**TABLE OF CONTENTS**

**ACKNOWLEDGEMENT**

|  |  |  |
| --- | --- | --- |
| **SERIAL**  **NO** | **TITLE** | **PAGE NO** |
| 1 | List of Tables |  |
| 2 | List of Figures |  |
| 3 | Abbreviations |  |
| 4 | Abstract | 1 |
| 5 | **CHAPTER 1 (Introduction)** | **2-4** |
|  | 1.1 Introduction | 3 |
|  | 1.2 Problem Statement and Project Relevance | 3 |
|  | 1.3 Aim and Scope of Project | 3 |
|  | 1.4 Existing System | 3 |
|  | 1.5 Proposed System | 4 |
| 6 | **CHAPTER 2 (Requirement Elicitation)** | **5-9** |
|  | 2.1 Feasibility Analysis | 6 |
|  | 2.2 Requirement Study | 7 |
|  | 2.3 Requirement Analysis and Classification | 7 |
| 7 | **CHAPTER 3 (Project Design)** | **10-30** |
|  | 3.1 Overall Project Planning | 11 |
|  | 3.2 Module Description | 12 |
|  | 3.3 Sequence Diagram | 14 |
|  | 3.4 DFD | 15 |
|  | 3.5 ER Diagram | 20 |
|  | 3.6 Usecase Diagram | 21 |
|  | 3.7 Activity Diagram | 22 |
|  | 3.8 Table Design | 24 |
| 8 | **CHAPTER 4 (System Requirements)** | **31-32** |
|  | 4.1 Software Requirements | 32 |
|  | 4.2 Hardware Requirements | 32 |
|  | 4.3 Development Environment | 32 |
|  | 4.4 Deployment Environment | 32 |
| 9 | **CHAPTER 5 (Project Implementation)** | **33-44** |
|  | 5.1 Implementation Details | 34 |
|  | 5.2 ScreenShots | 38 |
| 10 | **CHAPTER 6 (Testing)** | **45-49** |

|  |  |  |
| --- | --- | --- |
|  | 6.1 Testing methodologies and Strategies | 46 |
|  | 6.2 Unit Test Reports | 46 |
|  | 6.3 Integration Test Reports | 48 |
|  | 6.4 System Test Reports | 49 |
|  | 6.5 Performance Test Reports | 49 |
| 11 | **CHAPTER 7 (Conclusion and Future Enhancements**) | **50-52** |
| 12 | Bibliography | 53 |
| 13 | Appendices |  |

## LIST OF TABLES

* Table 2.3.1 MoSCoW Analysis
* 3.7.1 Table name : Companies
* 3.7.2 Table name : Aptitude test
* 3.7.3 Table name : HR section
* 3.7.4 Table name : college
* 3.7.5 Table name :interview schedule
* 3.7.6 Table name : feedback
* 3.7.7 Table name : placement
* 3.7.8 Table name : student
* 3.7.9 Table name : mock interview
* 3.7.10 Table name : college admin
* 3.7.11 Table name : real interview
* 3.7.12 Table name: test
* 6.2.1 Unit Test Report
* 6.3.1 Integration Test Report

# LIST OF FIGURES

Figure 1: Use Case diagram……………………………….…………………………………

Figure 2: Sequence diagram………………………………………………………………..

Figure 3: Activity diagram…………………………………………………………………

Figure 4: Data Flow diagram………………………………………………………………

Figure 5: ER diagram……………………………………………………………………

Figure 6: ScreenShot………………………………………………………………….

LIST OF ABBREVIATIONS

* DFD – Data Flow Diagram
* ERD – Entity-Relationship Diagram
* DB – Database
* HCL – Hardware Compatibility List
* MFS – Microsoft Foundations Classes

## ABSTRACT

Here we introduce an application where college placement cell admin can register and login, he can provide username and password to each students. Students have the provision to add there resumes and other academic details. CF management team will filter the resumes and conducts online aptitude test for 200 associate companies through this. Also they provide online communication classes and mock interviews. The different modules are,

* CF Management
* College Placement Admin
* Students

**CHAPTER** **I**

**INTRODUCTION**

## INTRODUCTION

Veterinary clinics offered various kind of services to pets such as treatments, vaccination and deworming, de-sexing, surgery cases, grooming, hostel services and boarding. People who have their pets will visit the clinic either to get the subsequent treatments or to make an appointment for their pets. So, pet owner have to fill in the registration form first in order to register before get a consultation with veterinarian. With the number of pets that visits to the clinic, an interactive and effective approach should be applied to build the system.

The project” VetCare” could help to manage the management of the clinics, since this system will response in real-time and easy to be managed. This system helps clinic to keep records of pet registration, treatment histories, setting up appointments and list down prescription to pets. This system helps to reduce problems when using manual system such as pet registration using form, pet’s and appointment records using calendar book and diagnosis pets’ record using form. Besides, this system also convenient as it can help the clinic to handle and manage their activity while at the same time allows staffs to generate report for the pet owner and pet who comes, for clinic references purposes. By using this kind of system, all the information of pets will be stored into database in a systematic and efficient. As all the records are kept in database, this can reduce the problem of lost data. This system connects all the Veterinary clinics where users can find easily nearby them.

In some cases of emergency users or persons cannot able to treat sick animals immediately to right doctors and specialist. The users have to do one time registration and after registered users can easily contact to the doctors and also see the details of doctor. The findings are that the research is broad and fragmented, characterized by several core complexities including confirmation bias, trivialization and reductionism. There is evidence that pets play a positive role in mental wellbeing for many people, including some who experience major social disadvantages.

## PROBLEM STATEMENT AND PROJECT RELEVANCE

The existing system is not fully digitalized and automated. This in turn wastes lots of valuable time and money .To avoid this new system is proposed. The new system provides a user friendly environment.

## PROJECT SCOPE

This is a software application that helps in maintaining issues related to care of animals. The aim of this application is to make the existing system automate because it is a manual system of maintaining records of doctors, pets can also use this software to make the manual system of marinating of data is to automate system. It includes the maintenance of liquid flows and stock.

## EXISTING SYSTEM

Currently there is no such system available in the market. The existing system is manual system. Farmer’s have to bring their pets to the clinic for their treatment. Everybody doesn’t feel good to take his pets to the doctor at initial stages. It’s time consuming and not in reach of distant people.

## PROPOSED SYSTEM

The proposed work “VetCare” that uses an online platform that makes the task of making an appointment from the doctor easy and reliable for the users. The project will aim to establish and develop an online platform that will allow the customers to set an appointment with the veterinary. The system helps to manage all the activities taking place at the VetCare Clinic and can automate the system, generate reports etc. the system is accessed by mainly 3 users Admin of the clinic, Doctors and users.

The proposed work “VetCare” that uses an online platform that makes the task of making an appointment from the doctor easy and reliable for the users. The project will aim to establish and develop an online platform that will allow the customers to set an appointment with the veterinary. The system helps to manage all the activities taking place at the VetCare Clinic and can automate the system, generate reports etc. the system is accessed by mainly 3 users Admin of the clinic, Doctors and users.

**ADVANTAGES**

* Only authorized users can log into the system
* Reduces paperwork
* Provides efficient communication mechanism with user friendly interface
* Reliable
* Portable
* Secured
* Searches and findings are very quick

.

# CHAPTER II

**REQUIREMENT ELICITATION**

## FEASIBILITY ANALYSIS

Feasibility study is a test of a system proposal according to its workability, impact on the organization, ability to meet users need and effective use of resources. The main aim of the feasibility study is to determine whether it would be financially and technically feasible to

develop the product. The feasibility study was divided into four: Technical, Economical, Operational and Behavioral. It is summarized below.

## Economic Feasibility

The economic analysis is to determine the benefits and savings that are expected from a candidate system and compare them with the costs. The system is economically feasible, as the s/w does not require any additional hardware resources other than a computer. It only requires an internet connection which works efficiently.

The overall process of the organization is performed through these software.It will reduce the staff number and also avoid stationary items such as books,pens,files etc due to this the software is enconomically feasible

## Technical Feasibility

It centers on the existing computer system and to what extent it can support the proposed addition. This involves financial considerations to accommodate technical enhancement. If the budget is a serious constraint, then the project is judged not feasible. This s/m is technically feasible since it only requires minimal technical requirements such as an IIS server and a browser.

This system is light weight software to handle the entire process of the organization. This software can executed in any machine or platform without changing the internal structure due to this the software is technically feasible.

## Operational Feasibility

The system operation is the longest phase in the development life cycle of a system. So, operational feasibility should be given much importance. The users of the system don’t need thorough training on the system. It has a user friendly interface and is easy to work on it

This software is applicable to any type of users from anywhere. Anyone can execute this software without any computer knowledge. The well designed forms and calculation process are very easy to the user. The registration,search and view operations can be accessible to the

user with a single mouse click due to this software is operationally feasible.

## REQUIREMENT STUDY

Requirement study also called requirements engineering is the process of determining user expectations for a new or modified product. These features called requirements must be quantifiable, relevant and detailed. In software engineering, such requirements are often called functional specifications. Requirements study is an important aspect of project management. Requirement study involves frequent communication with system users to determine specific feature expectations, resolution of conflict or ambiguity in requirements as demanded by the various users or groups of users avoidance of feature creep and documentation of all aspects of the project development process from start to finish. Energy should be directed towards ensuring that the final system or product conforms to client needs rather than attempting to mold user expectations to fit the requirements. Requirements study is a team effort that demands a combination of hardware, software and human factors engineering as well as skills in dealing with people.

## REQUIREMENT ANALYSIS AND CLASSIFICATION

Software requirement analysis is the starting point of the software development activity. Little importance was given to this phase in the early days of software development, but as the systems grew more complex it became evident that the goals of the entire system cannot be easily comprehended. Hence the need for the requirement analysis phase arose. Now, for large software systems, requirement analysis is the most difficult activity and also the most error prone. The requirement specification phase consists of two basic activities:

* Problem or requirement analysis
* Requirement Specification

The goal of problem analysis activity is to understand such different aspects as the requirements of the problem, its context, and how it fits in the clients organization. In the second activity the understood problem is specified or written, producing the SRS. The production of SRS is the goal of the requirement analysis phase. The software requirement specification (SRS) is a means of translating the ideas in the minds of clients into a formal document. This document forms the basis of development and software validation. The basic reason for the difficulty in software requirements specification comes from the fact that there are three interested parties – the client, the end users and the software developer. The requirements document has to be such that the client and the user can understand it easily and the developers can use it as a basis for software development. Due to the diverse parties involved in software requirement specification, a communication gap exists. This gap arises when the client does not understand the software or the software development process or when the developer does not understand the clients problem and application area. SRS

bridges this communication gap. A good SRS provides many benefits. Some

Establishes the basis of agreement between client and supplier on what the software product will do.

1. Reducing the development cost
2. Providing a reference for validation of the final product. The SRS assists the client in determining if the software meets the requirements.

Problem analysis is done to obtain a clear understanding of the needs of the clients and the users, and what exactly is desired from the software. Analysis leads to the actual specification. People performing the analysis called analysts, are also responsible for specifying the requirements. The software project is initiated by the client‟s needs. In the beginning these needs are in the minds of various people in the client organization. During the process of analysis, a massive amount of information is collected and this information is organized effectively for the software development. Also during the analysis the interpersonal skills of the analyst are more important than the technical skills. Thus during analysis good communication and interpersonal skills are perhaps the most important tools that the analyst needs. Once the analysis is complete, the requirements must be written or specified. The final output is the Software Requirement Specification Document (SRS)

**REQUIREMENT CLASSIFICATION**

The requirement classification is based on MoSCoW standards. MoSCoW Analysis is a prioritization technique used in the business analysis and software development to reach a common understanding with stakeholders on the importance they could and won‟t. There are very specific definitions for each of these four categories.

**Must (M):** These requirements are must have. Think of them as very high priority requirements for the project. They must be part of the final solution in order for that solution to be considered successful.

**Should(S):** These requirements are high priority requirements and are as important as the requirements in the above category. However, there might be work arounds that satisfy these requirements or they may not be as time critical.

**Could(C):** These desirable requirements are of lesser priority and are nice to have capabilities in the solution. They really don‟t affect anything else in the solution one way or the other and will be included if time and resources permit.

**Won’t (W):** These requirements will not be implemented in a given solution release. They may be considered for inclusion in a future release (future requirements that stakeholder would like to have) or be omitted from the solution altogether.

|  |  |  |
| --- | --- | --- |
| **REQ ID** | **REQUIREMENT DESCRIPTION** | **CLASSIFICATION** |
| Req 1 | Student can log into the system and register | **M** |
| Req 2 | Student can have username and password | **S** |
| Req 3 | Student can view their placement details | **M** |
| Req 4 | Student can search there desired companies according to their needs | **S** |
| Req 5 | College placement admin can register and log into the system | **M** |
| Req 6 | College placement admin can add student details | **S** |
| Req 7 | College placement admin can search companies | **M** |
| Req 8 | College placement admin can request interviews | **Co** |
| Req 9 | College placement admin schedule tests | **S** |
| Req 10 | Student can give feedback | **W** |
| Req 11 | The student can attend interviews and mock tests | **S** |

**Fig 2.3.1 MoSCoW Analyasis**

# CHAPTER III

# PROJECT DESIGN

## OVERALL PROJECT PLANNING

The design phase is the second phase in the system development life cycle. In this phase computer information system is designed in detail from the system specification generated during the study phase. The principle activities performed during the design phase are allocation of functions, identification of testing requirements, output screen and report design, input design and file design. In design phase, the analyst has the task of developing a detailed design of the system including layouts for all inputs, file and outputs. In the design phase the detailed design of the system selected in the study phase is accomplished and user oriented performance specification is converted into a technical design specifications. The system design is the process of developing specification for a candidate system that meet the criteria established in system analysis. The principle activities performed during design phase include the allocation of function of between computer programs, equipment, manual operations, design of database used by the computer programs, specification of the requirement for input, processing and output and the definition of system and computer program test requirements. There are many aspects to consider in the design of a piece of software. The importance of each should reflect the goals the software is trying to achieve. Some of these aspects are:

* Extensibility- New capabilities can be added to the software without major changes to the underlying architecture.
* Robustness- The software is able to operate under stress or tolerate unpredictable for invalid input.
* Robustness- The software is able to perform the required function under stated conditions for a specified period of time.
* Fault Tolerance-the software is resistant to and able to recover from component failure.
* Security– the software is able to withstand hostile acts and influences.
* Maintainability– the software can be restored to a specified condition within a specified period of time
* Compatibility– the software is able to operate with the other products that are designed for interoperability with other product.

In Camerinfolks Placement Cell , we use Use case diagram, Sequence diagram, and Schema Design. A use case describes event sequences for an actor to use the system. Sequence Diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. Schema diagram represents the elements of a system using abstract and graphic symbols: During the design phase of the project Camerinfolks Placement Cell the following design methodologies have been used. They are:

1. Sequence Diagram
2. Data Flow Diagram
3. ER- Diagram

4.Usecase Diagram

5. Activity Diagram

## MODULE DESCRIPTION

The main modules are :

**ADMIN**

Admin has the full access to the system which means he is able to manage any activitywith regard to the system. He is the highest privileged user who can access to thesystem.

***Keyfunctions:***

* View pet owners
* View complaints
* View clinics
* View doctors
* Generate report.

**Clinic**

Clinic need to provide the doctors details for the clinic

***Keyfunctions:***

* View pet owners details.
* Manage doctors.
* View appointments
* Manage report

**Doctor**

* View appointments
* View pet owners
* Chat with pet owners

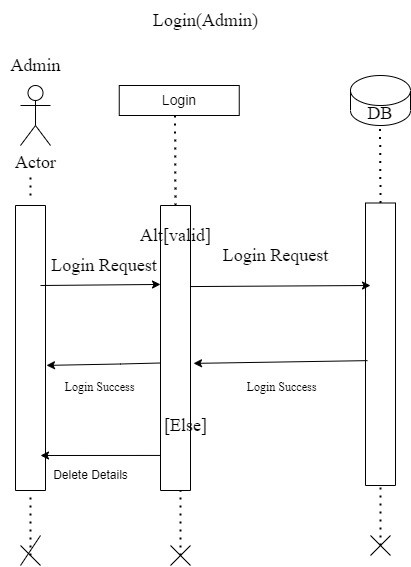
Pet owners

* View clinic
* View doctors
* Manage pet information’s.

Take appointment

## SEQUENCE DIAGRAM

A sequence diagram shows, as parallel vertical lines, different processes that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur



**Fig : 3.3.1- SEQUENCE DIAGRAM –CF Management Admin**

**3.4 DATA FLOW DIAGRAM**

A data flow diagram is a graphical technique that depicts information flow and transforms that are applied as data move from input to output. The DFD is used to represent increasing information flow and functional details. A level 0 DFD also called a fundamental system model represents the entire software elements as a single bible with input and output indicated by incoming and outgoing arrows respectively.

Data flow diagrams are the pictorial way of showing the flow of data into, around and out of the system. The users can understand them easily and there is no way to misinterpretation than textual description. With a dataflow diagram, users are able to visualize how the system will operate, what the system will accomplish and how the system will be implemented.

A data flow diagram illustrates the processes, data stores, and external entities in a business or other system and the connecting data flows. The four components of a data flow diagram (DFD) are:

* External Entities/Terminators/Sources/Sinks: Represented by a square
* Processes: Represented by a circle
* Data Flows: Represented by an arrow.
* Data Source: Represented by two parallel lines, connected by a vertical line

### External Entities/Terminators

Terminators represent where information comes from and where it goes. In designing a system, we have no idea about what these terminators do or how they do it

### Processes

Modify the inputs in the process of generating the outputs.

### Data Stores

Represent a place in the process where data comes to rest. A DFD does not say anything about the relative timing of the processes, so as data

store might be a place to accumulate data over a year for the annual accounting process.

### Data Flows

How data moves between terminators, processes, and data stores? And what are those data

## DATAFLOW DIAGRAMS

### Context Level

**LEVEL -0 CONTEXT LEVEL**



**LEVEL 1 ADMIN**



**CLINIC**



**DOCTOR**



**PET OWNER**



* 1. **ENTITY RELATIONSHIP DIAGRAM**



**2.10 USE CASE DIAGRAM**

**Fig : 3.4.1 – Context level**

**LEVEL 1 CF Manangement**

**Fig : 3.4.2 – Level 1 CF Mangement**

**LEVEL 1.1 College Placement Admin**

**Fig: 3.4.3 - Level 1.1 College Placement Admin**

**LEVEL 1.2**

**Fig : 3.4.5 – Level 1.2- Student**

* 1. **ER-DIAGRAM**

The entity-relationship diagram is a data modeling technique that graphically represent an information system entities and relationships between those entities. An ER-diagram is a conceptual and representational model of data which is used to represent the system framework infrastructure.

The ER-diagram contains following elements:

* Entities
* Relationships
* Attributes

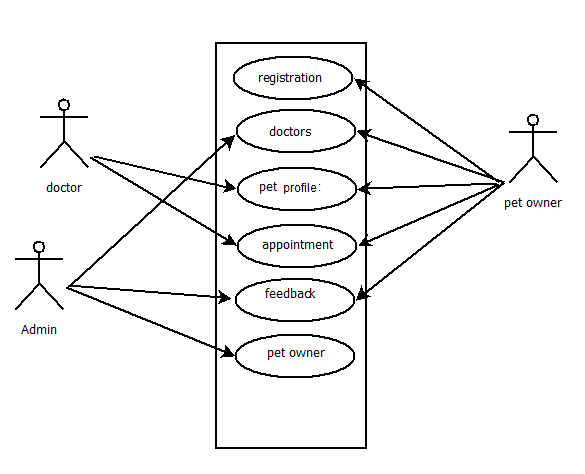
## Entity Relationship Diagram



### Fig : 3.5 – ER-DIAGRAM

* 1. **USECASE DIAGRAM**

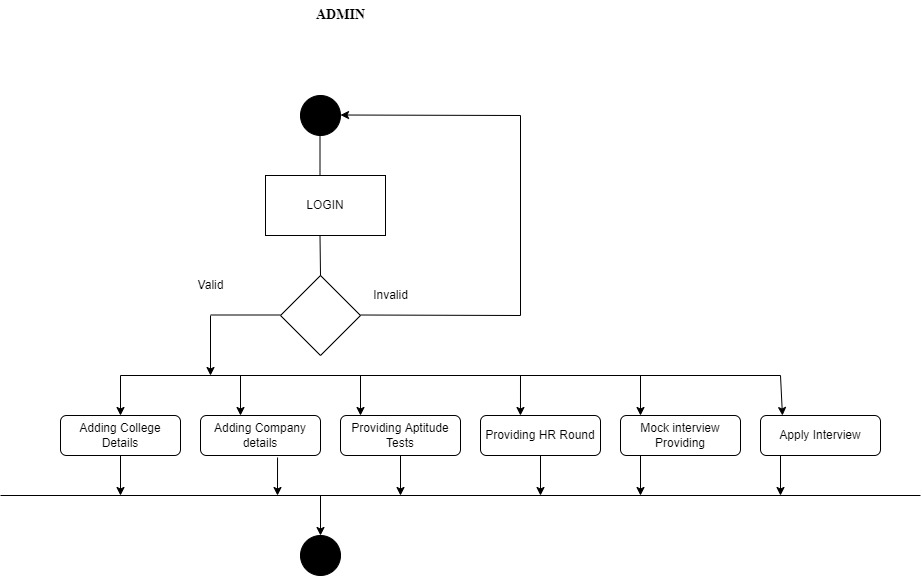
A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. The main purpose of a use case diagram is to show what system functions are performed for which actor. A use case describes a sequence of actions that provide something of measurable value to an actor and is drawn as a horizontal ellipse. An actor is a person, organization, or external system that plays a role in one or more interactions with the system. A rectangle is drawn around the use cases, called the system boundary box, to indicate the scope of the system.

****

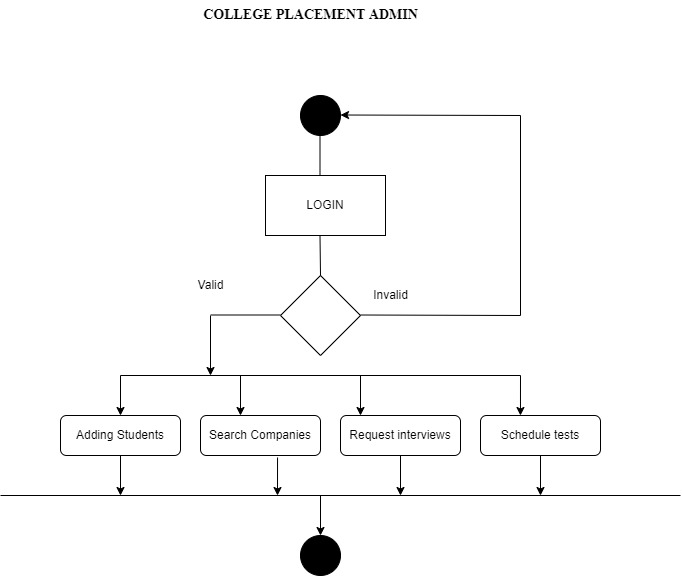
**Fig : 3.6. -USECASE – CF PLACEMENT CELL**

* 1. **ACTIVITY DIAGRAM**

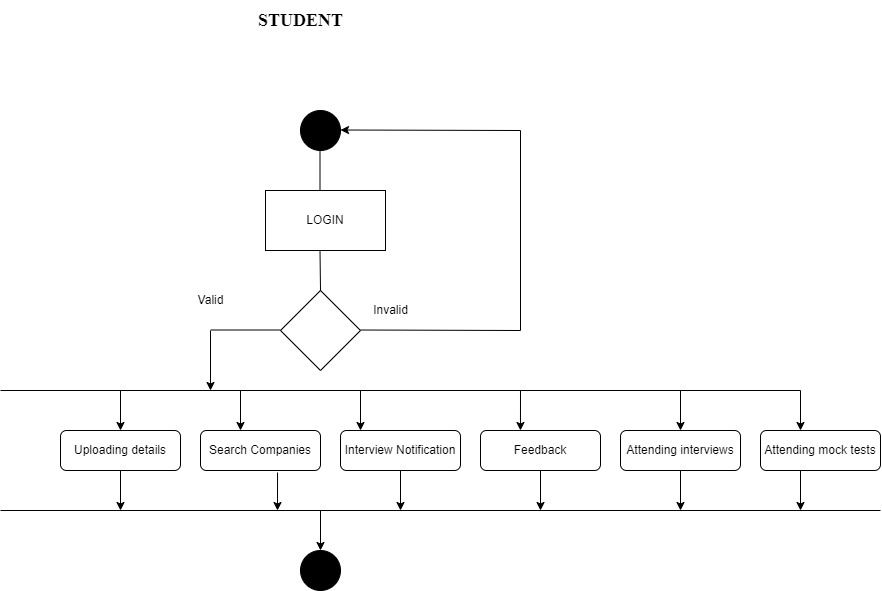
Activity diagram is another important behavioral diagram in uml diagram to describe dynamic aspects of the system. Activity diagram is essentially an advanced version of flow chart that modeling the flow from one activity to another activity.



**Fig : 3.7.1 -ACTIVITY – CF MANANGEMENT**

****

**Fig : 3.7.2 -ACTIVITY – COLLEGE PLACEMENT ADMIN**

****

**Fig : 3.7.3 -ACTIVITY – STUDENT**

* 1. **TABLE DESIGN**

**FEATURES OF DATABASE**

Compactness - where there is no need for the old paper files that has a big size.

Speed - Because of the computer can restore the stored Data Base and upgrading it very fast than the normal human manual hand can do.

Less drudgery - Because the computer do everything for you.

Currency – The more specific you can have when you asking for a Data Base information. Simplicity - An easy way to collect, access-connects, and display information.

Stability - To prevent unnecessary loss of data.

Security – To prevent unauthorized access to private data.

**PRIMARY KEY**

A primary key, also called a primary keyword, is a key in a relational database that is unique for each record. It is a unique identifier, such as a driver license number, telephone number (including area code), or vehicle identification number (VIN). A relational database must always have one and only one primary key.

**FOREIGN KEY**

In the context of relational databases, a foreign key is a field (or collection of fields) in one table that uniquely identifies a row of another table or the same table. In simpler words, the foreign key is defined in a second table, but it refers to the primary key in the first table.

**NORMALIZATION**

Normalization is the process of splitting relations into well structured relations that allow users to insert, delete, and update tuples without introducing database inconsistencies. Without normalization many problems can occur when trying to load an integrated conceptual model into the DBMS. These problems arise from relations that are generated directly from user views are called anomalies. There are three types of anomalies: update, deletion and insertion anomalies.

### Anomalies in DBMS

Anomalies are problems which arises when changes are made to the database. There are three types of anomalies:

INSERTION anomaly -This problem occurs due to inability to represent a certain information

Example :when information about new cannot be inserted in a DB, DELETION anomaly -This occurs due to loss of useful information

Example : when we delete information in a DB we will loss relevant information.

UPDATE anomaly -This type occurs due to redundant information make difficult to update.

To overcome these anomalies we need to normalize the data. In the next section we will discuss about normalization.

### Normalization

Here are the most commonly used normal forms:

* First normal form(1NF)
* Second normal form(2NF)
* Third normal form(3NF)

### First normal form (1NF)

As per the rule of first normal form, an attribute (column) of a table cannot hold multiple values. It should hold only atomic values.

### Second normal form (2NF)

A table is said to be in 2NF if both the following conditions hold:

* Table is in 1NF (First normal form)
* No non-prime attribute is dependent on the proper subset of any candidate key of table. An attribute that is not part of any candidate key is known as non-prime attribute.

### Third Normal form (3NF)

A table design is said to be in 3NF if both the following conditions hold:

* Table must be in 2NF
* [Transitive functional dependency](http://beginnersbook.com/2015/04/transitive-dependency-in-dbms/) of non-prime attribute on any super key should be removed.

**TABLES**

**TABLE DESIGN**

**Login**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **DATATYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| uid | int | Foreign KEY | User id |
| uname | Varchar(50) | Not Null | Username |
| upass | Varchar(50) | Not Null | User password |
| utype | Varchar(50) | Not Null | User type |

**tbl\_clinic**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **DATATYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| Clid | Int | Primary Key | Clinic Id |
| Clname | Varchar(20) | Not Null | Clinic Name |
| Adr | Varchar(20) | Not Null | Address |
| phn | Int | Not Null | Phone No |
| Emaild | Varchar(20) | Not Null | Email |
| Lni | Varchar(20) | Not Null | License Number |

**tbl­­\_breed**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **DATATYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| catid | Int | PRIMARY KEY | breed Id |
| ctname | Varchar(20) | Not Null | breed Name |

**tbl\_petowner**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **DATATYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| cid | Int | PRIMARY KEY | petowner Id |
| cname | Varchar(20) | Not Null | petowner Name |
| chno | Varchar(20) | Not Null | petowner House |
| street | Varchar(20) | Not Null | Street |
| city | Varchar(20) | Not Null | City |
| dist | Varchar(20) | Not Null | District |
| pin | Int | Not Null | Pin Code |
| phno | Varchar(20) | Not Null | Phone Number |
| em | Varchar(20) | Not Null | Email |

**tbl\_Complaints**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **DATATYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| fid | Int | PRIMARY KEY | complaint Id |
| cid | Int | Foreign KEY | Pet owner/client Id |
| cmpl | Varchar(500) | Not Null | complaint |

**tbl\_petprofile**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **DATATYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| Pid | Int(20) | PRIMARY KEY | Pet id |
| pname | Varchar(20) | Not Null | PET NAME |
| catid | Int | Foreign KEY | Breed/ pet type |
| Remakrs | Varchar(2000) | Not Null | Remakrs |
| Age | Int | Not Null | Age in months |
| img | Varchar(100) | Not Null | Image |
| cid | Int | Foreign key | Pet owner id |

**tbl\_appointment**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **DATATYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| Appid | Int(20) | PRIMARY KEY | Appointment id |
| Adate | Date | Not Null | Date |
| Did | Int | Foreign KEY | Doctor id |
| Detals | Varchar(2000) | Not Null | Details |
| Pid | Int | Foreign key | Pet id |
| Status | Varchar(100) | Not Null | Status |
| cid | Int | Foreign Key | Pet Owner id |
| Rmrk | Varchar(100) | Not Null | Remarks |

**tbl\_chatm**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **DATATYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| Chid | Int | PRIMARY KEY | Chat id |
| Cdate | Date | Not Null | Chat date |
| Did | Int | Foreign Key | Doctor id |
| cid | Int | Foreign Key | Pet Owner id |

**tbl\_chatc**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **DATATYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| Ccid | Int | PRIMARY KEY | Chat id |
| Chid | Int | Foreign Key | Chat date |
| Msg | Varchar(100) | Not Null | Message |
| Ctype | Varchar(20) | Not Null | Sender type |

**tbl\_doctors**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **DATATYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| Did | Int | PRIMARY KEY | Doctor id |
| Dname | Varchar(20) | Not Null | Doctor name |
| Address | Varchar(20) | Not Null | Address |
| dist | Varchar(20) | Not Null | District |
| pin | Int | Not Null | Pin Code |
| phno | Varchar(20) | Not Null | Phone Number |
| em | Varchar(20) | Not Null | Email |
| Qual | Varchar(20) | Not Null | Qualifications |
| Exp | Varchar(20) | Not Null | Experience |
| Clid | Int | Foreign Key | Clinic Id |

**CHAPTER IV SYSTEM REQUIREMENTS**

* 1. **SOFTWARE REQUIREMENTS:-**
     + Operating System : Windows 7 or higher versions
     + Front End : PHP
     + Back End : MYSQL

## HARDWARE REQUIREMENTS:-

* + - Processor : Intel Pentium Dual Core/ above
    - Hard Disk Space : 931 GB
    - RAM : 4.00 GB
    - Display : 14.1 “Colour Monitor(LCD,CRT or LED)

## DEVELOPMENT ENVIRONMENT REQUIREMENTS

The following list will give an idea about the hardware and software infrastructures used during the development phase of CAMERINFOLKS PLACEMENT CELL:-

## Hardware Requirements

* + - Processor : Intel Pentium(R)
    - Hard Disk Space : 931 GB
    - RAM : 4.00 GB

## Software Requirements

* Language: PHP
* Operating System : Windows 10
* Web Browser : Mozilla Firefox,Chrome

# CHAPTER V PROJECT IMPLEMENTATION

## INTRODUCTION

Implementation involves placing the complete and tested system software into actual work environment. Implementation is concerned with translating design specification with source code. The primary goal of implementation is to write the source code to its specification can easily be verified, and so that debugging, testing and modification can be eased. The goal can be achieved by making the source code clear and straight forward as possible. Implementation means the process of converting a new or revised system design into operational one. The three types of implementation are

* Implementation of a complete system to replace a manual system
* Implementation of a new system to replace existing one
* Implementation of a modified application to replace an existing one

## IMPLEMENTATION DETAILS

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built-in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective..

### Django

Django is a [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) based [free and open-source](https://en.wikipedia.org/wiki/Free_and_open-source_software) [web framework,](https://en.wikipedia.org/wiki/Web_framework) which follows the model-template- view (MTV) [architectural pattern.](https://en.wikipedia.org/wiki/Architectural_pattern_(computer_science)) Django's primary goal is to ease the creation of complex, database driven websites. The framework emphasizes [reusability](https://en.wikipedia.org/wiki/Reusability) and pluggability of components, less code, low coupling, rapid development, and the principle of [don't repeat yourself.](https://en.wikipedia.org/wiki/Don%27t_repeat_yourself) Python is used throughout, even for settings files and data models. Django also provides an optional administrative [create, read,](https://en.wikipedia.org/wiki/Create%2C_read%2C_update_and_delete) [update and delete](https://en.wikipedia.org/wiki/Create%2C_read%2C_update_and_delete) interface that is generated dynamically through [introspection](https://en.wikipedia.org/wiki/Type_introspection) and configured via admin models.

Despite having its own nomenclature, such as naming the callable objects generating the [HTTP](https://en.wikipedia.org/wiki/HTTP) responses views, the core Django framework can be seen as an [MVC](https://en.wikipedia.org/wiki/Model-view-controller) architecture. It consists of an [object-relational mapper](https://en.wikipedia.org/wiki/Object-relational_mapping) (ORM) that mediates between [data models](https://en.wikipedia.org/wiki/Data_modeling) (defined as Python classes) and a [relational database](https://en.wikipedia.org/wiki/Relational_database) (Model), a system for processing HTTP requests with a [web templating system](https://en.wikipedia.org/wiki/Web_template_system) (View), and a [regular-expression](https://en.wikipedia.org/wiki/Regular_expression)-based [URL](https://en.wikipedia.org/wiki/Uniform_Resource_Locator) dispatcher (Controll

Also included in the core framework are:

* A lightweight and standalone web server for development and testing.
* A form serialization and validation system that can translate between [HTML](https://en.wikipedia.org/wiki/HTML) forms and values suitable for storage in the database.
* A template system that utilizes the concept of [inheritance](https://en.wikipedia.org/wiki/Inheritance_(object-oriented_programming)) borrowed from object- oriented programming.
* A [caching](https://en.wikipedia.org/wiki/Web_cache) framework that can use any of several cache methods
* Support for [middleware](https://en.wikipedia.org/wiki/Middleware) classes that can intervene at various stages of request processing and carry out custom functions
* An internal dispatcher system that allows components of an application to communicate events to each other via pre-defined signal.
* An [internationalization](https://en.wikipedia.org/wiki/Internationalization_and_localization) system, including translations of Django's own components into a variety of languages
* A system for extending the capabilities of the template engine
* An interface to Python's built-[in unit test](https://en.wikipedia.org/wiki/Unit_test) framework
* Django REST framework is a powerful and flexible toolkit for building Web APIs

### Hypertext Mark-up Language (HTML)

It is the standard [mark-up language](https://en.wikipedia.org/wiki/Markup_language) for documents designed to be displayed in a [web browser.](https://en.wikipedia.org/wiki/Web_browser) It can be assisted by technologies such as [Cascading Style Sheets](https://en.wikipedia.org/wiki/Cascading_Style_Sheets) (CSS) and [scripting languages](https://en.wikipedia.org/wiki/Scripting_language) such as [JavaScript.](https://en.wikipedia.org/wiki/JavaScript)

[Web browsers](https://en.wikipedia.org/wiki/Web_browser) receive HTML documents from a [web server](https://en.wikipedia.org/wiki/Web_server) or from local storage and [render](https://en.wikipedia.org/wiki/Browser_engine) the documents into multimedia web pages. HTML describes the structure of a web page [semantically](https://en.wikipedia.org/wiki/Semantic_Web) and originally included cues for the appearance of the document.

[HTML elements](https://en.wikipedia.org/wiki/HTML_element) are the building blocks of HTML pages. With HTML constructs, [images](https://en.wikipedia.org/wiki/HTML_element#Images_and_objects) and other objects such as [interactive forms](https://en.wikipedia.org/wiki/Fieldset) may be embedded into the rendered page. HTML provides a means to create [structured documents](https://en.wikipedia.org/wiki/Structured_document) by denoting structural [semantics](https://en.wikipedia.org/wiki/Semantics) for text such as headings, paragraphs, lists, [links](https://en.wikipedia.org/wiki/Hyperlink), quotes and other items. HTML elements are delineated by *tags*, written using [angle brackets.](https://en.wikipedia.org/wiki/Bracket#Angle_brackets)

Tags such as <imp /> and <input /> directly introduce content into the page. Other tags such as <p> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags but use them to interpret the content of the page.

HTML can embed programs written in a [scripting language](https://en.wikipedia.org/wiki/Scripting_language) such as [JavaScript](https://en.wikipedia.org/wiki/JavaScript), which affects the behavior and content of web pages. Inclusion of CSS defines the look and layout of content.

### JavaScript

It is a lightweight, interpreted, object-oriented language with [first-class](https://en.wikipedia.org/wiki/First-class_function) [functions](https://en.wikipedia.org/wiki/First-class_function), and is best known as the scripting language for Web pages, but it's [used in many non-browser environments](https://en.wikipedia.org/wiki/JavaScript#Uses_outside_Web_pages) as well. It is a [prototype-based,](https://en.wikipedia.org/wiki/Prototype-based_programming) multi- paradigm scripting language that is dynamic, and supports object- oriented, imperative, and functional programming styles.

## ABOUT DATABASE

A database management system (DBMS) is a computer software application that interacts with the user, other applications, and the database itself to capture and analyze data. A general- purpose DBMS is designed to allow the definition, creation, querying, update, and administration of databases.

### Datatypes in our Database Int

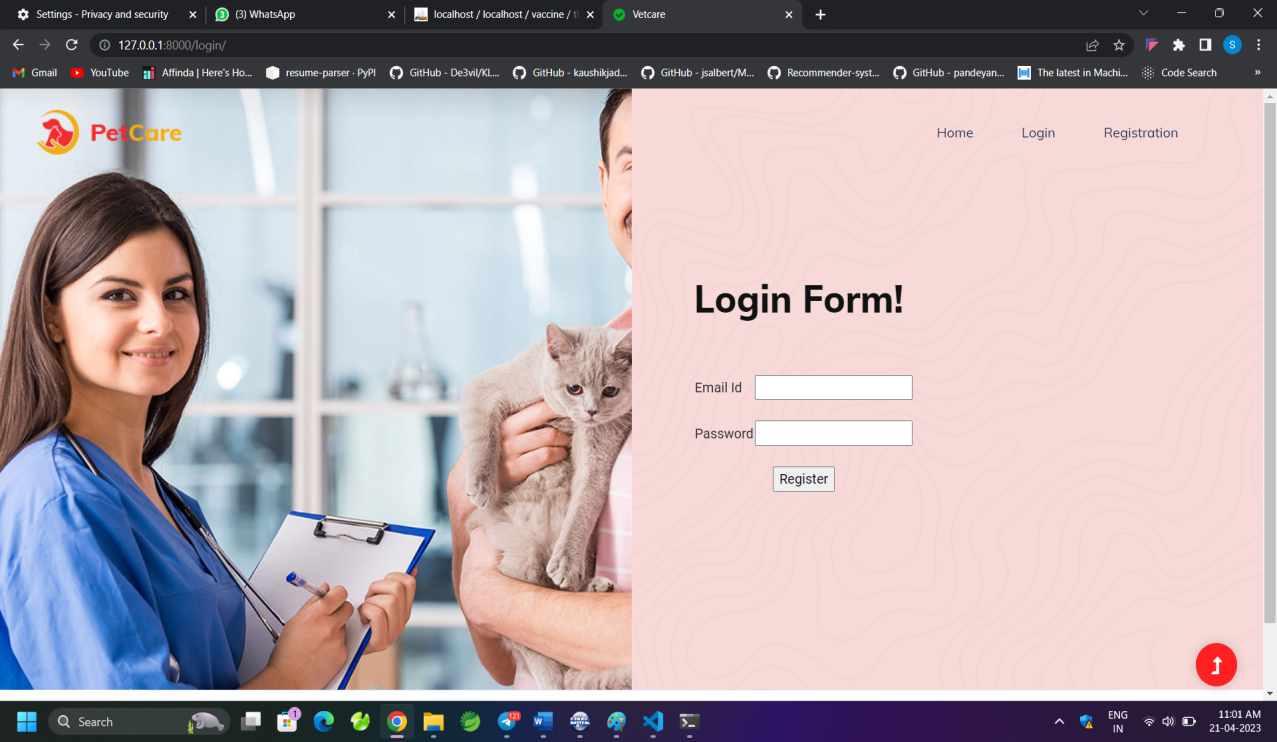
The int data type is the primary integer data type in SQL Server. The bigint data type is intended for use when integer values might exceed the range that is supported by the int data type. SQL Server does not automatically promote other integer data types (tiny int, small int, and int) to big int.

**Varchar**

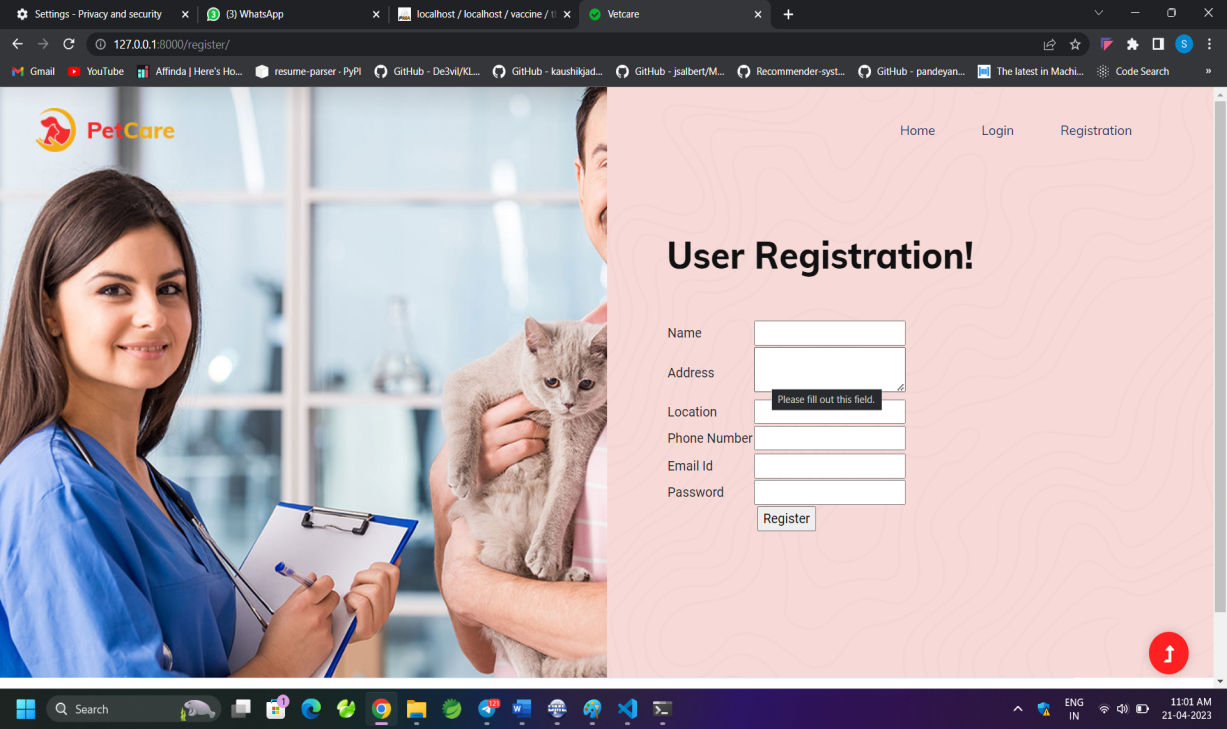
In SQL Server, each column, local variable, expression, and parameter has a related data type. The collation of the result is determined by the rules of collation precedence when the result data type is char, varchar, text, nchar, nvarchar, or ntext.

## SCREENSHOTS

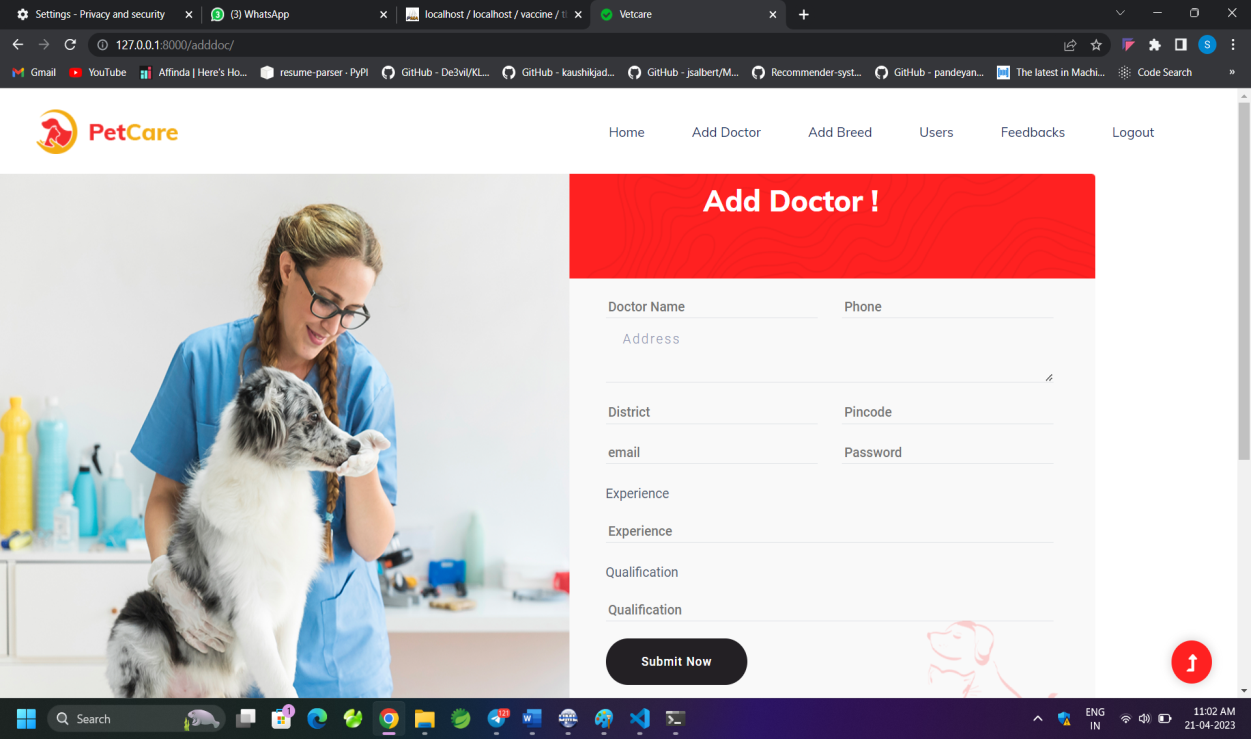
**Fig 5.2.1 Login**



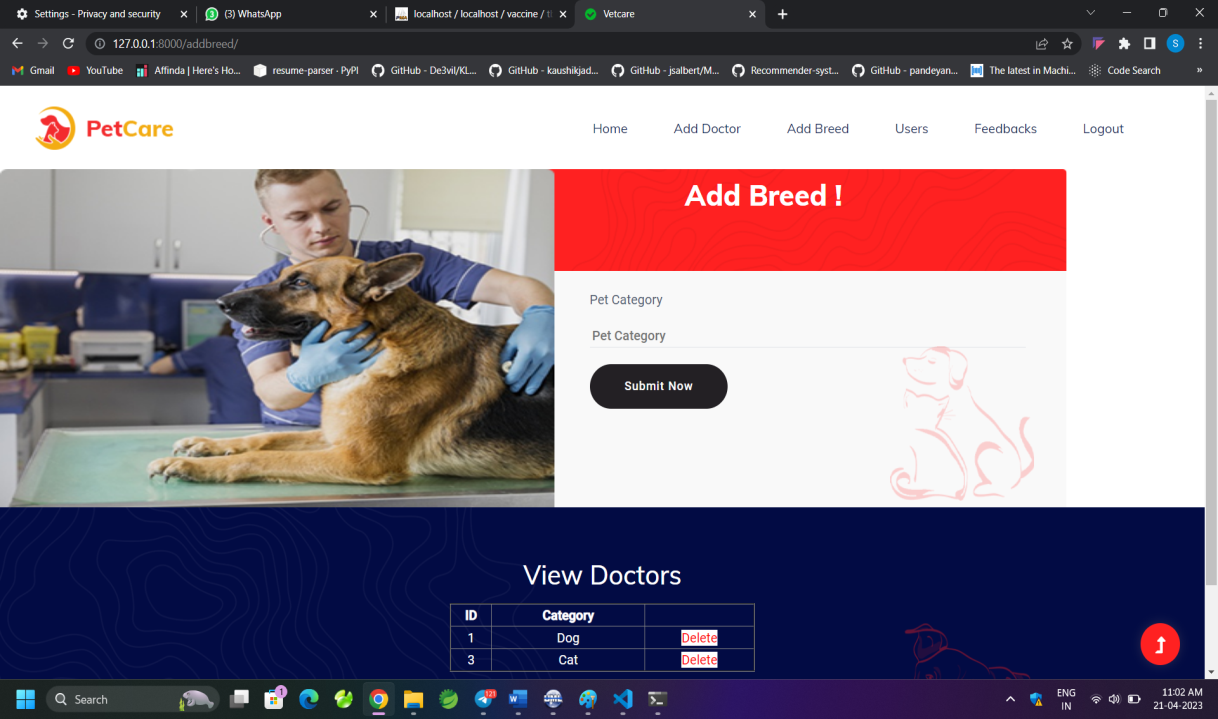
**Fig 5.2.2 User Registration**



**Fig 5.2.3 Add Doctors**



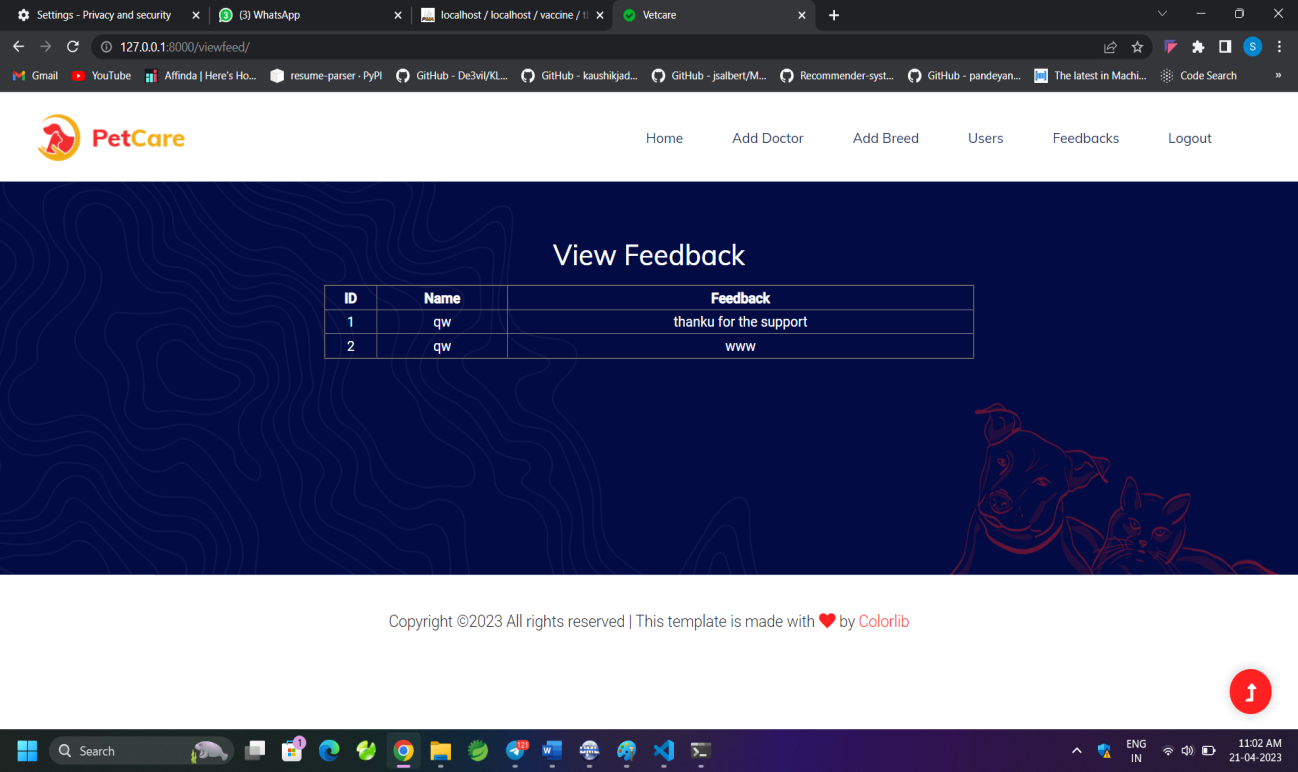
**Fig 5.2.4 Add Breed**



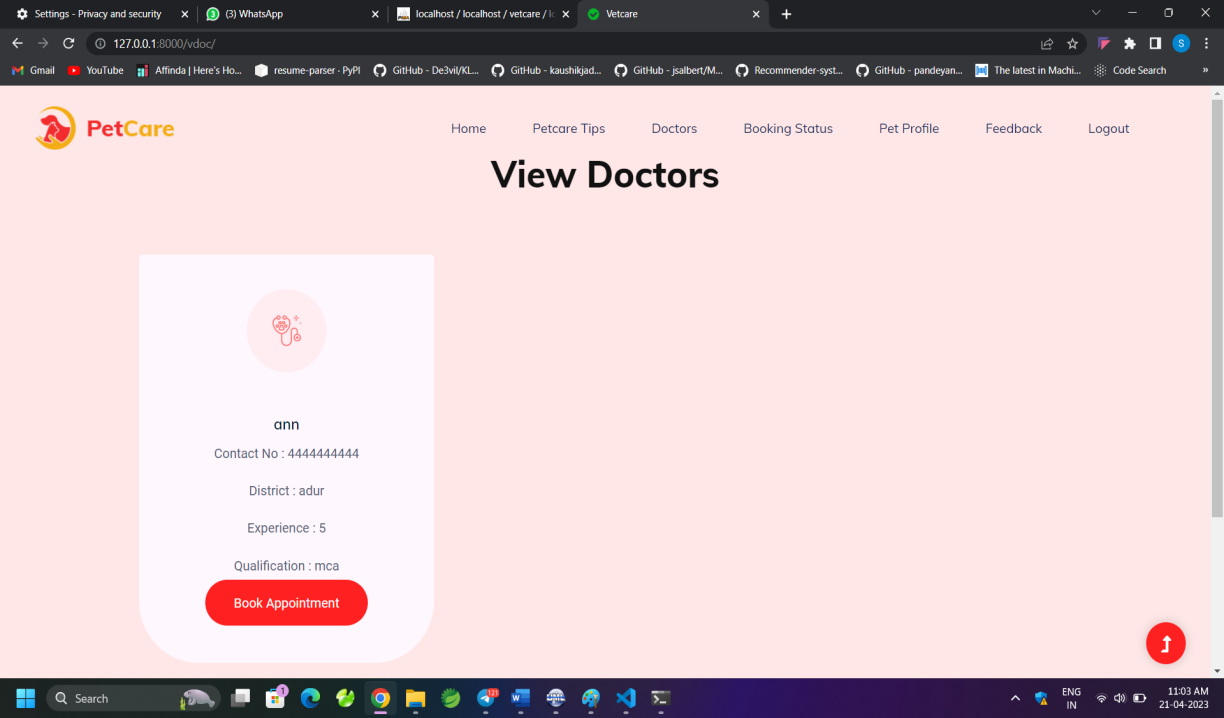
**Fig 5.2.5 View Users**



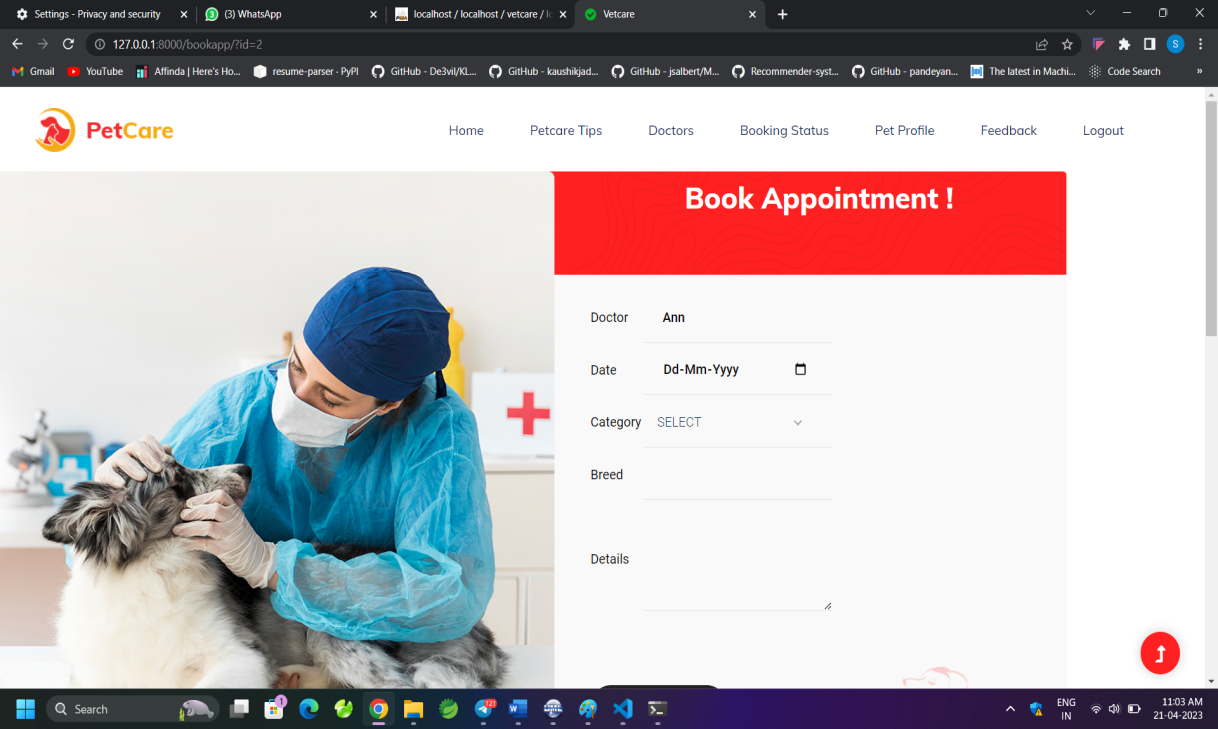
**Fig 5.2.6 View Feedback**



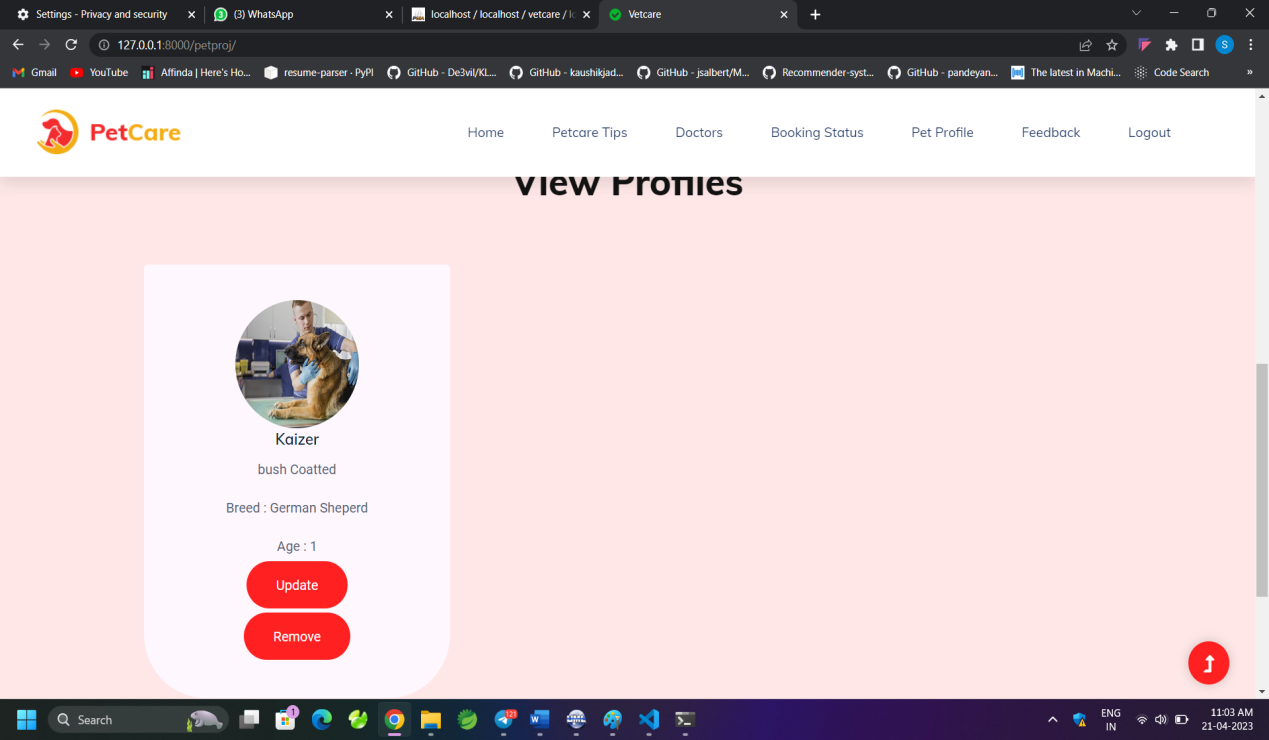
**Fig 5.2.7 View Doctors**



**Fig 5.2.8 Appointment Booking**



**Fig 5.2.9 View pet profile**



# CHAPTER VI

# TESTING

## TESTING METHODOLOGIES AND STRATEGIES

The Software Test Plan (STS) describes plans for qualification testing of software “Web in my hand”. Generally, s/m testing involves testing integration of each module in the s/m. The objective while testing the s/m is to test the discrepancies between the s/m and the original objective. The quality of an information s/m depends on its design, development, implementation and design. Testing is the most important activity in the development phase. Testing is the process of finding errors or bugs in the s/m. Testing ensure that the user needs are satisfied. In other words it is a process by which one detects the defects in the s/m.

The primary goal of test plan is to define testing procedures that will ensure that the s/w is functionally correct from a document perspective and will verify application scalability limits. It also proves that reliability and fail over aspects of the system can indeed survive instances of system failure. It describes and identifies the tests to be performed, and provide schedules for test activities.

## UNIT TESTING

The modules of the project are tested separately. Unit testing focuses verification efforts even in the smallest unit of software design in each module. This is known as “Module Testing”. This testing is carried out in the programming style itself. In this testing each module is focused to work satisfactorily as regard to expected output from the module. There are some validation checks for the fields. Unit testing gives stress on the modules of application independently of one another, to find errors. Unit testing is done for following modules:

* Pet Owner Portal
* Clinics Portal
* Veterinary Doctor Portal
* Admin Portal

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Name** | **Precondition** | **Reference module/name** | **Expected output** | **Result obtained** |
| TC01 | Enter valid  username and password | Login | Login to user profile | Expected output |
| TC02 | Enter invalid username and password | Login | Displays message please enter valid Email id /password | Expected output |
| TC03 | user enter valid data while registration | Login | Display message registered successfully | Expected output |
| TC04 | user keeps one row as blank | Login | Display message please fill out this field | Expected output |
| TC05 | user provide invalid username while login | Login | Displays message please enter valid email id /password | Expected output |
| TC06 | User enter valid  date while registration | Signup | Display message  registered successfully | Expected output |
| TC07 | User keeps one row as blank | Signup | Display message  please fill out this field | Expected output |
| TC08 | User provide invalid username while  Login | Signup | Displays message please enter valid email id /password | Expected output |

### Table no : 6.2.1-Unit test report

* 1. **INTEGRATION TESTING**

In this, whole modules of the project “Camerinfolks placement cell” are connected and tested. Data can be lost across an interface, one module can have an adverse effect on the other sub-functions, when combined may not produce the desired functions. Integrated testing is the systematic testing to uncover the errors within the interface. This testing is done with simple data and the developed system has run successfully with this simple data. The need for integrated system is to find the overall system performance.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Name** | **Description** | **Expected Result** | **Actual Result** | **Status** |
| TC01 | Verify that the student is able to navigate through different modules | When student login to the page select companies and must navigates to the details of that page | System navigates to the companies and produces the details | Pass |
| TC02 | Verify that all the user request are redirecting to correct page | When user  click on the  product ,the  system must redirects to its appropriate page | System redirects to its appropriate page | Passs |
| TC03 | Enter login  credentials and click on the sign in button | System must direct to user home | System is redirected to user home | Pass |

### Table no: 6.3.1-Integration testing

* 1. **SYSTEM TESTING**

The implementation of a computer based system requires that test data to be prepared and that the system and its elements be tested in a planned structured manner. The computer program component is a major sub-system of the computer-based information system and particular attention should be given to the testing of this system element as it is developed in a software development project, errors can be injected at any stage during development. Each will discuss different techniques for detecting and eliminating errors that originate in that phase. In software the use of testing is not limited to the testing phase. Here I have tested all the modules in my project separately and run successfully.

## PERFORMANCE TEST REPORTS

In functional testing the structure of the program is not considered. Test cases are decided solely on the basis of the requirements or specifications of the program or module and the internals of the module or the programs are not considered for selection of test cases. Due to its nature, functional testing is also called black box testing. The basis for deciding the test cases in functional testing is the requirements or the specifications of the system or module. In the structural testing, test cases are generated based on the actual code of the program or module to be tested. This structural testing is sometimes called glass-box testing. The intent of the structural testing is not to exercise all the different input or output conditions, but to exercise the different programming structures and data structures used in the program.

# CHAPTER VII CONCLUSION AND FUTURE ENHANCEMENTS

## CONCLUSION

## The application aims on digitizing the process of taking attendance across colleges, thus eliminating the manual process of taking attendance. This provides the university with an efficient automated alternative to expensive attendance management systems being developed and implemented. This gives a much more comfortable and user friendly experience to the student as well as the faculty to conserve their time and energy thus resulting in a better experience. The value that will add to the users is the overall structure of time value to the normal functioning of lectures and also be identified as adopters of digital technologies. The future scope of the project is enormous as to collaborating it into a learning management system and developing a complete experience for the university.

## FUTURE ENHANCEMENT

We have tried our best to present the information effectively, yet there can be further enhancement in the application. We have taken care of all the critical aspects, which were needed to be taken care of. Because of fast changes in the world of programming this system will gradually get outdated and less effective. For the time being it’s possible to overcome problems by amendments and minor modifications to acknowledgement the need of fundamental design.

Though the new system provides base for improving the efficiency of operations, there are a lot of future enhancements that can be added to this project. Keeping this in view, a provision has been made in the system to facilitate easy modification updating in the future. Any modification will not affect the normal working of the system.

The development system is very interactive, coded in such a way to ensure maximum user friendliness and also allows flexibility for future.

The following are the possible future enhancements:

* Introduce other facilites
* Enable more customized features and bookings.
* Mobile Application where users can easily get notifications

## BIBLIOGRAPHY

1. Elias. M. Awad, “System*Analysis and* Design”, Galgoeia publications – Second Edition2005.
2. Henry Korth,” Data*Base Design Concept*”, Mc grew hill – Fifth Edition 2001. Roger.S.Pressman,” Software*Engineering* “,Mc grew hill  Fifth Edition 2006.
3. [www.stackoverflow.com](http://www.stackoverflow.com)
4. [www.w3schools.com](http://www.w3schools.com)
5. [www.tutorialspoint.com](http://www.tutorialspoint.com)