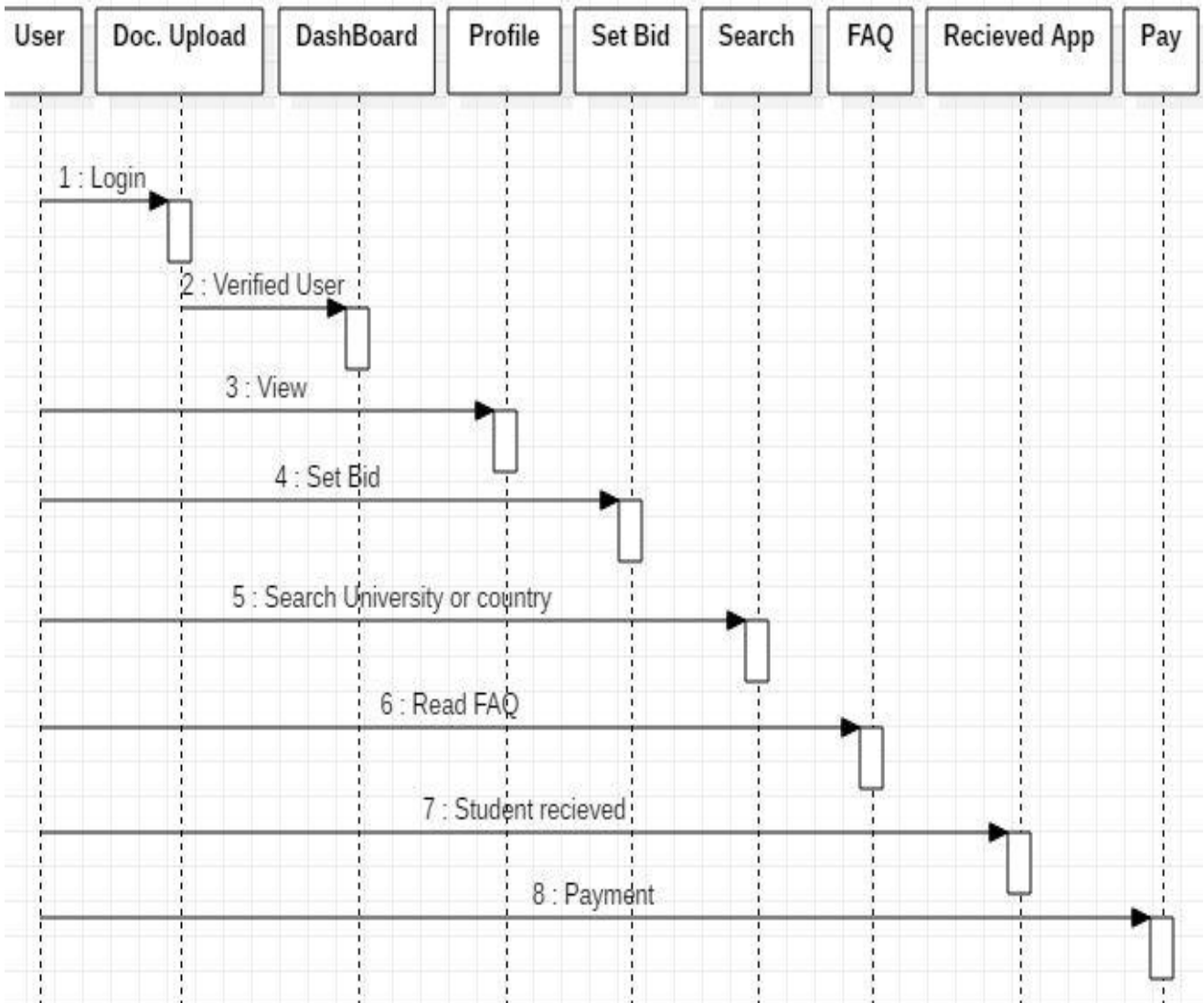
The class diagram is the main building block of object-oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code.



**Fig: 7.2**Class Diagram

## **2. SEQUENCE DIAGRAM:**

A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.



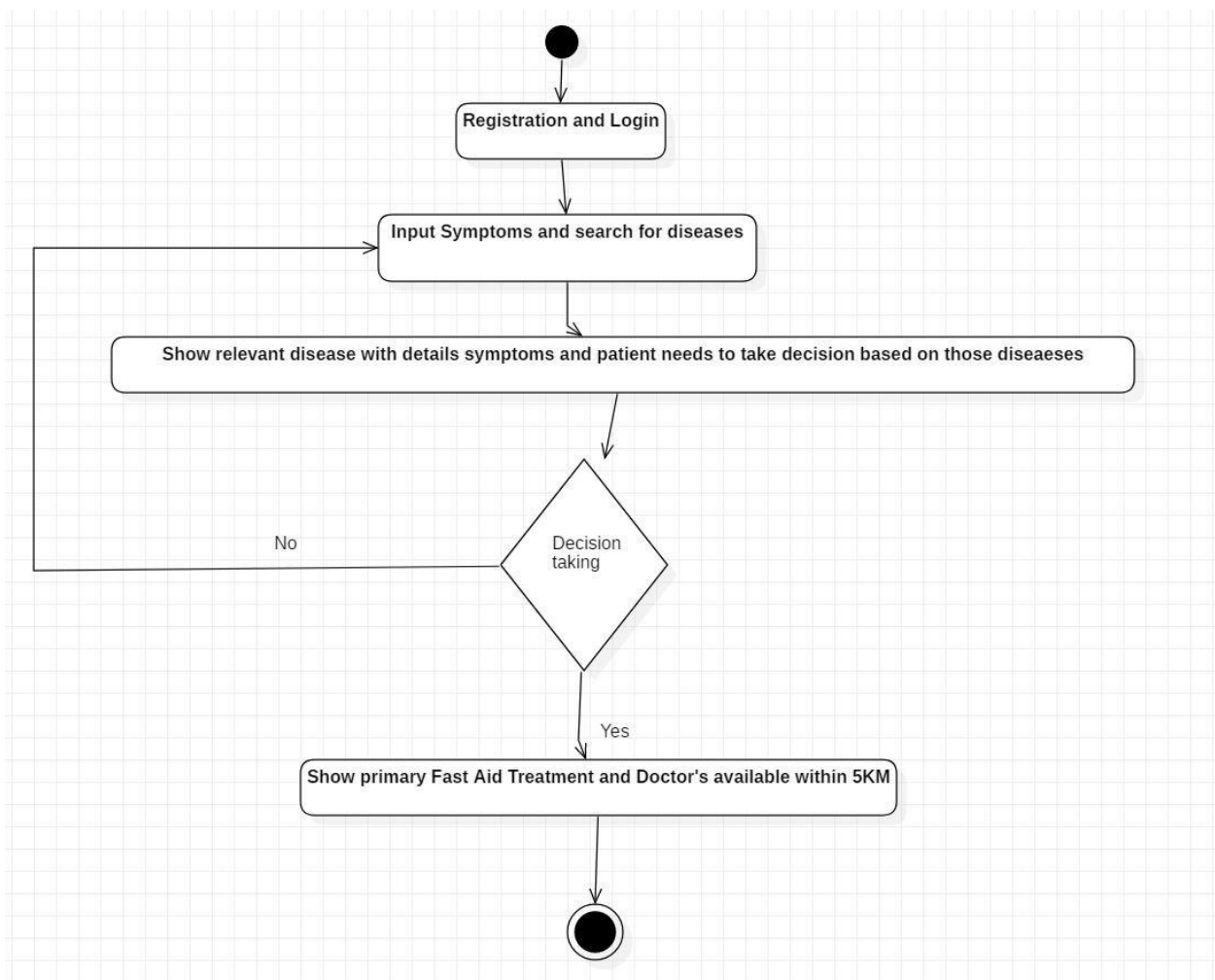
**Fig: 7.3**Sequence Diagram

### 3. ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram

shows the overall flow of control. Activity diagrams are constructed from a limited number of shapes, connected with Arrows. The most important shape types:

- Rounded rectangles represent activities;
- Diamonds represent decisions;
- Bars represent the start (split) or end (join) of concurrent activities;
- A black circle represents the start (initial state) of the workflow;
- An encircled black circle represents the end (final state).





**Fig: 7.4 Activity Diagram**

## **7.2 DATA FLOW DIAGRAM:**

A data flow diagram is graphical tool used to describe and analyze movement of data through a system. These are the central tool and the basis from which the other components are developed. The transformation of data from input to output, through processed, may be described logically and independently of physical components associated with the system. These are known as the logical data flow diagrams. The physical data flow diagrams show the actual implements and movement of data between people, departments and workstations. A full description of a system actually consists of a set of data flow diagrams. Using two familiar notations Yourdon, Gane and Sarson notation develops the data flow diagrams.

Each component in a DFD is labeled with a descriptive name. Process is further identified with a number that will be used for identification purpose. The development of DFD'S is done in several levels. Each process in lower level diagrams can be broken down into a more detailed DFD in the next level. The top-level diagram is often called context diagram. It consists of a single process bit, which plays vital role in studying the current system. The process in the context level diagram is exploded into other process at the first level DFD.

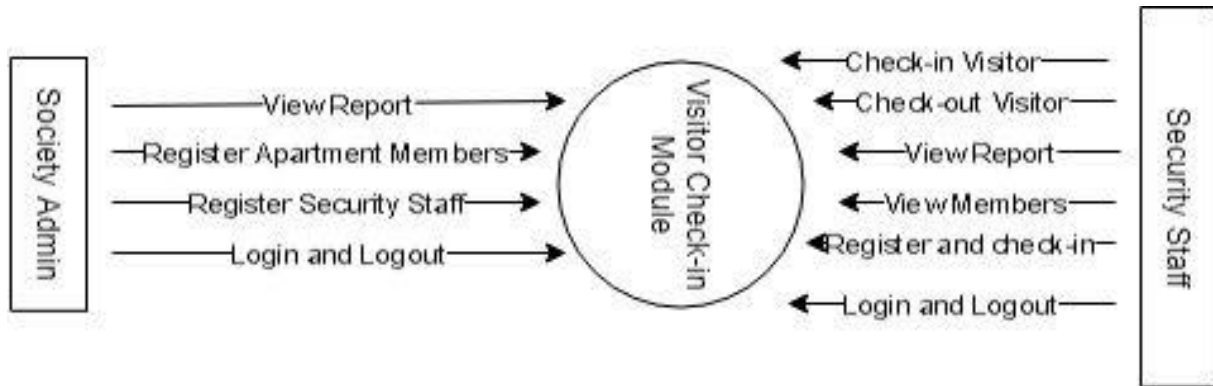
The idea behind the explosion of a process into more process is that understanding at one level of detail is exploded into greater detail at the next level. This is done until further explosion is necessary and an adequate amount of detail is described for

analyst to understand the process. Larry Constantine first developed the DFD as a way of expressing system requirements in a graphical form, this led to the modular design. A DFD is also known as a “bubble Chart” has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. So, it is the starting point of the design to the lowest level of detail. A DFD consists of a series of bubbles joined by data flows in the system.

**DFD symbols:**

- A square defines a source(originator) or destination of system data.
- An arrow identifies data flow. It is the pipeline through which the information flows.
- A circle or a bubble represents a process that transforms incoming data flow into outgoing data flows.
- An open rectangle is a data store, data at rest or a temporary repository of data.

**DFD for Visitor Check-in Module:**



**Fig: 7.7** DFD for Visitor Check-in Module

### 7.3 ENTITY RELATIONSHIP MODELING:

P. Chen introduced the E- R model. Entity – Relationship modeling is a details logical representation of the entities, associations and data elements for an organization or business area.

- **Entities**

An Entity is a person, place, thing or event of interest to the organization and about which data are captured, stored or processed.

- **Attributes**

Various types of data items that describe an entity are known as attributes.

- **Relationship**

An association of several entities in an Entity-Relation model is called relationship.

### Entity Relationship Diagram

The overall logical structure of a database can be expressed graphically by an entity relationship diagram.

**Name**

**Symbol**

**Meaning**

Rectangle Represents Entity set



Oval Represents



Attribute

Diamond Represents relationship among



entity set

Line



Sets and  
entity set to

relationship

**Three Types of relationship exist among entities:**

- **One to One Relationship (1:1)**

A one to one (1:1) relationship is an associated only between two entities.

- **One to Many Relationship (1: M)**

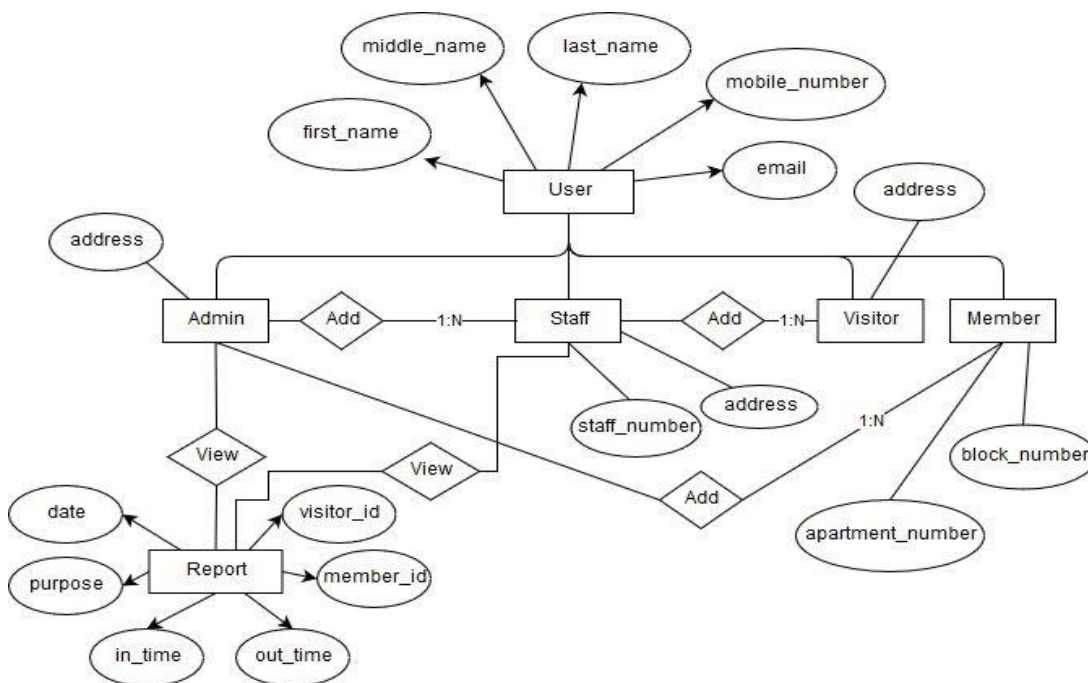
A one to many (1: M) relationship exist when one entity is related to more entities.

- **Many to Many Relationship (M: M)**

A many to many (M: M) relationship describe entities may have many relationships among each other.

## ER DIAGRAM

It is an abstract and conceptual representation of the data. Entity Relationship modeling is a Database Modeling Method, used to produce a type of conceptual schema. Entities: User, Administrator.



**Fig: 7.8** Entity Relationship Diagram

## Chapter-8

## Coding

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	num_preg	glucose_conc	diastolic_bp	thickness	insulin	bmi	diab_pred	age	skin	diabetes							
2	6	148	72	35	0	33.6	0.627	50	1.379	TRUE							
3	1	85	66	29	0	26.6	0.351	31	1.1426	FALSE							
4	8	183	64	0	0	23.3	0.672	32	0	TRUE							
5	1	89	66	23	94	28.1	0.167	21	0.9062	FALSE							
6	0	137	40	35	168	43.1	2.288	33	1.379	TRUE							
7	5	116	74	0	0	25.6	0.201	30	0	FALSE							
8	3	78	50	32	88	31	0.248	26	1.2608	TRUE							
9	10	115	0	0	0	35.3	0.134	29	0	FALSE							
10	2	197	70	45	543	30.5	0.158	53	1.773	TRUE							
11	8	125	96	0	0	0	0.232	54	0	TRUE							
12	4	110	92	0	0	37.6	0.191	30	0	FALSE							
13	10	168	74	0	0	38	0.537	34	0	TRUE							
14	10	139	80	0	0	27.1	1.441	57	0	FALSE							
15	1	189	60	23	846	30.1	0.398	59	0.9062	TRUE							
16	5	166	72	19	175	25.8	0.587	51	0.7486	TRUE							
17	7	100	0	0	0	30	0.484	32	0	TRUE							
18	0	118	84	47	230	45.8	0.551	31	1.8518	TRUE							
19	7	107	74	0	0	29.6	0.254	31	0	TRUE							
20	1	103	30	38	83	43.3	0.183	33	1.4972	FALSE							
21	1	115	70	30	96	34.6	0.529	32	1.182	TRUE							
22	3	126	88	41	235	39.3	0.704	27	1.6154	FALSE							
23	8	99	84	0	0	35.4	0.388	50	0	FALSE							
24	7	196	90	0	0	39.8	0.451	41	0	TRUE							
25	9	119	80	35	0	29	0.263	29	1.379	TRUE							

Fig 8.1 Data Set

The screenshot shows the Spyder Python IDE interface. The main editor displays a Python script for data visualization and model evaluation. The script includes the following code:

```

125
126
127 ##### Training And Testing #####
128
129 #Splitting the data in the ratio of 70% and 30%
130 #-----
131 from sklearn.model_selection import train_test_split
132 # from sklearn.cross_validation import train_test_split
133
134 feature_col_names = ['num_preg', 'glucose_conc', 'diastolic_bp', 'thickness', 'in
135 predicted_class_names = ['diabetes']
136
137 X = df[feature_col_names].values # predictor feature columns (8 X m)
138 y = df[predicted_class_names].values # predicted class (1=true, 0=false) column (
139 split_test_size = 0.30
140
141 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=split_test_si
142
143 print("{}:0.2f% in training set".format(len(X_train)/len(df.index) * 100))
144 print("{}:0.2f% in test set".format(len(X_test)/len(df.index) * 100))
145
146
147 #Verifying predicted value
148 #-----
149 print("Original True: {} ({}:0.2f%)" .format(len(df.loc[df['diabetes'] == 1]), (
150 print("Original False: {} ({}:0.2f%)" .format(len(df.loc[df['diabetes'] == 0]),
151
152 print("Training True: {} ({}:0.2f%)" .format(len(y_train[y_train[:] == 1]), (le
153 print("Training False: {} ({}:0.2f%)" .format(len(y_train[y_train[:] == 0]), (le
154
155 print("Test True: {} ({}:0.2f%)" .format(len(y_test[y_test[:] == 1]), (len(y_tes
156 print("Test False: {} ({}:0.2f%)" .format(len(y_test[y_test[:] == 0]), (len(y_tes
157
158
159 df.head()
160
161 #Impute with the mean
162 #-----

```

The IPython console on the right displays the output of the script, showing the distribution of the 'diabetes' variable and the results of the training and testing process:

```

...: #Verifying predicted value
...: #-----
...: print("Original True: {} ({}:0.2f%)" .format(len(df.loc[df['diabetes'] == 1]),
(len(df.loc[df['diabetes'] == 1])/len(df.index) * 100))
...: print("Original False: {} ({}:0.2f%)" .format(len(df.loc[df['diabetes'] == 0]),
(len(df.loc[df['diabetes'] == 0])/len(df.index) * 100))
...:
...: print("Training True: {} ({}:0.2f%)" .format(len(y_train[y_train[:] == 1]),
(len(y_train[y_train[:] == 1])/len(y_train) * 100))
...: print("Training False: {} ({}:0.2f%)" .format(len(y_train[y_train[:] == 0]),
(len(y_train[y_train[:] == 0])/len(y_train) * 100))
...:
...: print("Test True: {} ({}:0.2f%)" .format(len(y_test[y_test[:] == 1]),
(len(y_test[y_test[:] == 1])/len(y_test) * 100))
...: print("Test False: {} ({}:0.2f%)" .format(len(y_test[y_test[:] == 0]),
(len(y_test[y_test[:] == 0])/len(y_test) * 100))
...:
69.92% in training set
30.08% in test set
Original True: 268 (34.90%)
Original False: 500 (65.10%)
Training True: 188 (35.01%)
Training False: 349 (64.99%)
Test True: 80 (34.63%)
Test False: 151 (65.37%)

```

The status bar at the bottom indicates the file permissions (RW), end-of-lines (CRLF), encoding (ASCII), and the current position (Line: 160, Column: 1, Memory: 74 %).

Fig 8.2 Main Activity(data visualization)

The screenshot shows a Jupyter Notebook with two main sections. The left section contains Python code for training and testing an SVM model. The right section shows the IPython console output, which includes a warning about a DataConversionWarning and the execution of the test function.

```

20 sklearn.svm import SVC
21 model = SVC(kernel='linear')
22 svc=model.fit(X_train,Y_train)
23
24 #Save Model As Pickle File
25 with open('svc.pkl','wb') as m:
26     pickle.dump(svc,m)
27     test(X_test,Y_test)
28
29 #Test accuracy of the model
30 def test(X_test,Y_test):
31     with open('svc.pkl','rb') as mod:
32         p=pickle.load(mod)
33     pre=p.predict(X_test)
34     print (accuracy_score(Y_test,pre)) #Prints the accuracy of the model.
35
36
37 def find_data_file(filename):
38     if getattr(sys, "frozen", False):
39         # The application is frozen.
40         datadir = os.path.dirname(sys.executable)
41     else:
42         # The application is not frozen.
43         datadir = os.path.dirname(__file__)
44
45     return os.path.join(datadir, filename)
46
47
48 def check_input(data) ->int :
49     df=pd.DataFrame(data=data,index=[0])
50     with open(find_data_file('svc.pkl'),'rb') as model:
51         p=pickle.load(model)
52     op=p.predict(df)
53     return op[0]
54 if __name__=='__main__':
55     train()
56

```

The IPython console output shows the following code being executed:

```

.... datadir = os.path.dirname(__file__)
.... return os.path.join(datadir, filename)
....
.... def check_input(data) ->int :
....     df=pd.DataFrame(data=data,index=[0])
....     with open(find_data_file('svc.pkl'),'rb') as model:
....         p=pickle.load(model)
....         op=p.predict(df)
....         return op[0]
....
.... if __name__=='__main__':
....     train()
....
C:\Users\JAY\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
y = column_or_1d(y, warn=True)
0.7727272727272727
In [12]:

```

Fig 8.3 Finding Accuracy

The screenshot shows a Python script for a PyQt5 user interface. The script imports various PyQt5 modules and defines a class named Diabetes, which inherits from QWidget. The class has an \_\_init\_\_ method that initializes the UI components, including labels for patient details and input fields for various medical parameters.

```

5
6 import sys
7
8 from PyQt5.QtWidgets import QWidget, QLabel, QPushButton, QLineEdit, QVBoxLayout, QHBoxLayout, QApplication, QMainWindow
9 from PyQt5.QtGui import QDoubleValidator, QFont
10 from PyQt5.QtCore import Qt, QLine
11
12 import Diabetes
13
14 class Diabetes(QWidget):
15
16     def __init__(self) -> None :
17         super(Diabetes, self).__init__()
18         self.sub_head = QLabel("Patient's Details")
19         self.sub_head.setFont(QFont("Times",24, weight=QFont.Bold))
20         self.l0 = QLineEdit()
21         self.l1 = QLineEdit()
22         self.l2 = QLineEdit()
23         self.l3 = QLineEdit()
24         self.l4 = QLineEdit()
25         self.l5 = QLineEdit()
26         self.l6 = QLineEdit()
27         self.l7 = QLineEdit()
28         self.l8 = QLineEdit()
29         self.t0 = QLabel("Patient's Name:")
30         self.t1 = QLabel("Glucose conc:")
31         self.t2 = QLabel("Diastolic Bp:")
32         self.t3 = QLabel("Thickness:")

```

Fig 8.4(1) user interface code)

**Chapter-9****SYSTEM TESTING****What is ‘Software Testing’?**

Testing involves operation of a system or application under controlled conditions and evaluating the results. The controlled conditions should include both normal and abnormal conditions. Testing should intentionally attempt to make things go wrong to determine if things happen when they don't happen when they should. It is oriented to 'detection'.

**The need for Testing:**

- No matter how good a programmer is, no application will ever be one hundred percent correct. Testing was important to us in order to ensure that the application works as efficient as possible and conforms to the needs of the system.
- Testing was carried out throughout the development of the application, not just the application has been developed, as at this stage it took a great deal of effort to fix any bugs or design problems that were occurred.

The common view of testing held by users is that it is performed to prove that there are no errors in a program. This is extremely difficult since designer cannot prove to be one hundred percent accurate. Therefore, the most useful and practical approach is with the understanding that testing is the process of executing a program with explicit intention of finding errors that make the program fail.



Testing has its own cycle. The testing process begins with the product requirements phase and from there parallels the entire development process. In other words, for each phase of the development process there is an important testing activity. Successful testing requires a methodical approach. It requires focusing on basic critical factors:

- Planning
- Project and process control
- Risk management
- Inspections
- Measurement tools
- Organization and professionalism

**Test Plan:**

Before going for testing, first we have to decide upon the type of testing to be carried out. The following factors are taken into consideration:

- To ensure that information properly flows into and out of program
- To find out whether the local data structures maintains its integrity during all steps in an algorithm execution
- To ensure that the module operate properly at boundaries established to limit or restrict processing
- To find out whether error - handling paths are working correctly or not.
- To find out whether the values are correctly updated or not
- Check for validations

**Unit Testing:**

Unit or module testing is the process of testing the individual components (subprograms or procedures) of a program. The purpose is to discover discrepancies between the modules interface specification and its actual behavior. In our system each module must be tested independently for validation. In order to have more reliable and accurate prediction results, ensemble method is a well-proven approach practiced in research for attaining highly accurate classification of data by hybridizing different classifiers. The improved prediction performance is a well-known in-built feature of ensemble methodology. This study proposes a weighted vote-based classifier ensemble technique, overcoming the limitations of conventional DM techniques by employing the ensemble of two heterogeneous classifiers:

Naive Bayesian and classification via decision tree.

**Validation Testing:**

Validation testing provides the final assurance that software meets all functional, behavioral and performance requirement. The software once validated must be combined with other system elements. System testing verifies that as elements combine properly and that overall system function and performance is achieved.

**Integration Testing:**

Integration testing is the process of combining and testing multiple components together. The primary objective of integration testing is to discover errors in the interfaces between the components. In our system each of the modules mentioned above, are tested for checking the integration between them, after each of them are tested individually.

**White Box Testing:**

White box testing strategy deals with the internal logic and structure of the code.

White box testing is also called as glass, structural, open box or clear box testing.

The tests written based on the white box testing strategy incorporate coverage of the code written, branches, paths, statements and internal logic of the code etc. In order to implement white box testing, the tester has to deal with the code and hence is needed to possess knowledge of coding and logic i.e. internal working of the code.

White box test also needs the tester to look into the code and find out which unit/statement/chunk of the code is malfunctioning.

**Black box Testing:**

Black Box Testing is not a type of testing; it instead is a testing strategy, which does not need any knowledge of internal design or code etc. As the name "black box" suggests, no knowledge of internal logic or code structure is required. The types of testing under this strategy are totally based/focused on the testing for requirements and functionality of the work product/software application. Black box testing is sometimes also called as "Opaque Testing", "Functional/Behavioral Testing" and

"Closed Box Testing". The base of the Black box testing strategy lies in the selection of appropriate data as per functionality and testing it against the functional specifications in order to check for normal and abnormal behavior of the system.

The decision tree approach is more powerful for classification problems. There are two steps in this technique building a tree & applying the tree to the dataset. There are many popular decision tree algorithms CART, ID3, C4.5, CHAID, and J48. From these J48 algorithm is used for this system. J48 algorithm uses pruning method to build a tree. Pruning is a technique that reduces size of tree by removing over fitting data, which leads to poor accuracy in predications. The J48 algorithm recursively classifies data until it has been categorized as perfectly as possible. This technique gives maximum accuracy on training data. The overall concept is to build a tree that provides balance of flexibility & accuracy.

### TEST CASES:

**Table 8.1**  
**Test Cases**

<b>TEST REPORT WITH TEST DATA</b>		
<b>Project Name: Diabetes prediction using ML</b>		
<b>S no.</b>	<b>Testing Parameter</b>	<b>Observations</b>
A.	INTERFACE TESTING	
	1) User-friendliness	OK
	2) Consistent menus	OK

B.	<b>CONTROL FLOW TESTING</b> 1) IF-THEN-ELSE 2) DO WHILE 3) CASE-SWITCH	OK NA NA
C.	<b>VALIDATION TESTING</b> 1) Check for improper or inconsistent typing 2) Check for erroneous initialization or default values 3) Check for incorrect variable names 4) Check for inconsistent Data Types 5) Check for relational/arithmetic operators	OK OK OK OK OK
D.	<b>DATA INTEGRITY/SECURITY TESTING</b> 1) Data Insertion/ Deletion/ Updating 2) Boundary condition (Underflow, Overflow Exception) 3) Check for unauthorized access of data 4) Check for data availability	OK OK OK OK
E.	<b>EFFICIENCY TESTING</b> 1) Throughput of the system 2) Response time of the system 3) Online disk storage required by the system 4) Primary memory required by the system	OK OK OK OK

F.	<b>ERROR HANDLING ROUTINES</b>  1) Error description are intelligent/ understandable  2) Error recovery is smooth.	  OK  OK
----	---	----------------------

## Chapter-10

## PROJECT SCREENSHOTS

```

....:
Original True: 268 (34.90%)
Original False: 500 (65.10%)
Training True: 188 (35.01%)
Training False: 349 (64.99%)
Test True: 80 (34.63%)
Test True: 151 (65.37%)

```

Fig 10.1 Accuracy

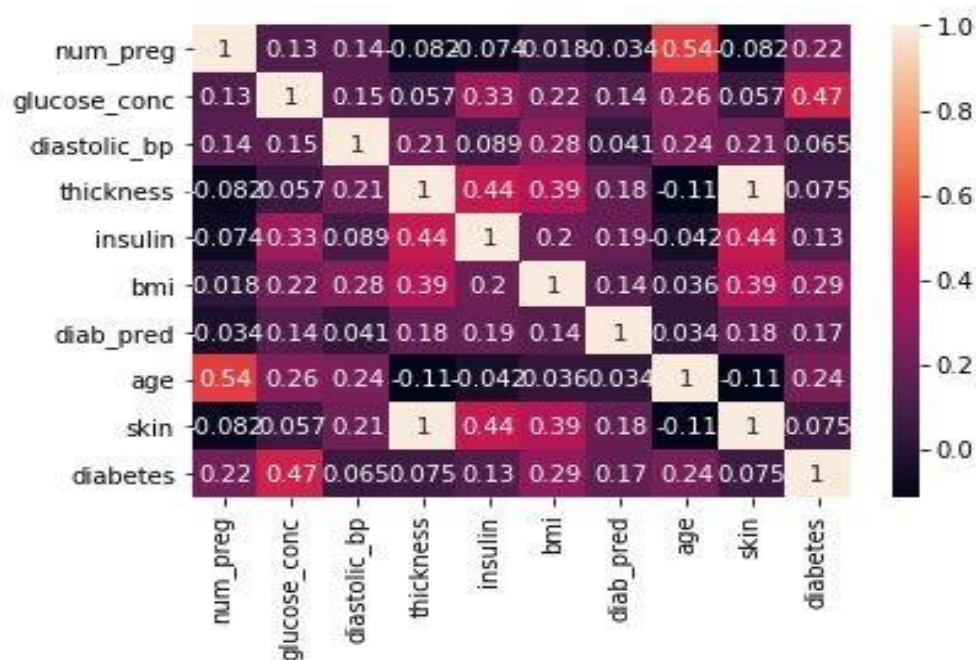
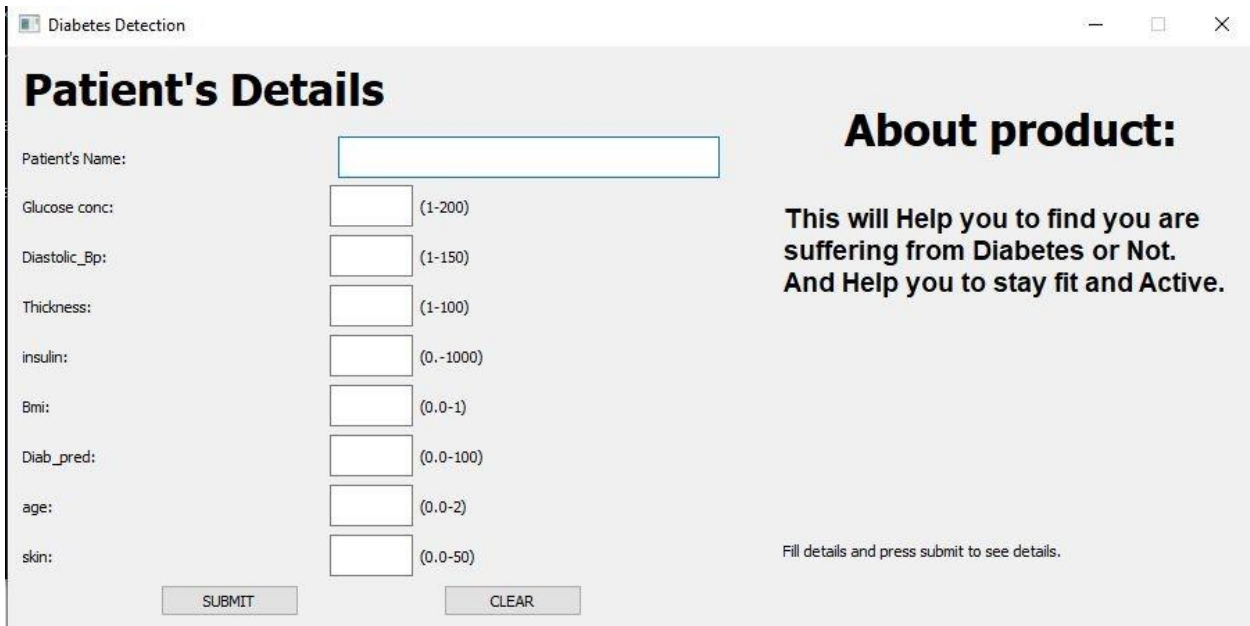
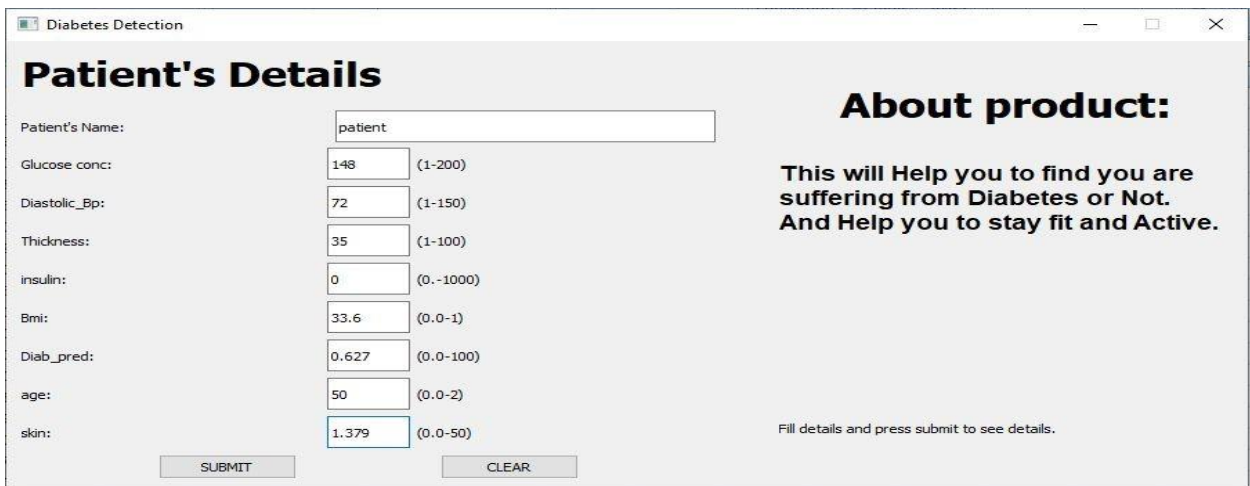


Fig 10.2 screenshot 2



The screenshot shows a window titled "Diabetes Detection" with a light gray background. On the left, under the heading "Patient's Details", there is a list of input fields: "Patient's Name:" (a long text box), "Glucose conc:" (a numeric box with range "(1-200)"), "Diastolic\_Bp:" (a numeric box with range "(1-150)"), "Thickness:" (a numeric box with range "(1-100)"), "insulin:" (a numeric box with range "(0.-1000)"), "Bmi:" (a numeric box with range "(0.0-1)"), "Diab\_pred:" (a numeric box with range "(0.0-100)"), "age:" (a numeric box with range "(0.0-2)"), and "skin:" (a numeric box with range "(0.0-50)"). At the bottom of this section are "SUBMIT" and "CLEAR" buttons. On the right, under the heading "About product:", there is a paragraph: "This will Help you to find you are suffering from Diabetes or Not. And Help you to stay fit and Active." Below this is a small text: "Fill details and press submit to see details."

Field	Range
Glucose conc:	(1-200)
Diastolic_Bp:	(1-150)
Thickness:	(1-100)
insulin:	(0.-1000)
Bmi:	(0.0-1)
Diab_pred:	(0.0-100)
age:	(0.0-2)
skin:	(0.0-50)

**Fig 10.3 Columns for Detail**

The screenshot shows the same "Diabetes Detection" window, but now the input fields are filled with sample data. The "Patient's Name" field contains the text "patient". The other fields contain numeric values: "Glucose conc:" is 148, "Diastolic\_Bp:" is 72, "Thickness:" is 35, "insulin:" is 0, "Bmi:" is 33.6, "Diab\_pred:" is 0.627, "age:" is 50, and "skin:" is 1.379. The "SUBMIT" and "CLEAR" buttons remain at the bottom. The "About product:" section on the right is unchanged.

Field	Value	Range
Glucose conc:	148	(1-200)
Diastolic_Bp:	72	(1-150)
Thickness:	35	(1-100)
insulin:	0	(0.-1000)
Bmi:	33.6	(0.0-1)
Diab_pred:	0.627	(0.0-100)
age:	50	(0.0-2)
skin:	1.379	(0.0-50)

**Fig 10.4 Patient's detail**



Diabetes Detection

## Patient's Details

Patient's Name:

Glucose conc:  (1-200)

Diastolic\_Bp:  (1-150)

Thickness:  (1-100)

insulin:  (0.-1000)

Bmi:  (0.0-1)

Diab\_pred:  (0.0-100)

age:  (0.0-2)

skin:  (0.0-50)

## Reports

**Patient's name: patient**  
**Body Glucose: 148**  
**Blood pressure: 72**  
**Mean Perimeter: 35**  
**insulin taken: 0**  
**H\*w ratio: 33.6**  
**Diabetes pedigree: 0.627**  
**your curret age: 50**  
**skin fold: 1.379**

**Our suggests patient DOES SUFFER FROM Diabetes**  
**Please get checked soon.**

For More Go to AI Doctor

**Fig 10.5 Report**

## Chapter-11

### CONCLUSION AND FUTURE WORK

The proposed system is GUI-based, user-friendly, scalable, reliable and an expandable system. The proposed working model can also help in reducing treatment costs by providing initial diagnosis in time. The model can also serve the purpose of training tool for medical students and will be a soft diagnosis tool available for physician and cardiologist. General physicians can utilize this tool for initial diagnosis of patients. There are many possible improvements that could be explored to improve the scalability and accuracy of this prediction system. As we have developed a generalized system, in future we can use this system for analysis of different data sets. The performance of the health's diagnosis can be improved significantly by handling numerous class labels in the prediction process, and it can be another positive direction of research. In DM warehouse, generally, the dimensions of the database is high, so identification and selection of significant attribute for better diagnosis of heart disease are very challenging tasks for future research.

The proposed system is GUI-based, user-friendly, scalable, reliable and an expandable system. The proposed working model can also help in reducing treatment costs by providing Initial diagnostics in time. The model can also serve

the purpose of training tool for medical students and will be a soft diagnostic tool available for physician and cardiologist. General physicians can utilize this tool for initial diagnosis of cardio-patients. There are many possible improvements that could be explored to improve the scalability and accuracy of this prediction system. As we have developed a generalized system, in future we can use this system for the analysis of different data sets. The performance of the health's diagnosis can be improved significantly by handling numerous class labels in the prediction process, and it can be another positive direction of research. In DM warehouse, generally, the dimensionality of the heart database is high, so identification and selection of significant attributes for better diagnosis of heart disease are very challenging tasks for future research.

**References:**

- Tensorflow Documentation at <https://www.tensorflow.org>
- IEEE. IEEE STD 830-1998
- IEEE recommended practice for software requirements specifications.
- IEEE computer society, 1998.
- Data set From Kaggle.

