

# **VETERINARY AND PETS MANAGEMENT SYSTEM**

*Project Report Submitted By*

**ADARSH S**

**Reg. No:AJC20MCA-2003**

*In Partial fulfillment for the Award of the Degree Of*

**MASTER OF COMPUTER APPLICATIONS (2 Year)  
(MCA)**

**APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY**



**AMAL JYOTHI COLLEGE OF ENGINEERING  
KANJIRAPPALLY**

[Affiliated to APJ Abdul Kalam Technological University, Kerala. Approved by AICTE, Accredited by NAAC with 'A' grade. Koovappally, Kanjirappally, Kottayam, Kerala – 686518]

**2020-2022**

**DEPARTMENT OF COMPUTER APPLICATIONS**  
**AMAL JYOTHI COLLEGE OF ENGINEERING**  
**KANJIRAPPALLY**



**CERTIFICATE**

This is to certify that the Project report, “**VETERINARY AND PETS MANAGEMENT SYSTEM**” is the bonafide work of **ADARSH S (Reg.No:AJC20MCA-2003)** in partial fulfillment of the requirements for the award of the Degree of Master of Computer Applications under APJ Abdul Kalam Technological University during the year 2020-2022.

**Ms. Merin Chacko**  
**Internal Guide**

**Rev. Fr. Dr. Rubin Thottupurathu Jose**  
**Coordinator**

**Rev. Fr. Dr. Rubin Thottupurathu Jose**  
**Head of the Department**

**External Examiner**

**(Add CERTIFICATE ON PLAGIARISM CHECK here)**

:

## **DECLARATION**

I hereby declare that the project report “**Veterinary and Pets Management System**” is a bonafided work done at Amal Jyothi College of Engineering, towards the partial fulfilment of the requirements for the award of the Degree of Master of Computer Applications (MCA) from APJ Abdul Kalam Technological University, during the academic year 2020-2022.

**Date: 21-7-2022**

**KANJIRAPPALLY**

**ADARSH S**

**Reg. No: AJC20MCA-2003**

## ACKNOWLEDGEMENT

First and foremost, I thank God almighty for his eternal love and protection throughout the project. I take this opportunity to express my gratitude to all who helped me in completing this project successfully. It has been said that gratitude is the memory of the heart. I wish to express my sincere gratitude to our manager **Rev. Fr. Dr. Mathew Paikatt** and Principal **Dr. Lillykutty Jacob** for providing good faculty for guidance.

I owe a great depth of gratitude towards our Head of the Department **Rev. Fr. Dr. Rubin Thottupurathu Jose** for helping us. I extend my whole hearted thanks to the project coordinators **Rev. Fr. Dr. Rubin Thottupurathu Jose** and **Ms. Nimmy Francis** for their valuable suggestions and for overwhelming concern and guidance from the beginning to the end of the project. I would also like to express sincere gratitude to my guide **Ms. Merin Chacko** for his inspiration and helping hand.

I thank our beloved teachers for their cooperation and suggestions that helped me throughout the project. I express my thanks to all my friends and classmates for their interest, dedication, and encouragement shown towards the project. I convey my hearty thanks to my family for the moral support, suggestions, and encouragement to make this venture a success.

ADARSH S

## **ABSTRACT**

**VETERINARY AND PETS MANAGEMENT SYSTEM** was designed to perform systematic manner through its function requirement. For Veterinary service the user must register to book an appointment for taking vaccination for pet or to adopting a pet. The admin can add the doctors and pets. The user can view the doctor and they can book appointments of suitable doctor. The user can view the pets and they can adopt pets. The doctor can view the appointment status and he has right for approving or rejecting the appointment. The admin can view the orders and he has right for approving or rejecting orders

# CONTENT

<b>Sl. No</b>	<b>Topic</b>	<b>Page No</b>
<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
<b>1.1</b>	<b>PROJECT OVERVIEW</b>	<b>2</b>
<b>1.2</b>	<b>PROJECT SPECIFICATION</b>	<b>2</b>
<b>2</b>	<b>SYSTEM STUDY</b>	<b>4</b>
<b>2.1</b>	<b>INTRODUCTION</b>	<b>5</b>
<b>2.2</b>	<b>EXISTING SYSTEM</b>	<b>6</b>
<b>2.3</b>	<b>DRAWBACKS OF EXISTING SYSTEM</b>	<b>6</b>
<b>2.4</b>	<b>PROPOSED SYSTEM</b>	<b>6</b>
<b>2.5</b>	<b>ADVANTAGES OF PROPOSED SYSTEM</b>	<b>7</b>
<b>3</b>	<b>REQUIREMENT ANALYSIS</b>	<b>9</b>
<b>3.1</b>	<b>FEASIBILITY STUDY</b>	<b>10</b>
<b>3.1.1</b>	<b>ECONOMICAL FEASIBILITY</b>	<b>10</b>
<b>3.1.2</b>	<b>TECHNICAL FEASIBILITY</b>	<b>11</b>
<b>3.1.3</b>	<b>BEHAVIORAL FEASIBILITY</b>	<b>11</b>
<b>3.2</b>	<b>SYSTEM SPECIFICATION</b>	<b>12</b>
<b>3.2.1</b>	<b>HARDWARE SPECIFICATION</b>	<b>12</b>
<b>3.2.2</b>	<b>SOFTWARE SPECIFICATION</b>	<b>12</b>
<b>3.3</b>	<b>SOFTWARE DESCRIPTION</b>	<b>12</b>
<b>3.3.1</b>	<b>PHP</b>	<b>12</b>
<b>3.3.2</b>	<b>MYSQL</b>	<b>13</b>
<b>4</b>	<b>SYSTEM DESIGN</b>	<b>15</b>
<b>4.1</b>	<b>INTRODUCTION</b>	<b>16</b>
<b>4.2</b>	<b>UML DIAGRAM</b>	<b>16</b>
<b>4.2.1</b>	<b>USE CASE DIAGRAM</b>	<b>17</b>
<b>4.2.2</b>	<b>SEQUENCE DIAGRAM</b>	<b>20</b>
<b>4.2.3</b>	<b>STATE CHART DIAGRAM</b>	<b>23</b>
<b>4.2.4</b>	<b>ACTIVITY DIAGRAM</b>	<b>24</b>
<b>4.2.5</b>	<b>CLASS DIAGRAM</b>	<b>28</b>
<b>4.2.6</b>	<b>OBJECT DIAGRAM</b>	<b>29</b>
<b>4.2.7</b>	<b>COMPONENT DIAGRAM</b>	<b>30</b>
<b>4.2.8</b>	<b>DEPLOYMENT DIAGRAM</b>	<b>31</b>

<b>4.3</b>	<b>USER INTERFACE DESIGN</b>	<b>32</b>
<b>4.4</b>	<b>DATA BASE DESIGN</b>	<b>36</b>
<b>5</b>	<b>SYSTEM TESTING</b>	<b>45</b>
<b>5.1</b>	<b>INTRODUCTION</b>	<b>46</b>
<b>5.2</b>	<b>TEST PLAN</b>	<b>47</b>
<b>5.2.1</b>	<b>UNIT TESTING</b>	<b>47</b>
<b>5.2.2</b>	<b>INTEGRATION TESTING</b>	<b>48</b>
<b>5.2.3</b>	<b>VALIDATION TESTING</b>	<b>48</b>
<b>5.2.4</b>	<b>USER ACCEPTANCE TESTING</b>	<b>49</b>
<b>6</b>	<b>IMPLEMENTATION</b>	<b>52</b>
<b>6.1</b>	<b>INTRODUCTION</b>	<b>53</b>
<b>6.2</b>	<b>IMPLEMENTATION PROCEDURE</b>	<b>53</b>
<b>6.2.1</b>	<b>USER TRAINING</b>	<b>54</b>
<b>6.2.2</b>	<b>TRAINING ON APPLICATION SOFTWARE</b>	<b>54</b>
<b>6.2.3</b>	<b>SYSTEM MAINTENANCE</b>	<b>54</b>
<b>6.2.4</b>	<b>HOSTING</b>	<b>55</b>
<b>7</b>	<b>CONCLUSION &amp; FUTURE SCOPE</b>	<b>56</b>
<b>7.1</b>	<b>CONCLUSION</b>	<b>57</b>
<b>7.2</b>	<b>FUTURE SCOPE</b>	<b>57</b>
<b>8</b>	<b>BIBLIOGRAPHY</b>	<b>58</b>
<b>9</b>	<b>APPENDIX</b>	<b>59</b>
<b>9.1</b>	<b>SAMPLE CODE</b>	<b>60</b>
<b>9.2</b>	<b>SCREEN SHOTS</b>	<b>75</b>
<b>9.3</b>	<b>PLAGIARISM REPORT</b>	<b>81</b>



## **List of Abbreviation**

IDE	-	Integrated Development Environment
PHP	-	Hyper Text Markup Language.
CSS	-	Cascading Style Sheet
SQL	-	Structured Query Language
UML	-	Unified Modeling Language

## **CHAPTER 1**

### **INTRODUCTION**

## 1.1 PROJECT OVERVIEW

“VETERINARY AND PETS MANAGEMENT SYSTEM” was designed to perform systematic manner through its function requirement. For Veterinary service the user must register to book an appointment for taking vaccination for pet or to adopting a pet. The admin can add the doctors and pets. The user can view the doctor and they can book appointments of suitable doctor. The user can view the pets and they can adopt pets. The doctor can view the appointment status and he has right for approving or rejecting the appointment. The admin can view the orders and he has right for approving or rejecting orders

## 1.2 PROJECT SPECIFICATION

The proposed solution attempts to make it simple and straightforward for users to book appointment for veterinary doctor. The system is made up of Three modules. They are:

### 1. Admin Module

Admin have rights to add doctor and pets view doctor details.

Admin can view the orders.

### 2. User Module

User must register to book an appointment of doctor or to adopt any pet. User can view the doctor and book appointment directly. The user can view the appointment status from doctor (approved/declined). The user can view the order status from admin (approved/declined).

### 3. Doctor Module

The doctor can view all his appointments. The doctor can approve or reject the appointments.

.

## **CHAPTER 2**

### **SYSTEM STUDY**

## 2.1 INTRODUCTION

System analysis is a process of collecting and interpreting information to identify and diagnose system problems and suggest improvements. It is an activity with intense communication between system users and system developers. System analysis is an important part of any system development process. The system is examined and analyzed in great detail. The system analyst is responsible for questioning the current system and understanding how it works. The system is analyzed as a whole and the system entry is identified. The production of your organization is traced to various processes. Systems analysis involves the recognition of problems, the identification of relevant decision variables, the analysis and synthesis of various factors, and the determination of optimal or at least satisfactory solutions or courses of action. A detailed study of the process should be conducted using various techniques such as interviews, questionnaires, etc. The data gathered from these sources should be carefully evaluated in order to reach a conclusion. The main thing is to understand how the system works. It is the existing system. The existing system is reviewed in detail and problems identified. The designer now works as a problem solver, trying to solve business difficulties. The solutions are given for information only. The proposal will be evaluated against the current system and the best option will be selected. The proposal is proposed to the user for approval. At the request of the user, the offer will be revised and, if necessary, appropriate changes will be made. Pre-investigation is the process of gathering and interpreting facts which can then be used for further investigation of the system. Pre-study is a troubleshooting activity that requires intensive communication between system users and system developers. The company conducts feasibility studies to determine how much work is required to complete a project. With this information, the company can decide how to best implement the project. With this information, the company can decide how best to carry out the project.

## 2.2 EXISTING SYSTEM

Systems used offline are not entirely automated. The offline approach requires the user to physically visit the hospital to book appointment, and complete the payment before leaving. In proposed system patient can book appointment through online. The shortcomings of the current system are remedied by the suggested system. They are effective in terms of user friendliness and order tracking. Using the new system user can viewing the profile and view the appointment details

## 2.3 DRAWBACKS OF EXISTING SYSTEM

- Less convenient in managing details.
- Human effort is needed.

## 2.4 PROPOSED SYSTEM

Nowadays petting has become a new trend. Lots of individuals discover trouble with how to deal with their pets when they leave town for a couple of days. User can adopt pets whenever they wish to. Veterinary and pets management system is available for that purpose. This Project of Veterinary and Pets Management System is a web application.

## 2.5 ADVANTAGES OF PROPOSED SYSTEM

The system design is very simple and can be easily implemented. The system requires very little system resources and works in almost all configurations. The item has several qualities that make it desirable.

- **Performance of the system: -**

Response time is very good for given piece of work. The system will support multi-user environment.

- **Ensure data accuracy: -**

The Proposed system eliminates human error when entering user data during registration

- **Better service: -**

The system reduces the amount of hardcopy storage required. We can save time and energy by doing the same. Data can be stored for a longer period of time without data loss

## **CHAPTER 3**

### **REQUIREMENT ANALYSIS**



### 3.1 FEASIBILITY STUDY

A feasibility study is carried out to determine whether, once completed, the project will serve the organization's purpose for the amount of work, time, and effort put in it. The feasibility examination enables the developer to forecast the project's future and benefits. A feasibility evaluation of a device concept is entirely focused on its practicality, that is, the impact on the organisation, the ability to satisfy the desires of its users, and the effective use of resources. As a result, when a brand new software is presented, it frequently goes through a proof of concept before being authorised for development.

The report explains the feasibility of the proposed project and lists various areas that have been carefully considered at some time in the feasibility examination of this project, such as: B. the technological, financial, and operational feasibility. Following are its characteristics:-

#### 3.1.1 Economical Feasibility

The growing machine must be justified in terms of cost and benefit. Criteria to ensure that the effort is focused on the project, with the purpose of providing the best, go back to the beginning. The cost of a new machine is one of the aspects that influence its development.

The following are some of the important financial questions asked during preliminary investigation:

- The costs conduct a full system investigation.
- The cost of the hardware and software.
- The benefits in the form of reduced costs or fewer costly errors.

The suggested system is being developed as part of the project work, so there are no manual costs associated with it. Furthermore, all resources are currently available, indicating that the system may be developed affordably.

The costs of the e-wallet project were separated according to the system used, development expenditures, and hosting charges. The project was developed at a modest cost after all of the calculations were completed. Because it was created entirely utilising open source software.

### 3.1.2 Technical Feasibility

The system must first be evaluated from a technical point of view. The assessment of this feasibility must be based on a blueprint of the system requirement in terms of inputs, outputs, programs and procedures. After a scheme of the system has been identified, research must continue to propose the type of equipment and method required to develop the system to operate the system once it has been designed.

Technical issues raised during the investigation are:

- Does the existing technology sufficient for the suggested one?
- Can the system expand if developed?

The project should be planned so that the key features and performance may be achieved while staying within the constraints. A high resolution scanning device and cryptographic techniques are required for the project. Since newer versions of the same programme support older ones, the system may continue be used even though the technology may become outdated over time. Thus, the project has little restrictions. The system was created using PHP for the front end and MySQL for the back end; it is theoretically feasible to develop the project. The system was built using PHP for the front end and MySQL for the back end; it is theoretically feasible to develop the project. The machine used has a high-performance Intel i3 core processor, 4GB of RAM, and a 1TB hard drive

### 3.1.3 Behavioral Feasibility

The proposed system includes the following questions:

- Is there enough assistance for the users?
- Will the proposed system harm any one?

When created and implemented, the project would be advantageous since it achieves the goals. It is determined that the project is behaviorally feasible after thoroughly examining all behavioural factors.

Users may easily utilize VETERINARY AND PETS MANAGEMENT SYSTEM because of its straightforward GUI. There is no need for training because VETERINARY AND PETS MANAGEMENT SYSTEM is so straightforward.

## 3.2 SYSTEM SPECIFICATION

### 3.2.1 Hardware Specification

Processor	- Intel core i5
RAM	- 2 GB
Hard disk	- 1 TB

### 3.2.2 Software Specification

Front End	- PHP
Backend	- MYSQL
Client on PC	- Windows 7 and above.
Technologies used	- JS, HTML5,CSS

## 3.3 SOFTWARE DESCRIPTION

### 3.3.1 PHP

PHP is a general-purpose computer language with server-side scripting capabilities that is used for web development. Today, 2.1 million web servers and over 244 million websites use PHP. Rasmus Ledorf formed the PHP group in 1995, which at the moment creates the standard PHP implementation. PHP is now shortened to PHP: Hypertext Predecessor, despite the fact that it was originally an acronym for personal home page. The final web page is produced after PHP code has been processed by a web server using a PHP processor module. PHP commands may be incorporated directly into an HTML source document to execute data rather than contacting an external file. It has also developed to include a command-line interface capability and the ability to be used standalone, which is incompatible with the GNU General Public License (GPL) due to limitations on the usage of the term PHP. On almost all web servers and as a standalone shell on almost all platforms and operating systems, PHP is free to install.

### 3.3.2 MySQL

Oracle Corporation develops, distributes, and supports MySQL, the most popular Open Source SQL database management system. The MySQL Web site contains up-to-date information regarding MySQL software.

- **MySQL is a database management system.**

A database is a well-organized collection of data. It could be anything from a basic grocery list to a photo gallery or the massive volumes of data in a business network. A database management system, such as MySQL Server, is required to add, access, and process data contained in a computer database. Database management systems play an important role in computing, either as standalone utilities or as components of other programs, because computers are exceptionally adept at processing enormous volumes of data.

- **MySQL databases are relational.**

A relational database stores data in distinct tables rather than in a single large storehouse. The database structures are grouped into physical files that are optimised for performance. The logical model, which includes objects like databases, tables, views, rows, and columns, provides a versatile programming environment. You define the relationships between distinct data fields, such as one-to-one, one-to-many, unique, required, or optional, as well as "points" between different tables. The database enforces these constraints, ensuring that your application never encounters inconsistent, duplicate, orphan, out-of-date, or missing data. SQL is an abbreviation for "Structured Query Language." SQL is the most widely used standardised language for accessing databases. Depending on your programming environment, you may immediately enter SQL (for example, to generate reports), embed SQL statements in other languages' programs, or employ a language-specific API that hides the SQL syntax. The ANSI/ISO SQL Standard defines SQL. Since 1986, the SQL standard has been changing, and various variants exist. In this document, "SQL92" refers to the 1992 standard, "SQL: 1999" refers to the 1999 standard, and "SQL: 2003" refers to the current version of the standard. We refer to "the SQL standard" as the current version of the SQL Standard at any given time.

- **MySQL software is Open Source.**

Open Source means that anybody may use and change the application. Anyone may use MySQL software for free by downloading it from the Internet. You are free to inspect the source code and alter it to suit your needs. The MySQL software uses the GPL (GNU General Public License) to define what you may and cannot do with the program in certain contexts. If you are dissatisfied with the GPL or need to incorporate MySQL code into a business product, you may purchase a commercially licenced version from us. More information may be found in the MySQL Licensing Overview.

- **The MySQL Database Server is very fast, reliable, scalable, and easy to use.**

Give it a shot if that's what you're searching for. With little to no maintenance, MySQL Server may happily cohabit on a desktop or laptop with your other programmes, web servers, and other devices. You may adjust the settings to utilise full RAM, CPU, and I/O capacity if you dedicate an entire computer to MySQL.

- **MySQL Server works in client/server or embedded systems.**

An array of different client applications and libraries, administrative tools, a multi-threaded SQL server that supports a variety of back-ends, and a broad variety of application programming interfaces are all included in the client/server system known as MySQL Database Software (APIs). Additionally, we provide MySQL Server as a multi-threaded integrated library that you can use in your product to build a standalone solution that is more compact, quick, and manageable.

**CHAPTER 4**

**SYSTEM DESIGN**

## 4.1 INTRODUCTION

The initial step in the development process of any designed product or system is design. Design is an artistic process. A good design is essential for an efficient system. The term "design" is described as "the process of applying numerous approaches and concepts to specify a process or a system in sufficient detail to allow its physical realisation." It can be defined as the process of employing multiple methodologies and principles to specify a device, process, or system in sufficient depth to allow physical implementation. Software design is the technical core of the software engineering process and is employed independently of development methodology. The architectural detail required to produce a system or product is developed through the system design. As with any methodical methodology, this programme has gone through the best design phase possible, fine refining all efficiency, performance, and accuracy levels. The design phase is the move from a document for users to a document for programmers or database workers. System design is divided into two stages: logical design and physical design.

## 4.2 UML DIAGRAM

UML is a standard language for describing, visualising, building, and documenting software system artefacts. The Object Management Group (OMG) produced UML, and a draught of the UML 1.0 definition was proposed to the OMG in January 1997.

UML is an abbreviation for Unified Modeling Language. UML is distinct from other mainstream programming languages such as C++, Java, COBOL, and so on. UML is a graphical language that is used to create software blueprints. UML is a general-purpose visual modelling language used to design, specify, build, and document software systems. Although UML is commonly used to represent software systems, it is not confined to this. It is also used to simulate non-software systems. For example, consider the process flow in a manufacturing unit. UML is not a programming language, but UML diagrams can be used to create code in a variety of languages. UML is inextricably linked to object-oriented analysis and design.

After some standardization, UML has become an OMG standard. All the elements, relationships are used to make a complete UML diagram and the diagram represents a system. The visual effect of the UML diagram is the most important part of the entire process. All the other elements are used to make it complete. UML includes the following nine diagrams.

- Class diagram
- Object diagram
- Use case diagram
- Sequence diagram
- Collaboration diagram
- Activity diagram
- Statechart diagram
- Deployment diagram
- Component diagram

#### **4.2.1 USE CASE DIAGRAM**

A use case diagram shows the relationships between system components graphically. To find, explain, and organise system requirements, system analysts employ use cases. The word "system" in this context refers to something that is being created or operated, such as a Web site for the sales and support of mail-order goods. Use case diagrams are a component of UML (Unified Describing Language), a standard language for modelling systems and objects seen in the real world.

Creating general requirements, validating hardware designs, testing and debugging software products while they are still in development, offering online support resources, or executing consumer service-focused tasks are some examples of system objectives. Use cases in a product sales scenario, for instance, may include ordering items, catalogue updates, payment processing, and customer contacts. A use case diagram consists of four sections..

- The border that defines the system of interest with regard to the rest of the world.
- The actors, usually individuals involved with the system defined according to their roles.

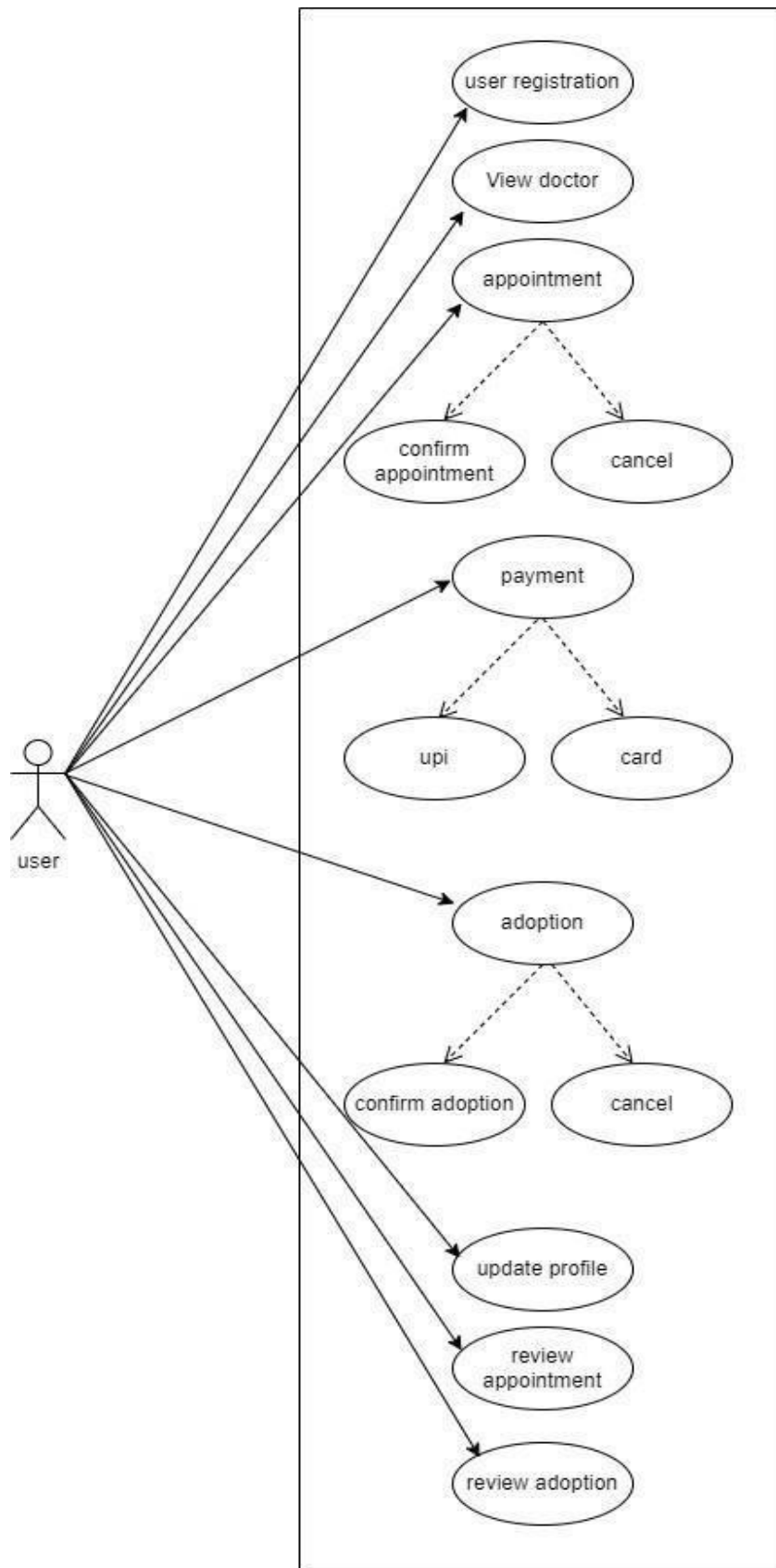


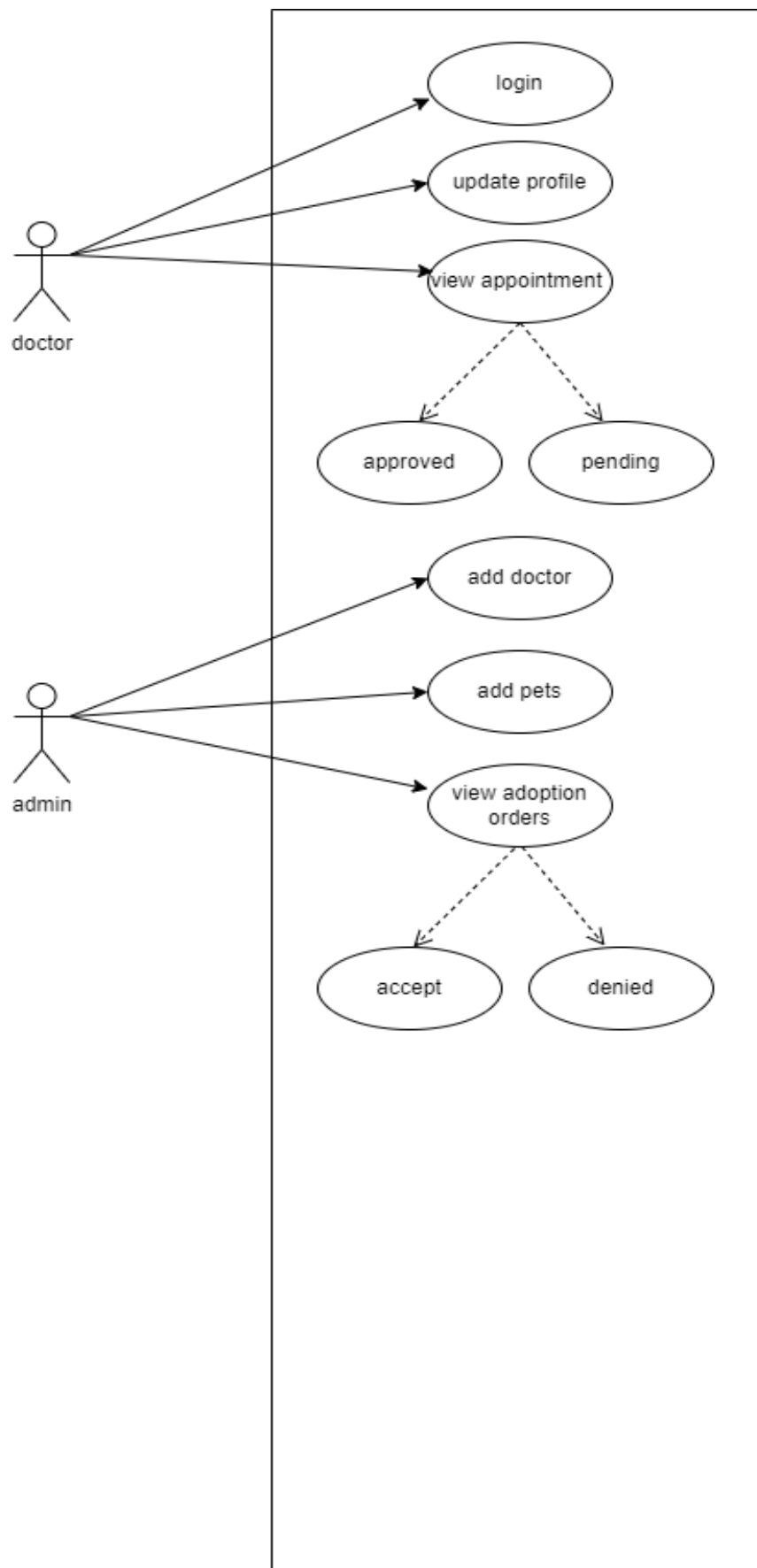
- The use cases, which are the specific roles are played by the actors within and around the system.
- The connections and interactions between the actors and use cases.

The functional requirements of a system are captured using use case diagrams. To develop a successful use case diagram, we must first identify the items mentioned above and then adhere to the following guidelines

- The crucial to pay attention to a use case's name. The name ought to be chosen in a way that makes it clear what functions are being carried out
- Give the actors names that fit them
- Clearly depict links and dependencies in the diagram.
- The primary goal of the diagram is to define the needs, thus avoid attempting to include all possible links..
- When necessary, take notes to help you remember some crucial details.

Fig 1 : Use case diagram for VETERINARY AND PETS MANAGEMENT SYSTEM





---

### 4.2.2 SEQUENCE DIAGRAM

The interactions between items are simply shown in a sequential order, or the order in which they occur, in a sequence diagram. The words event diagrams and event scenarios can also be used to describe a sequence diagram. Sequence diagrams show the actions that the components of a system take. Software engineers and businesspeople alike frequently use these diagrams to describe and comprehend the requirements for both new and current systems.

.

#### Sequence Diagram Notations –

- i. **Actors** – In a UML diagram, an actor represents a type of role that interacts with the system and its objects. It is vital to note that an actor is always beyond the scope of the system that we are attempting to depict using the UML diagram. Actors play a variety of roles, including human users and other external subjects. A stick person notation is used to represent an actor in a UML diagram. A sequence diagram can have several actors.
- ii. **Lifelines** – A lifeline is a named element in a sequence diagram that represents an individual participant. So, in a sequence diagram, each incident is represented by a lifeline. A sequence diagram's lifeline elements are at the top.
- iii. **Messages** – Messages are used to illustrate communication between items. The messages appear on the lifeline in chronological sequence. Messages are represented by arrows. A sequence diagram is built around lifelines and messages.

Messages can be broadly classified into the following categories:

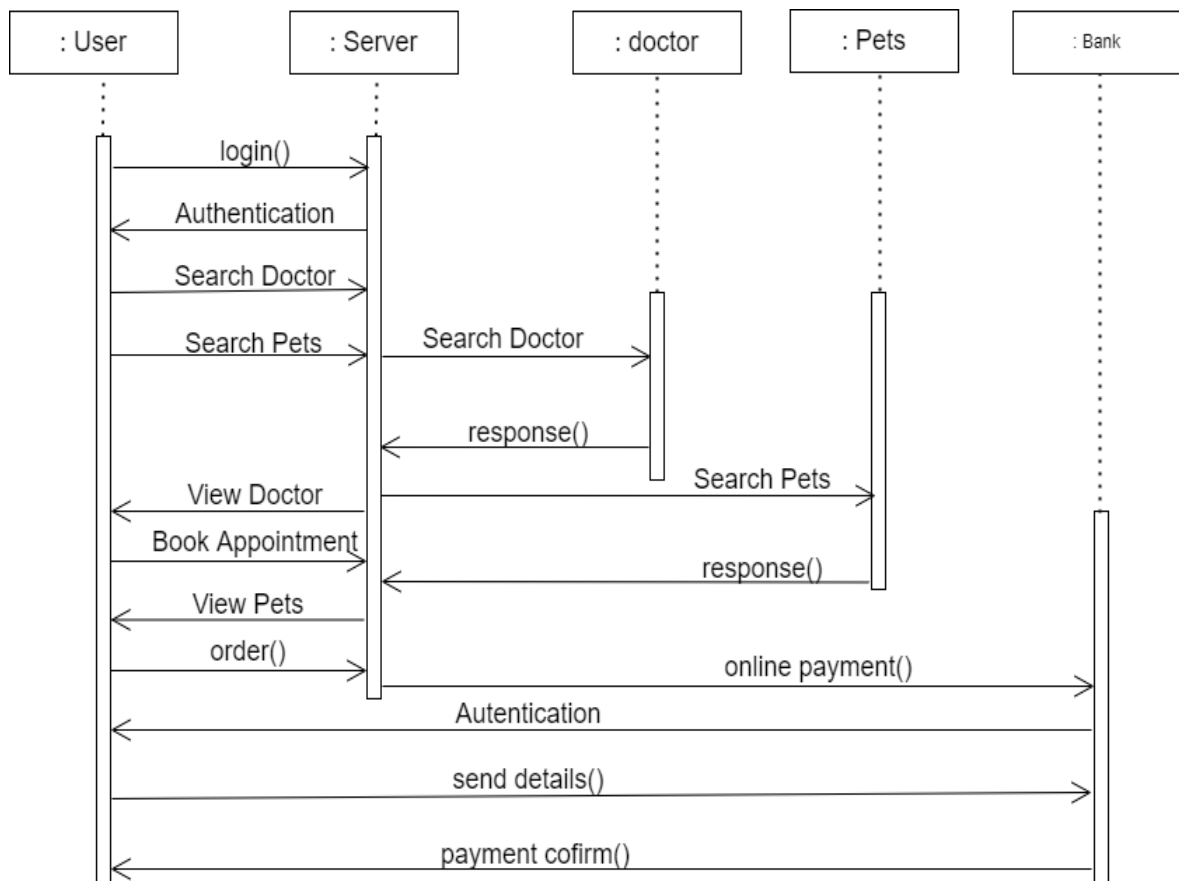
- Synchronous messages
- Asynchronous Messages
- Create message
- Delete Message
- Self-Message
- Reply Message
- Found Message
- Lost Message

**iv. Guards** – In UML, we utilise guards to model circumstances. They are used to restrict the flow of messages under the guise of a condition being met. Guards play a significant function in informing software developers about the limits associated with a system or a certain procedure.

Uses of sequence diagrams –

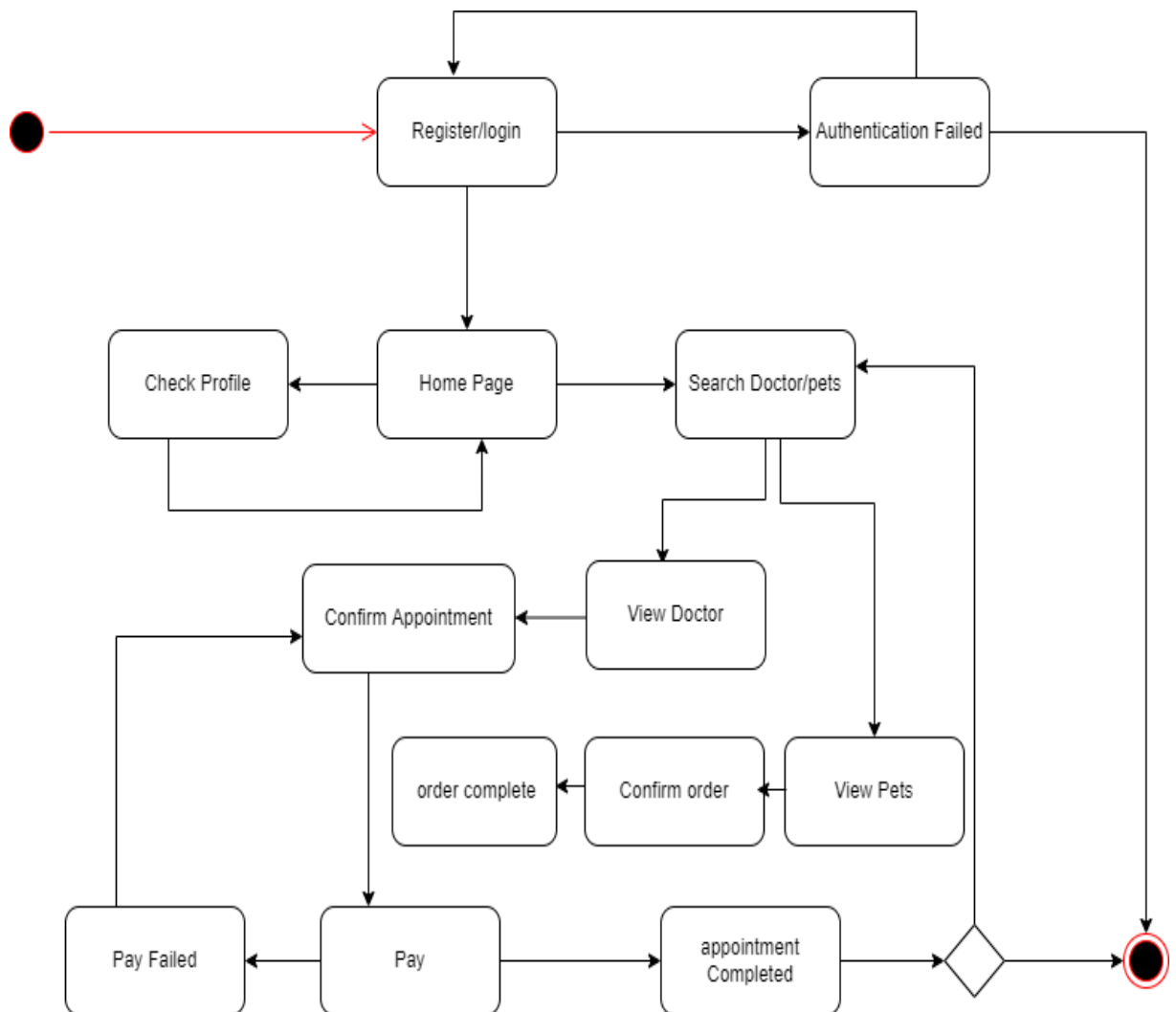
- Used to model and visualize the logic behind a sophisticated function, operation or procedure.
- They are also used to show details of UML use case diagrams.
- Used to understand the detailed functionality of current or future systems.
- Visualise how messages and tasks move between objects or components in a system.

Fig 1 : Sequence diagram for VETERINARY AND PETS MANAGEMENT SYSTEM



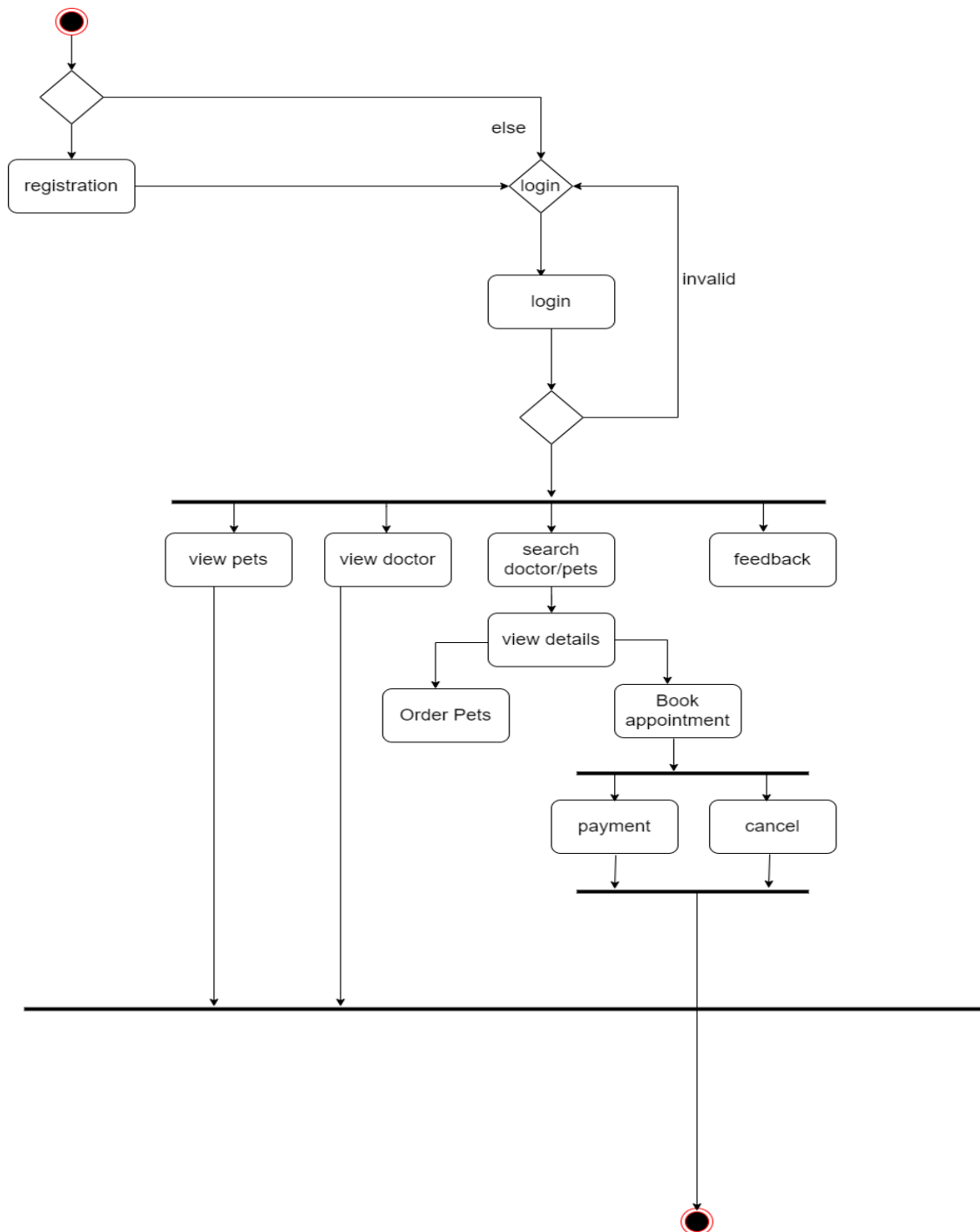
### 4.2.3 State Chart Diagram

A state diagram is used to represent the behavior of a software system. State machine diagrams in UML can be used to model the behavior of a class, a subsystem, a package, or an entire system. It's also referred to as a state chart or a state transition diagram. State chart diagrams are an efficient technique to model the interactions or communication that occur between external entities and within a system. The event-based system is shown by these diagrams. An event is used to control the status of an item. State chart diagrams are used in application systems to describe the various states of an entity.



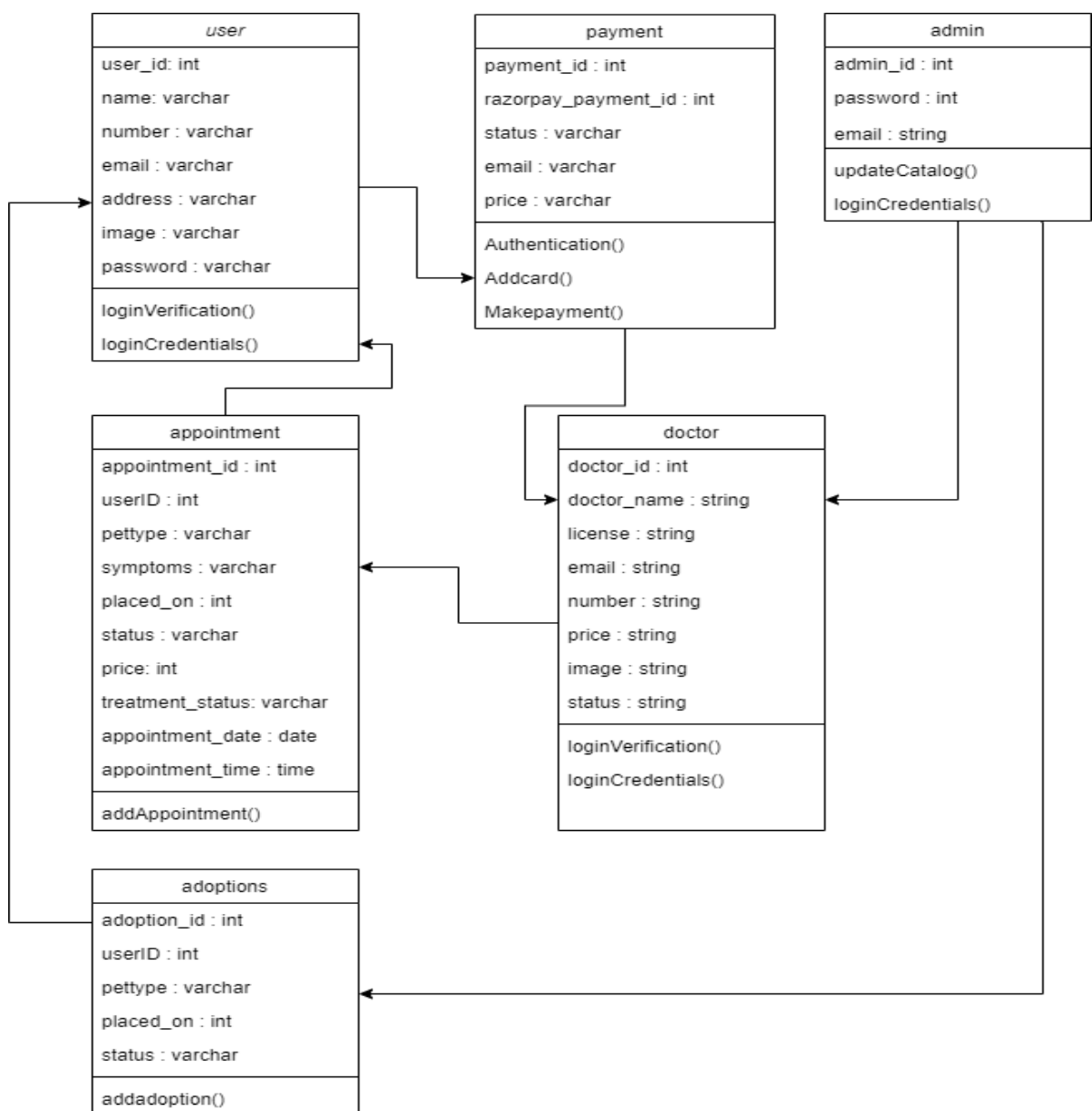
#### 4.2.4 Activity Diagram

Activity Diagrams explain how activities are coordinated to provide a service at various levels of abstraction. Typically, an event must be accomplished by some operations, particularly when the operation is intended to accomplish a number of different things that necessitate coordination, or how the events in a single use case relate to one another, particularly in use cases where activities may overlap and necessitate coordination. It is also appropriate for representing how a set of use cases collaborate to represent business workflows.



#### 4.2.5 Class Diagram

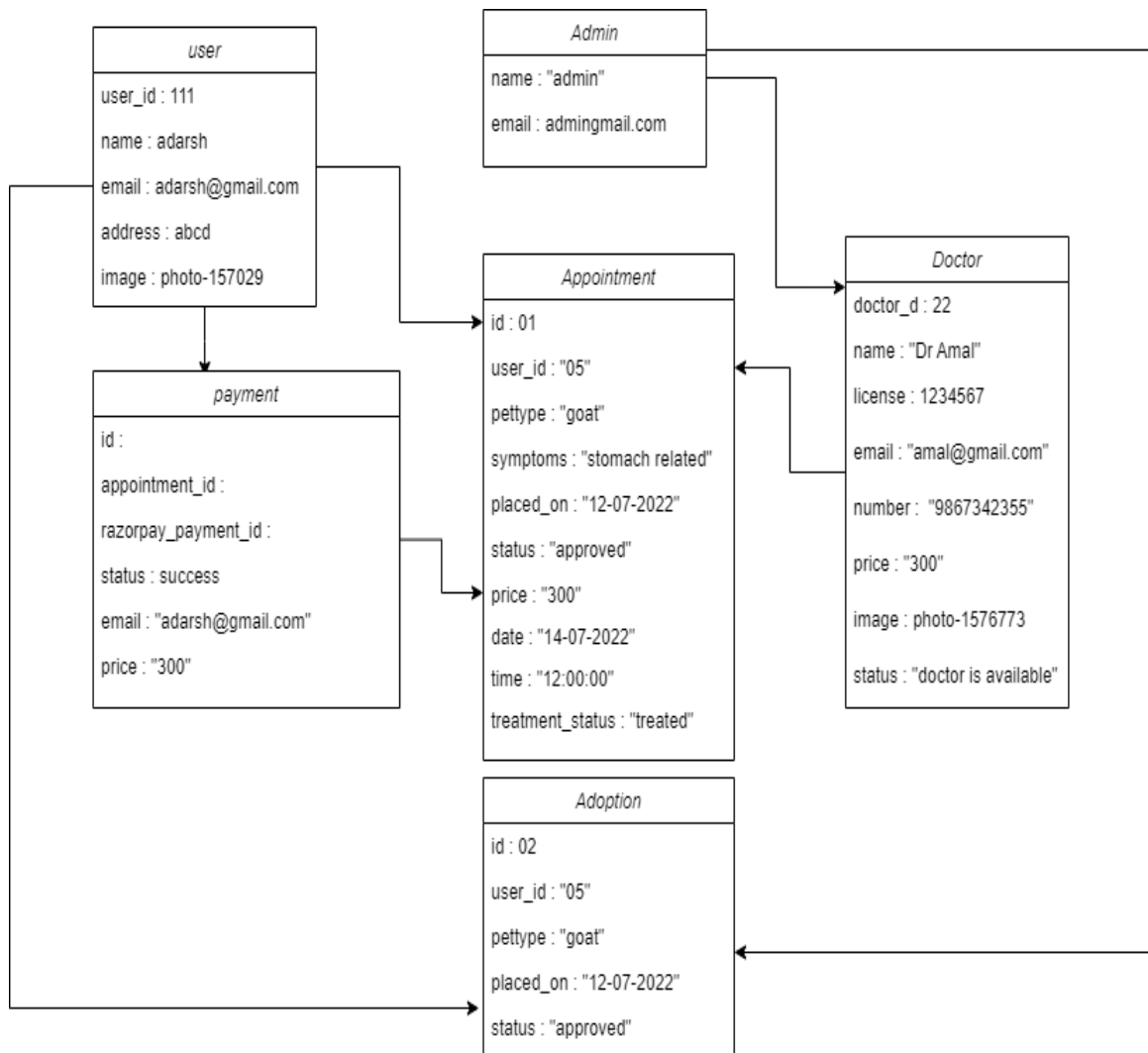
A static diagram is a class diagram. It depicts an application's static view. A class diagram is used not only for visualizing, describing, and documenting many parts of a system, but also for building executable code for a software programmer. A class diagram illustrates a class's attributes and operations, as well as the limitations imposed on the system. Because they are the only UML diagrams that can be mapped directly to object-oriented languages, class diagrams are frequently utilised in the modelling of object-oriented systems. A class diagram is a visual representation of a collection of classes, interfaces, affiliations, collaborations, and constraints. A structural diagram is another name for it.





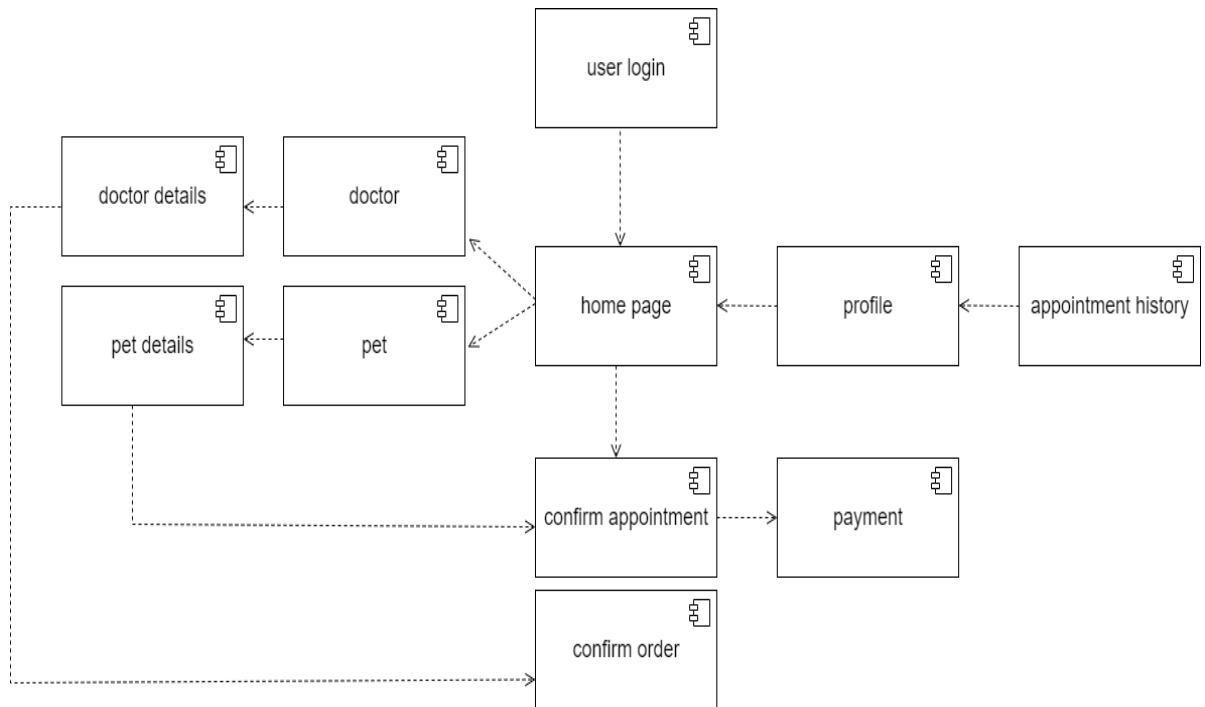
#### 4.2.6 Object Diagram

Object diagrams are evolved from class diagrams, hence they are reliant on class diagrams. A class diagram is represented by an object diagram. The fundamental notions of class and object diagrams are the same. Object diagrams also reflect a system's static view, however this static view is a snapshot of the system at a specific point in time. Object diagrams are used to create an instance of a collection of things and their relationships.



#### 4.2.7 Component Diagram

Component diagrams differ in their nature and behaviour. Component diagrams are used to represent the physical characteristics of a system. Physical aspects are the elements that live in a node, such as executables, libraries, files, documents, and so on. Component diagrams are used to depict the arrangement and relationships of system components. These diagrams are also employed in the creation of executable systems.



#### 4.2.8 Deployment Diagram

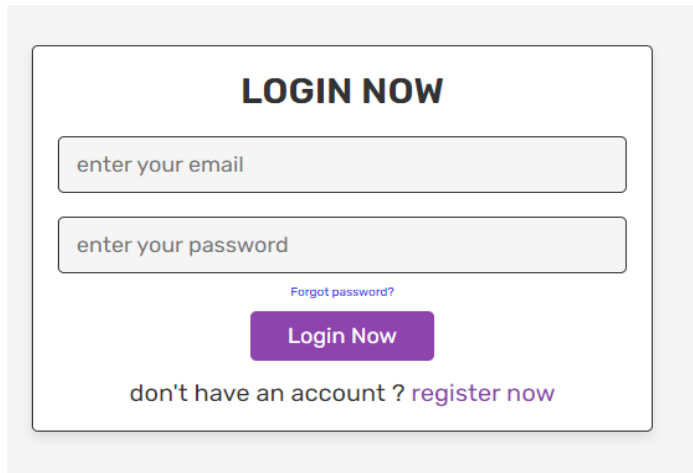
Deployment diagrams are used to depict the topology of a system's physical components, as well as the locations of software components. Deployment diagrams are used to describe a system's static deployment view. Nodes and their relationships are shown in deployment diagrams.



## 4.3 USER INTERFACE DESIGN

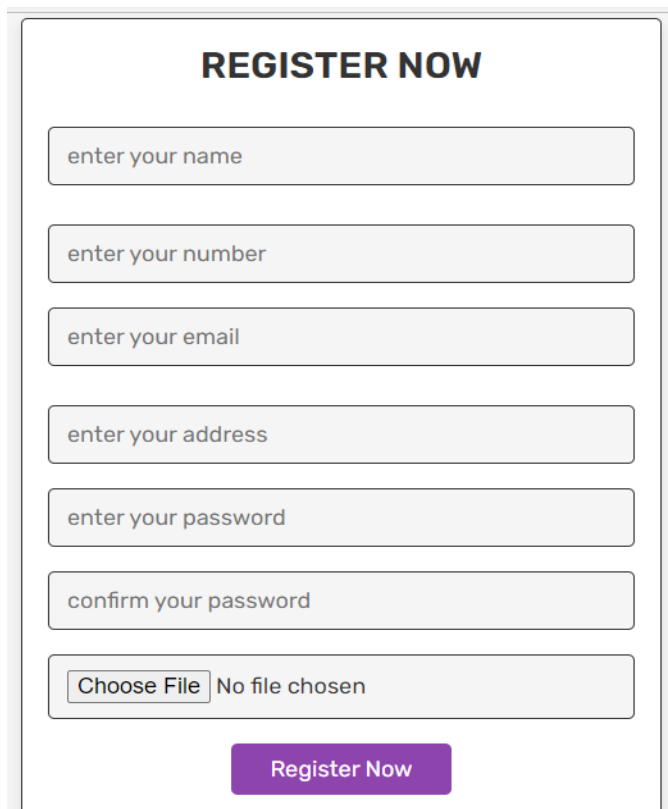
### 4.3.1-INPUT DESIGN

Form Name : Login Page



The login page features a central white box with a purple border. At the top, the text "LOGIN NOW" is displayed in bold black font. Below this, there are two input fields: "enter your email" and "enter your password", both with light gray borders. A blue link "Forgot password?" is positioned below the password field. A purple button labeled "Login Now" is centered below the input fields. At the bottom, the text "don't have an account ? register now" is shown, with "register now" in purple.

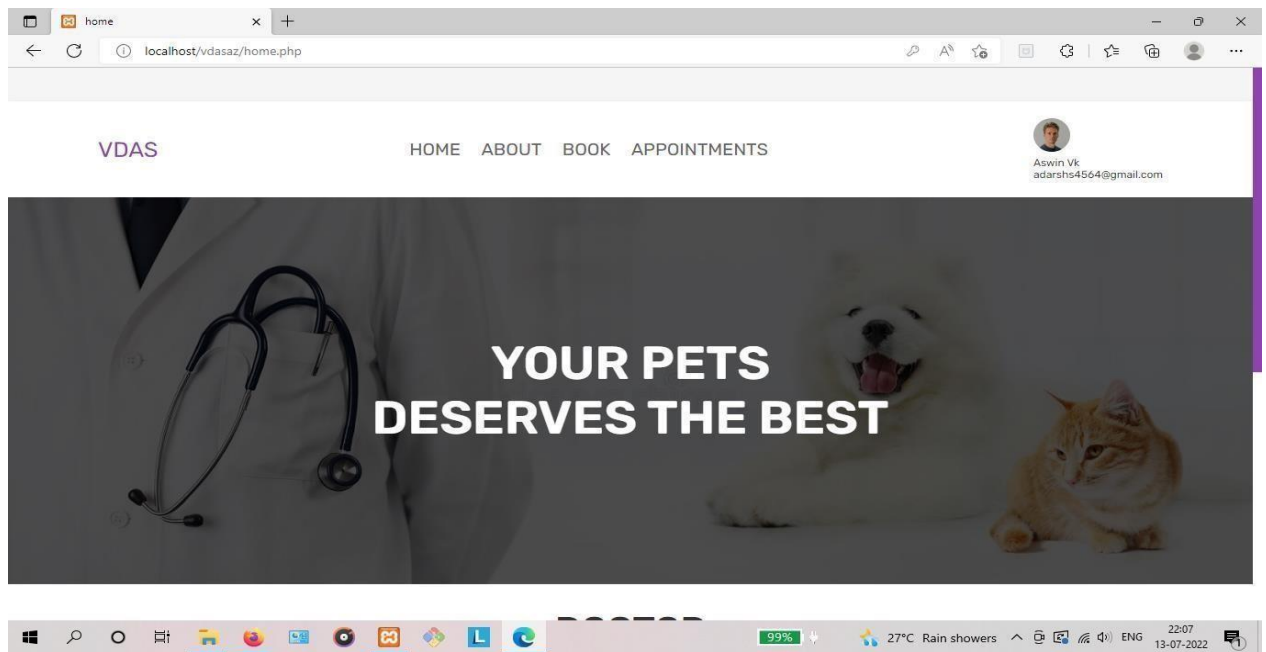
Form Name : User Registration



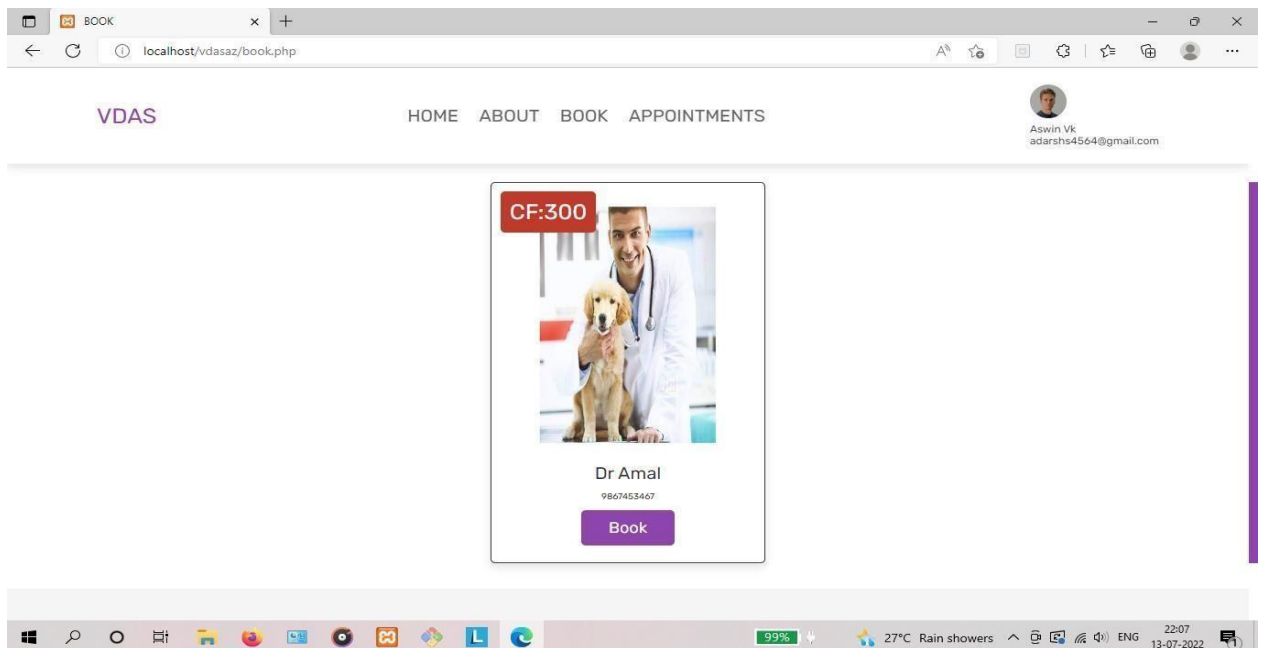
The registration page features a central white box with a purple border. At the top, the text "REGISTER NOW" is displayed in bold black font. Below this, there are six input fields: "enter your name", "enter your number", "enter your email", "enter your address", "enter your password", and "confirm your password", all with light gray borders. Below the password fields, there is a file upload section with a "Choose File" button and the text "No file chosen". A purple button labeled "Register Now" is centered at the bottom of the form.

### 4.3.2 OUTPUT DESIGN

#### Home Page



#### Book Appointment



## 4.4 DATABASE DESIGN

A database is an organised technique for keeping information that allows a user to retrieve stored information in an effective and efficient manner. The objective of any database is to store data, which must be kept secure.

The database design process is divided into two stages. The initial stage is to collect user needs and create a database that meets these requirements as clearly as feasible. This is known as Information Level Design, and it is performed independently of any specific DBMS.

The second stage is to translate this information level design into a design for the specific DBMS that will be used to implement the system in issue. This process is known as Physical Level Design, and it is concerned with the properties of the DBMS that will be used. A database design runs concurrently with a system design. The database's data arrangement aims to meet the two key goals listed below.

- Data Integrity
- Data independence

### 4.6.1 Relational Database Management System (RDBMS)

A relational model depicts a database as a set of relationships. Each relationship is similar to a table of values or a file of records. A row is referred to as a tuple in formal relational model terminology, a column header is referred to as an attribute, and the table is referred to as a relation. A relational database is made up of a collection of tables, each with its own name. A row in a story represents a collection of connected values.

#### Relations, Domains & Attributes

A table is a type of relation. Tuples are the rows of a table. A tuple is a set of  $n$  elements that is ordered. Columns are known as characteristics. Every table in the database has relationships set up. This protects the integrity of both the referential and entity relationships. Domains are collections of atomic values. A common way to specify a domain is to declare a data type from which the domain's data values are pulled. It is also useful to give the domain a name to aid in interpreting its values. Every value in a relation is atomic, that is not decomposable.

### Relationships

- Table relationships are established using Key. The two main keys of prime importance are Primary Key & Foreign Key. Entity Integrity and Referential Integrity Relationships can be established with these keys.
- Entity Integrity enforces that no Primary Key can have null values.
- Referential Integrity enforces that no Primary Key can have null values.
- Referential Integrity for each distinct Foreign Key value, there must exist a matching Primary Key value in the same domain. Other key are Super Key and Candidate Keys.

#### 4.6.2 Normalization

Data are put together in the simplest way possible so that future modifications have the least influence on data structures. Normalization is the formal process of arranging data structures in ways that reduce redundancy and improve integrity. Normalization is the process of separating duplicate fields and dividing a huge table into smaller ones. It is also used to prevent insertion, deletion, and update errors. In normal form, two notions are used in data modelling: keys and relationships. A key is used to uniquely identify a row in a table. Primary keys and foreign keys are the two sorts of keys. A primary key is a table element or combination of components that is used to identify records from the same table. A foreign key is a table column that uniquely identifies records from another table. Up to the third normal form, all of the tables have been normalised.

As the name implies, it refers to returning things to their original state. The application developer attempts to produce a sensible structure of data into correct tables and columns, with names that can be easily correlated to the data by the user, through normalisation. Normalization removes recurring data groupings, avoiding data redundancy, which is a significant drain on computer resources. These include:

- Normalize the data.
- Choose proper names for the tables and columns.
- Choose the proper name for the data.

**First Normal Form**

According to the First Normal Form, an attribute's domain must include only atomic values, and the value of every attribute in a tuple must be a single value from that attribute's domain. To put it another way, 1NF forbids "relations within relations" or "relations as attribute values within tuples." 1NF only allows single atomic or indivisible values for attribute values. The initial step is to convert the data to First Normal Form. This can be accomplished by separating data into different tables with similar data types in each table. Each table is assigned a Primary Key or a Foreign Key based on the project's needs. For each non-atomic attribute, we create a new relationship.

**Second Normal Form**

According to Second Normal Form, no non-key attribute should be functionally dependent on a component of the primary key in relations when the primary key has multiple attributes. We decompose and create a new relation for each partial key and its dependent attributes in this step. Maintain a relationship with the original primary key and any properties that are completely functionally dependent on it. This phase assists in extracting data that is solely dependent on a portion of the key. A connection is said to be in second normal form if and only if it meets all of the first normal form conditions for the primary key and all of the relation's non-primary key qualities are entirely dependent on its primary key alone.

**Third Normal Form**

Relationships should not have a non-key attribute that is functionally determined by another non-key attribute or a group of non-key attributes, according to Third Normal Form. That is, no transitive dependency on the main key should exist. We decompose and establish a relationship that includes non-key attributes that functionally determine other non-key attributes. This step is used to remove everything that does not completely dependant on the Primary Key. A relation is said to be in third normal form if it is solely in second normal form and the non-key qualities of the relation do not depend on each other.



**TABLE DESIGN****Table No        01****Table Name    : tbl\_admin****Primary Key    : admin\_id****Table Description : To store admin login information**

<b>Name</b>	<b>Type(Size)</b>	<b>Constraints</b>	<b>Description</b>
<b>admin_id</b>	Int(20)	Primary Key	Id of admin
<b>name</b>	Varchar2(30)	Not Null	admin's name for login
<b>email</b>	Varchar2(30)	Not Null	admin's email for login
<b>password</b>	Varchar2(20)	Not Null	admin's password for login

**Table No        02****Table Name    : tbl\_user****Primary Key    : user\_id****Table Description : To store user login information**

<b>Name</b>	<b>Type(Size)</b>	<b>Constraints</b>	<b>Description</b>
<b>user_id</b>	Int(20)	Primary Key	Id of user
<b>name</b>	Varchar2(20)	Not Null	Name of user
<b>email</b>	Varchar2(20)	Not Null	email of user
<b>address</b>	Varchar2(20)	Not Null	address of user
<b>password</b>	Varchar2(20)	Not Null	password of user

**Table No        03**

**Table Name     : tbl\_doctor**

**Primary Key    : doctor\_id**

**Table Description: To store doctor details**

<b>Name</b>	<b>Type(Size)</b>	<b>Constraints</b>	<b>Description</b>
<b>doctor_id</b>	Int(20)	Primary Key	Id of doctor
<b>doctor_name</b>	Varchar2(20)	Not Null	Name of doctor
<b>license</b>	Varchar2(20)	Not Null	license of doctor
<b>email</b>	Varchar2(20)	Not Null	Email of doctor
<b>number</b>	Varchar2(20)	Not Null	Number of doctor
<b>fee</b>	Varchar2(20)	Not Null	fee of doctor
<b>image</b>	Varchar2(20)	Not Null	image of doctor
<b>status</b>	Varchar2(20)	Foreign Key	Status of doctor

**Table No            04**

**Table Name        : tbl\_payment**

**Primary Key        : id**

**Foreign Key        : email**

**Table Description: To store payment details**

<b>Name</b>	<b>Type(Size)</b>	<b>Constraints</b>	<b>Description</b>
<b>id</b>	Int(20)	Primary Key	Id of payment
<b>appointment_id</b>	Varchar2(20)	Not Null	Appointment id of payment
<b>razorpay_payment_id</b>	Varchar2(20)	Not Null	Razorpay payment id
<b>status</b>	Varchar2(20)	Not Null	status of payment
<b>email</b>	Varchar2(20)	Foreign Key	Foreign key from user table
<b>price</b>	Int(20)	Not Null	Price of payment

**Table No            05**

**Table Name        : tbl\_pets**

**Primary Key        : pet\_id**

**Table Description: To store pets details**

<b>Name</b>	<b>TYPE(Size)</b>	<b>Constraints</b>	<b>Description</b>
<b>pet_id</b>	Int(10)	Primary Key	Id of pets
<b>pet_name</b>	Varchar2(20)	Not Null	Name of pets
<b>image</b>	Varchar2(20)	Not Null	Image of pets
<b>status</b>	Varchar2(20)	Not Null	Status of pets

**Table No        06**

**Table Name     : tbl\_appointment**

**Primary Key    : app\_id**

**Foreign Key     : user\_id**

**Table Description: To store appointment details**

<b>Name</b>	<b>Type(Size)</b>	<b>Constraints</b>	<b>Description</b>
<b>app_id</b>	Int(20)	Primary Key	Id of appointment
<b>User_id</b>	Int(20)	Foreign Key	Foreign key from user table
<b>pettype</b>	Varchar2(20)	Not Null	Pettytype of appointment
<b>symptoms</b>	Varchar2(20)	Not Null	Symptoms of appointment
<b>placed_on</b>	Date(10)	Not Null	Placed_on of appointment
<b>status</b>	Varchar2(20)	Not Null	Status of appointment
<b>fee</b>	Int(3)	Not Null	Fee of appointment
<b>Date</b>	Date(10)	Not Null	Date of appointment
<b>time</b>	Time(6)	Not Null	Time of appointment
<b>treatmen_status</b>	Varchar2(20)	Not Null	Treatment status of appointment

**Table No        07**

**Table Name     : tbl\_adoption**

**PrimaryKey     : adoption\_id**

**Foreign Key     : user\_id**

**Table Description: To store adoption details**

<b>Name</b>	<b>TYPE(Size)</b>	<b>Constraints</b>	<b>Description</b>
<b>adoption_id</b>	Int(20)	Primary Key	Id of adoption
<b>user_id</b>	Int(20)	Foreign Key	Foreign Key from user table
<b>name</b>	Varchar2(20)	Not Null	name of adoption
<b>email</b>	Varchar2(30)	Not Null	Email of adoption
<b>address</b>	Varchar2(20)	Not Null	Address of adoption
<b>status</b>	Varchar2(20)	Not Null	Status of adoption
<b>placed_on</b>	Date(10)	Not Null	Placed on of adoption

## **CHAPTER 5**

### **SYSTEM TESTING**

## 5.1 INTRODUCTION

Software testing is the controlled execution of software in order to answer the question, "Does the software behave as specified?" The terms software testing and verification and validation are sometimes used interchangeably. The inspection or testing of goods, including software, for conformance and consistency with an accompanying specification is known as validation. Software testing is only one type of verification; other methods include reviews, analysis, inspections, and walkthroughs. Validation is the process of ensuring that what has been specified is actually what the user desired.

Static and dynamic analysis are two other processes that are frequently connected with software testing. Static analysis examines the source code of software for issues and metrics without actually running the code. Dynamic analysis examines software behaviour while it is running to offer information such as execution traces, timing profiles, and test coverage.

Testing is a series of activities that can be planned ahead of time and carried out in a methodical manner. Testing begins with modules and progresses to the integration of the full computer-based system. Nothing is complete without testing, as testing objectives are critical to the success of the system. There are various guidelines that might serve as testing objectives. They are:

Testing is a process of executing a program with the intent of finding an error.

- A good test case is one that has high possibility of finding an undiscovered error.
- A successful test is one that uncovers an undiscovered error.

If a testing is conducted successfully according to the objectives as stated above, it would uncover errors in the software. Also testing demonstrate that the software function appear to be working according to the specification, that performance requirement appear to have been met.

There are three ways to test program.

- For correctness
- For implementation efficiency
- For computational complexity

Test for correctness are supposed to verify that a program does exactly what it was designed to do. This is much more difficult than it may at first appear, especially for large programs.

## 5.2 TEST PLAN

A test plan denotes a series of intended actions to be taken in order to complete various testing methodologies. The Test Plan serves as a blueprint for the action to be taken. Software engineers build computer programmes, documentation, and data structures. The software developers are always in charge of evaluating the individual parts of the programmes to ensure that they accomplish the function for which they were developed. An independent test group (ITG) exists to eliminate the inherent issues involved with allowing the builder to test the finished product. The testing objectives should be specified in measurable ways. As a result, the mean time to failure, the cost of finding and fixing the flaws, the remaining defect density or frequency of occurrence, and test work-hours per regression test should all be specified in the test plan. The levels of testing include:

- Unit testing
- Integration Testing
- Data validation Testing
- Output Testing

### 5.2.1 Unit Testing

The smallest unit of software design - the software component or module - is the subject of unit testing. Important control pathways are checked within the module's border using the component level design description as a guide. Unit testing's relative difficulty and uncovered scope are established. Unit testing is white-box in nature, and numerous components can be tested in parallel. The modular interface is checked to ensure that data flows into and out of the software unit under test effectively. The local data structure is inspected to guarantee that data stored temporarily retains its integrity throughout all phases in the execution of an algorithm. To ensure that all statements in a module have been performed at least once, boundary conditions are tested. Finally, all error handling paths are put through their paces.



Before starting any other tests, data flow over a module interface must be tested. All other tests are rendered ineffective if data does not enter and exit appropriately. During the unit test, it is critical to perform selective testing of execution pathways. Error circumstances should be expected and error handling paths should be set up to reroute or cleanly end operations when an error occurs. The final duty of the unit testing process is boundary testing. Software frequently fails at its limits.

In the Sell-Soft System, unit testing was performed by treating each module as a separate entity and testing each one with a wide range of test inputs. Some weaknesses in the internal logic of the modules were discovered and fixed. Following coding, each module is tested and run independently. All extraneous code was deleted, and all modules were tested to ensure that they functioned properly and produced the intended results.

### **5.2.2 Integration Testing**

Integration testing is a methodical technique for building the programme structure while also conducting tests to detect interfacing issues. The goal is to use unit-tested components and develop a programme structure governed by design. The entire programme is tested in its entirety. Correction is difficult since isolating causes is confounded by the program's enormous scope. Once these flaws are addressed, new ones develop, and the process appears to loop indefinitely. Following unit testing in the system, all modules were integrated to test for interface inconsistencies. Furthermore, discrepancies in programme architectures were eliminated, and a single programme structure formed.

### **5.2.3 Validation Testing or System Testing**

This is the last stage of testing. The complete system was tested in its entirety, including all forms, code, modules, and class modules. This type of testing is sometimes referred to as Black Box testing or System testing.

The Black Box testing method focuses on the software's functional requirements. In other words, Black Box testing allows a software engineer to generate sets of input conditions that fully exercise all functional requirements for a programme. Black Box testing looks for defects in the following categories: erroneous or missing functions, interface errors, data structure or external data access errors, performance errors, initialization mistakes, and termination faults.

### 5.2.4 Output Testing or User Acceptance Testing

The system under consideration has been tested for user acceptance; it should meet the needs of the firm. The software should maintain contact with the prospective system and user while developing and making changes as needed. This is done in relation to the following points:

- Input Screen Designs,
- Output Screen Designs

The aforesaid testing is carried out using several types of test data. The preparation of test data is critical in system testing. The system under study is tested using the test data that has been prepared. While testing the system, test data mistakes are discovered and fixed using the above testing processes, and the corrections are also recorded for future use.

## Test cases for User Registration

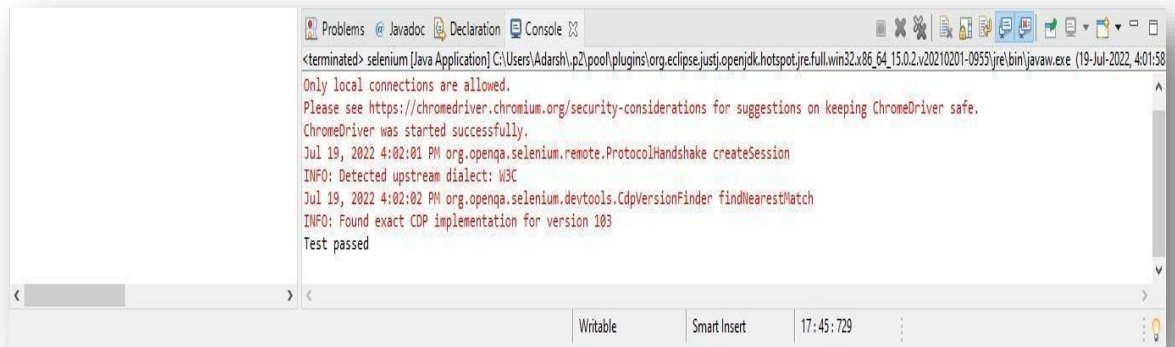
Project Name: Veterinary and Pets Management System					
Updation Test Case					
Test Case ID: registration			Test Designed By: Adarsh S		
Test Priority (Low/Medium/High): High			Test Designed Date: 17-05-2022		
Module Name: Login Screen			Test Executed By: Ms. Merin Chacko		
Test Title: User Registration Details			Test Execution Date: 18-05-2022		
Description: Register to system and Registration is completed then login , if some error occurs, test will fail					
Pre-Condition: User has valid user name and password					
Step	Test Step	Test Data	Expected Result	Actual Result	Status (Pass/Fail)
1	Navigation to Register Page		Register Page shouldbe	Registrtrion page displayed	Pass
2	Provide Valid Registration details	User Name: adarsh s4564@gmail.com	User should be able to Register	User registrion Completed after go to the login page	Pass
3					
4	Click on Login button				
5	Provide profile details	Input profile details	User will be redirected to Login page	Use will be redirected to Login page	Pass
7	Click on register button				
8	Provide invalid information	Input invalid profile details.	User will be stay in register page	User will be stay on that page showing error message	Pass
9	Click on register button				
Post-Condition: User is validated with database and successfully login into account. The Account session details are logged in database.					

**Code**

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class Testing
{
    public static void main(String[] args)
    {
        System.setProperty("webdriver.chrome.driver","C:\\Users\\Adarsh\\eclipse-workspace\\Selenium
        Testing\\chromedriver.exe");
        WebDriver driver=new
        ChromeDriver();
        driver.get("http://localhost/vdas");
        driver.findElement(By.name("email")).sendKeys("adarshs4564@gmail.com");
        driver.findElement(By.name("password")).sendKeys("123456");
        driver.findElement(By.name("submit")).click();
        String
        actualUrl="http://localhost/vdas/home.php";
        String expectedUrl= driver.getCurrentUrl();
        if(actualUrl.equalsIgnoreCase(expectedUrl))
        {
        }
        else
        {
        }
    }
}
```

## Output



The screenshot shows an IDE console window with the following output:

```
<terminated> selenium [Java Application] C:\Users\Adarsh.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_15.0.2.v20210201-0955\jre\bin\javaw.exe (19-Jul-2022, 4:01:58)
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe.
ChromeDriver was started successfully.
Jul 19, 2022 4:02:01 PM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected upstream dialect: W3C
Jul 19, 2022 4:02:02 PM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
INFO: Found exact CDP implementation for version 103
Test passed
```

The console window includes standard IDE controls like tabs (Problems, Javadoc, Declaration, Console), a toolbar, and a status bar at the bottom showing 'Writable', 'Smart Insert', and a time indicator '17:45:729'.

## **CHAPTER 6**

### **IMPLEMENTATION**

## 6.1 INTRODUCTION

The implementation stage of a project is when the theoretical design is transformed into a working system. It is the most important stage in developing a successful new system since it gives users trust that the new system will work, be effective, and accurate. Its primary focus is on user training and documentation. Conversion normally occurs at the same time as the user is being instructed, or later. Implementation is the process of translating a new improved system design into an operational one.

At this point, the user department is responsible for the majority of the work, the most disruption, and the most influence on the existing system. Chaos and confusion might result if the implementation is not thoroughly planned and controlled.

Implementation encompasses all activities undertaken to convert from the present system to the new system. The new system could be completely new, replacing an existing human or automated system, or a tweak to an existing system. Proper implementation is required to produce a dependable system that meets the needs of the organisation. System implementation refers to the process of putting the produced system into real use. This comprises all activities required to transition from the old to the new system. The system can only be installed after extensive testing and confirming that it meets the standards. The system personnel examine the system's practicality. The more the complexity of the system being implemented, the greater the effort necessary for system analysis and design to accomplish the three key aspects: education and training, system testing, and changeover.

The implementation state involves the following tasks:

- ☐ Careful planning.
- ☐ Investigation of system and constraints.
- ☐ Design of methods to achieve the changeover.

## 6.2 IMPLEMENTATION PROCEDURES

The ultimate installation of the package in its real surroundings, to the satisfaction of the intended purposes and the operation of the system, is referred to as software implementation. Many organisations will commission the software development project by someone who will not be operating it. People are sceptical of the programmer at first, but we must persevere.

Ensuring that resistance does not build up, since it is necessary to ensure that:

- ☐ The active user must be aware of the benefits of using the new system.
- ☐ Their confidence in the software is built up.
- ☐ Proper guidance is imparted to the user so that he is comfortable in using the application.

Before proceeding to view the system, the user should be aware that in order to view the results, the server software must be operating on the server. The real process will not take place if the server object is not up and running on the server.

### **6.2.1 User Training**

User training is intended to prepare the user for system testing and conversion. To achieve the goals and benefits anticipated from a computer-based system, the people who will be participating must be confident in their role in the new system. The demand for training grows as the system becomes more sophisticated. User training teaches the user how to enter data, reply to error signals, query the database, and invoke routines that generate reports and do other important operations.

### **6.2.2 Training on the Application Software**

After receiving the essential fundamental computer awareness training, the user will need to be instructed on the new application software. This will provide the fundamental philosophy of the new system's use, such as the screen flow, screen design, kind of help on the screen, type of mistakes while entering data, the associated validation check at each entry, and methods to correct the date entered. It should then cover the information required by the specific user/group to use the system or a portion of the system while imparting programme training on the application. This training may change amongst user groups and at different levels of organisation.

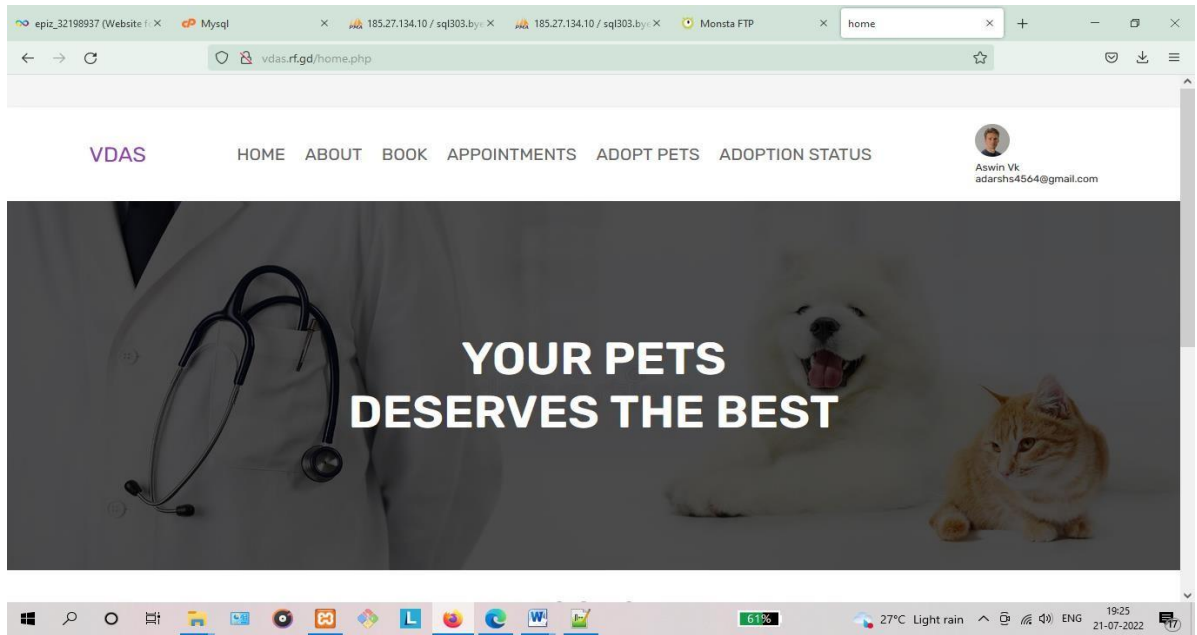
### **6.2.3 System Maintenance**

The riddle of system development is maintenance. The maintenance phase of the software life cycle is when a software product does valuable work. After a system has been effectively implemented, it must be properly maintained. System maintenance is a critical component of the software development life cycle. The purpose of system maintenance is to make the system more flexible to changes in the system environment. Of all, software maintenance entails significantly more than "Finding Mistakes."



### 6.2.4 Hosting

Infinityfree is a free website hosting solution that provides an array of valuable features, including a website builder, WordPress support, and no ads.



## **CHAPTER 7**

### **CONCLUSION AND FUTURE SCOPE**

## 7.1 CONCLUSION

Veterinary and Pets Management System is a website that provides useful information to the users. User can fillup online appointment form through the website. This project was expected to become of the most useful systems for veterinary service because clients can avoid wasting their time, energy and money.

## 7.2 FUTURE SCOPE

- Should provide more number of doctors.
- Data security can be enhanced.
- Users can able to do advanced search options
- Users can able to add complaints and feed backs etc.

## **CHAPTER 8**

## **BIBLIOGRAPHY**

**REFERENCES:**

- Gary B. Shelly, Harry J. Rosenblatt, “*System Analysis and Design*”, 2009.
- Roger S Pressman, “*Software Engineering*”, 1994.
- PankajJalote, “*Software engineering: a precise approach*”, 2006.
- James lee and Brent ware Addison, “Open source web development with LAMP”, 2003
- IEEE Std 1016 Recommended Practice for Software Design Descriptions.

**WEBSITES:**

- [www.w3schools.com](http://www.w3schools.com)
- [www.jquery.com](http://www.jquery.com)
- <https://app.diagrams.net>
- <http://homepages.dcc.ufmg.br/~rodolfo/es-1-03/IEEE-Std-830-1998.pdf>
- [www.agilemodeling.com/artifacts/useCaseDiagram.html](http://www.agilemodeling.com/artifacts/useCaseDiagram.html)

## **CHAPTER 9**

## **APPENDIX**

## 9.1 Sample code

### User

#### Login.php

```
<?php

include
'config.p
hp';
session_s
tart();
if(isset($_POST['mail'])=="forgot Password"){

    $html="http://localhost/veter/f

    orgot.php";

    include('smtp/PHPMailerAuto

    load.php');
    $mail=new PHPMailer(true);
    $mail->isSMTP();
    $mail->Host="smtp.gmail.com";
    $mail->Port=587;
    $mail->SMTPSecure="tls";
    $mail->SMTPAuth=true;
    $mail->Username="veterinary545@gmail.com";
    $mail->Password="wqizfelmgvgtkcda";
    $mail->SetFrom("veterinary545@gmail.com");
    $mail->addAddress("adarshh769@gmail.com");
    $mail->IsHTML(true);
    $mail->Subject="New Contact Us";
    $mail->Body=$html;
    $mail-

    >SMTPOptions=array('ssl'=>array(

    'verify_peer'=>false,

    'verify_peer_name'=>false,

    'allow_self_signed'=>false
    ));
    if($mail->send()){
```

```
        echo "Mail send";
    }
    else{

        echo "Error occur";
    }
}

else if(isset($_POST['submit']))
$email= mysqli_real_escape_string($conn, $_POST['email']);
$pass = mysqli_real_escape_string($conn, md5($_POST['password']));
$select_users = mysqli_query($conn, "SELECT * FROM `users` WHERE email = '$email' AND

password = '$pass'") or die('query failed');

if(mysqli_num_rows($select_users) > 0){

    $row = mysqli_fetch_assoc($select_users);
    $_SESSION['user_name'] = $row['name'];
    $_SESSION['user_email'] = $row['email'];
    $_SESSION['user_id'] = $row['id'];
    header('location:home.php');

}
else{

    echo '<script>alert("invalid email or password")</script>';
}

$select_admin = mysqli_query($conn, "SELECT * FROM `admin` WHERE email = '$email' AND
password = '$pass'") or die('query failed');

if(mysqli_num_rows($select_admin) > 0){

    $row = mysqli_fetch_assoc($select_admin);
    $_SESSION['admin_name'] = $row['name'];
    $_SESSION['admin_email'] = $row['email'];
    $_SESSION['admin_id'] = $row['id'];
    header('location:admin_page.php');

}

$select_doctor = mysqli_query($conn, "SELECT * FROM `doctor` WHERE email = '$email' AND
password = '$pass'") or die('query failed');

if(mysqli_num_rows($select_doctor) > 0){

    $row = mysqli_fetch_assoc($select_doctor);
    $_SESSION['doctor_name'] = $row['name'];
    $_SESSION['doctor_email'] = $row['email'];
```



```
$_SESSION['doc_id'] = $row['id'];
header('location:doctor_page.php');

}

}
?>

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>login</title>

  <!-- font awesome cdn link -->
  <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.0.0/css/all.min.css">

  <!-- custom css file link -->
  <link rel="stylesheet" href="css/style.css">

</head>
<body>

<div class="form-container">

  <form action="" method="post">
    <h3>login now</h3>

    <input type="email" name="email" placeholder="enter your email" class="box">
    <input type="password" name="password" placeholder="enter your password" class="box">
    <div class="link forget-pass text-left"><a href="reset.php">Forgot password?</a>
    </div>

    <input type="submit" name="submit" value="login now" class="btn">
    <p>don't have an account ? <a href="register.php">register now</a></p>

  </form>

</div>

</body>
</html>
```

**Book.php**

```
<?php

include 'config.php';

session_start();

$user_id = $_SESSION['user_id'];

if(!isset($user_id)){
    header('location:login.php');
}

$sql="select * from appointments where user_id=$user_id; ";
$sql_exe=mysqli_query($conn,$sql);
$s1=mysqli_num_rows($sql_exe);

if(isset($_POST['checkout'])){

    $doctor_name = $_POST['doctor_name'];
    $doctor_price = $_POST['doctor_price'];
    $doctor_number = $_POST['doctor_number'];
    $doctor_image = $_POST['doctor_image'];

}

?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>BOOK</title>

    <!-- font awesome cdn link -->
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/font-awesome/6.0.0/css/all.min.css">

    <!-- custom css file link -->
    <link rel="stylesheet" href="css/style.css">
```

```

<style>
    .isDisabled {
color: currentColor;
cursor: not-allowed;
opacity: 0.5;
text-decoration: none;
pointer-events: none;
}
</style>

</head>
<body>

<?php include 'header.php'; ?>

<div class="heading">
    <h3>book appointment</h3>
    <p> <a href="home.php">home</a> / book </p>
</div>

<section class="doctors">

    <h1 class="title">DOCTOR</h1>

    <div class="box-container">

        <?php
            $select_doctors = mysqli_query($conn, "SELECT * FROM `doctor`") or die('query failed');
            if(mysqli_num_rows($select_doctors) > 0){
                while($fetch_doctors = mysqli_fetch_assoc($select_doctors)){
                    ?>
                    <form action="" method="post" class="box">
                        
                        <div class="name"><?php echo $fetch_doctors['name']; ?></div>
                        <div class="number"><?php echo $fetch_doctors['number']; ?></div>
                        <div class="price">CF:<?php echo $fetch_doctors['price']; ?></div>

                        <input type="hidden" name="doctor_name" value="<?php echo $fetch_doctors['name']; ?>">
                        <input type="hidden" name="doctor_price" value="<?php echo $fetch_doctors['price']; ?>">
                        <input type="hidden" name="doctor_number" value="<?php echo $fetch_doctors['number']; ?>">
                        <input type="hidden" name="doctor_image" value="<?php echo $fetch_doctors['image']; ?>">
                        <p><span style="color:<?php if($fetch_doctors['status'] == 'TODAY DOCTOR IS LEAVE'){
echo 'red'; }else{ echo 'green'; } ?>";><?php echo $fetch_doctors['status'];

```

```
<?php
if($s1>0){
    ?>
    <h2 style="color: red; font-size: 16px"> <?php echo "Appointment already booked" ?></h2>
    <?php
}else{
    ?>

<?php
if($fetch_doctors['status'] == 'TODAY DOCTOR IS LEAVE')
{
    ?>
    <a class = "isDisabled" href = "checkout.php" class = "btn" name = "order_btn" >BOOK</a>
    <h2 style="color: red; font-size: 16px"> <?php echo "" ?></h2>
    <?php
}else{
    ?>
    <a href="checkout.php" class="btn" name="order_btn">book</a>
    <?php
    }
    ?>

    <?php
    }
    ?>
</form>
<?php
    }
}else{
    echo '<p class="empty">no doctors added yet!</p>';
    }
    ?>
</div>

</section>
<?php include 'footer1.php'; ?>

<!-- custom js file link -->
<script src="js/script.js"></script>
</body>
</html>
```

---

**Appointment.php**

```
<?php

include 'config.php';

session_start();

$user_id = $_SESSION['user_id'];

if(!isset($user_id)){
    header('location:index.php');
}

if(isset($_GET['delete'])){
    $delete_id = $_GET['delete'];
    mysqli_query($conn, "DELETE FROM `appointments` WHERE id = '$delete_id'") or die('query failed');
    header('location:appointments.php');
}

?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>appointments</title>

    <!-- font awesome cdn link -->
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/font-awesome@6.0.0/css/all.min.css">

    <!-- custom css file link -->
    <link rel="stylesheet" href="css/style.css">

</head>
<body>

<?php include 'header.php'; ?>

<div class="heading">
    <h3>your appointments</h3>
    <p> <a href="home.php">home</a> / appointments </p>
</div>

<section class="placed-appointments">

    <h1 class="title">placed appointments</h1>

    <div class="box-container">

        <?php
        $appointment_query = mysqli_query($conn, "SELECT * FROM `appointments` WHERE user_id =
```

---

---

```

$user_id") or die('query failed');
    if(mysqli_num_rows($appointment_query) > 0){
        while($fetch_appointments = mysqli_fetch_assoc($appointment_query)){
            ?>
            <div class="box">
                <p> Placed on : <span><?php echo $fetch_appointments['placed_on']; ?></span> </p>

                <p> Pettype : <span><?php echo $fetch_appointments['pettype']; ?></span> </p>

                <p>Appointment Date : <span><?php echo $fetch_appointments['Date']; ?></span> </p>
                <p>Appointment Time : <span><?php echo $fetch_appointments['Time']; ?></span> </p>
                <p> Status : <span style="color:<?php if($fetch_appointments['status'] == 'REJECTED'){ echo
'red'; }else{ echo 'green'; } ?>";><?php echo $fetch_appointments['status']; ?></span> </p>

                <p> Consultancy Fee : <span><?php echo $fetch_appointments['price']; ?></span> </p>
                <p> Treatment status : <span><?php echo $fetch_appointments['treatment_status'];
?></span> </p>

            <?php
            $appointment_query = mysqli_query($conn, "SELECT * FROM `orders`") or die('query
failed');
            if(mysqli_num_rows($appointment_query) > 0){
                while($fetch_appointments = mysqli_fetch_assoc($appointment_query)){
                    ?>

                    <p> razorpay_payment_id : <span><?php echo $fetch_appointments['razorpay_payment_id'];
?></span> </p>

                    <p> status : <span><?php echo $fetch_appointments['status']; ?></span> </p>

                }
            }
            ?>

            <?php
            }
            }else{
                echo '<p class="empty">Payment is not done yet!</p>';
            }
            ?>

            <center><button id="PrintButton" onclick="PrintPage()" class ="option-
btn">Download</button></center>
            <center><a href = "http://localhost/vdasaz/RazorPay/" class ="option-btn">Pay
Now</a></center>
            <a href="appointments.php?delete=<?php echo $fetch_appointments['id']; ?>"
onclick="return confirm('cancel this appointment?');" class="delete-btn">cancel</a>
            </div>
            <?php
            }
            }else{
                echo '<p class="empty">no appointments placed yet!</p>';
            }
            }

```

---

```
?>
</div>

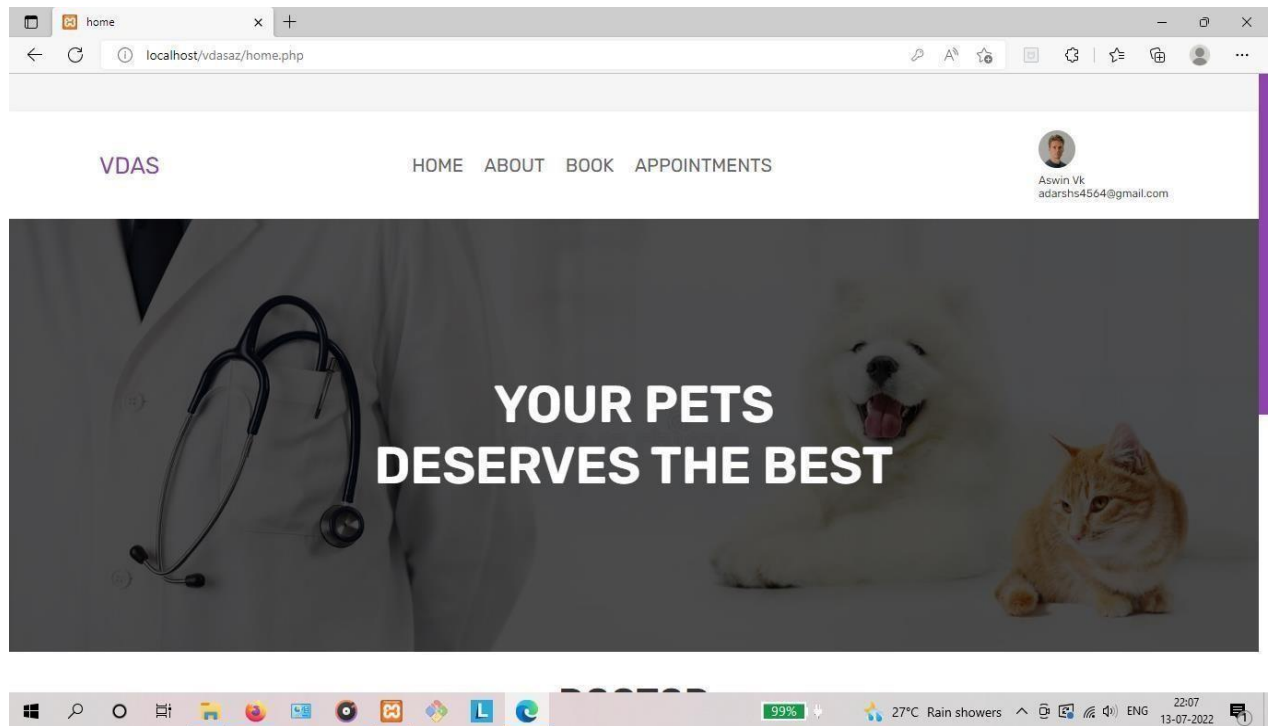
</section>
<?php include 'footer1.php'; ?>
<!-- custom js file link -->
<script src="js/script.js"></script>
<script type="text/javascript">
    function PrintPage() {
        window.print();
    }
</script>

</body>
</html>
```

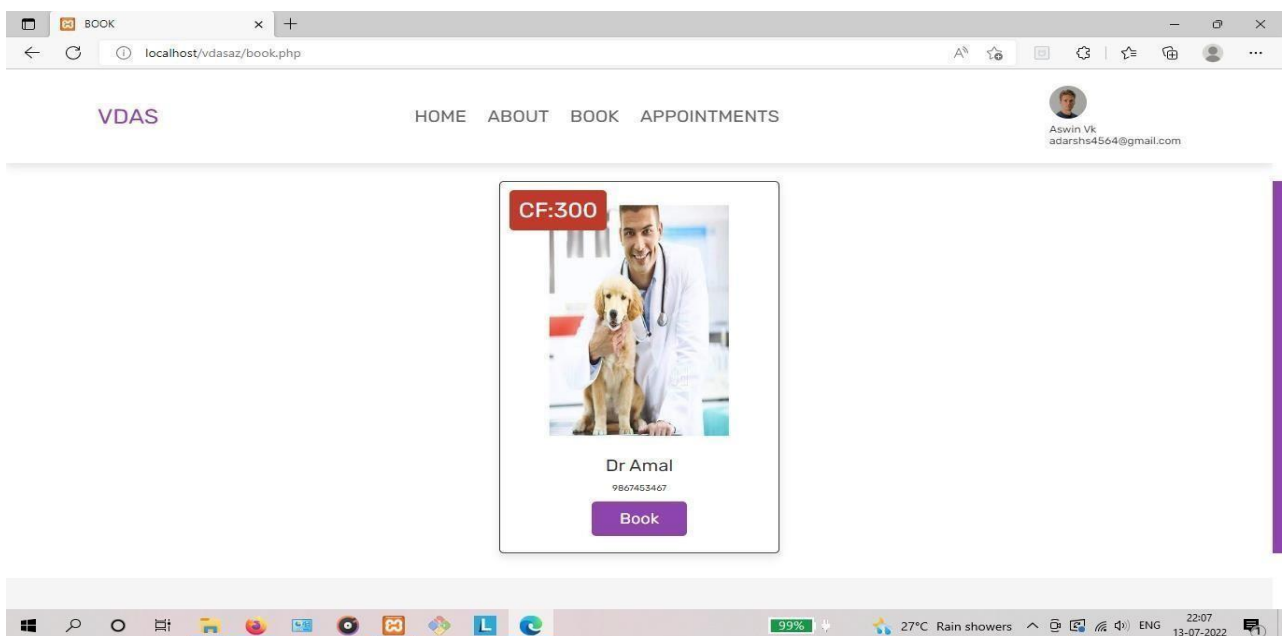
## 9.1 Screen Shots

### USER PAGES

#### User Home page




#### Book





## Appointments

**VDAS**HOME ABOUT BOOK APPOINTMENTS

  
Aswin V  
adarshs4564@gmail.com

Placed on : 13-Jul-2022

Pettytype : goat

Appointment Date : 2022-07-14

Appointment Time : 11:08:10.000000

Status :

Consultancy Fee : 300

Treatment status : 0

Payment status

Download


Pay Now


Cancel

## Adopt Pets


ORDER

localhost/vdasaz/shop.php

HOME ABOUT BOOK APPOINTMENTS ADOPT PETS


  
Aswin V  
adarshs4564@gmail.com

### PETS




Dog

Order




Goat

Order



Cat

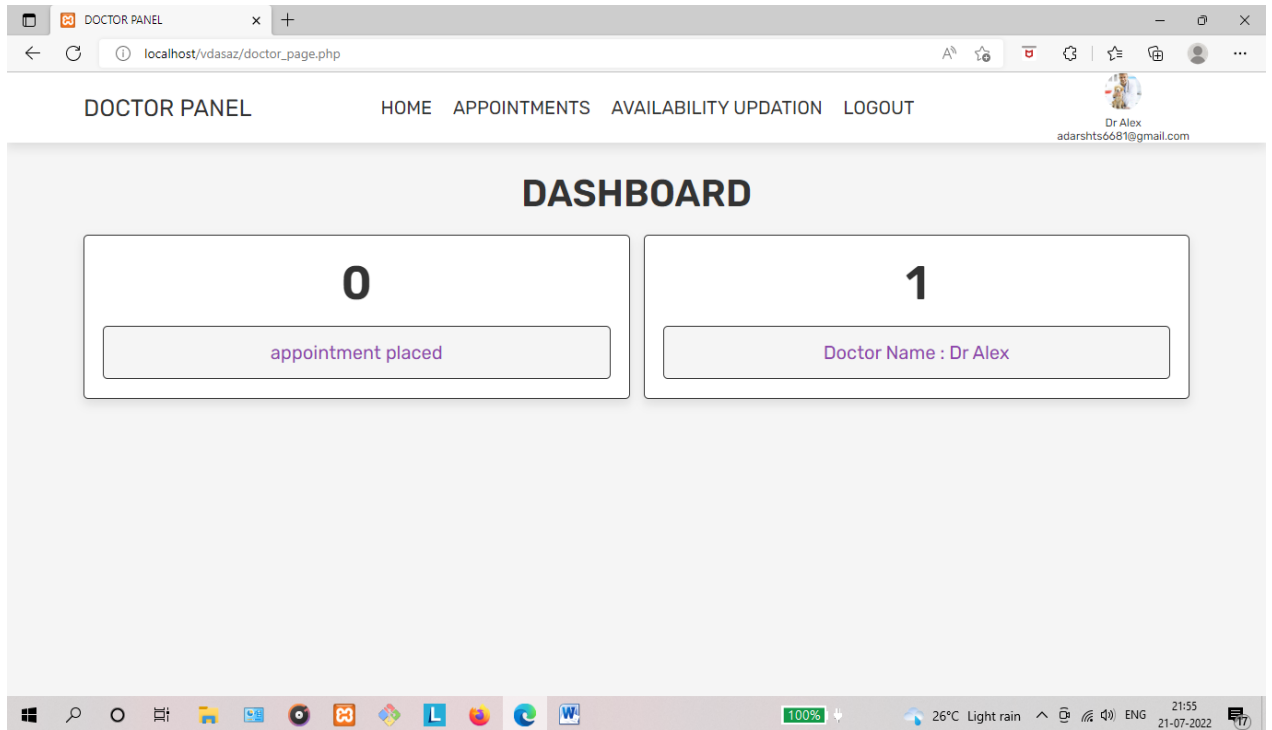
Order



Amal Jyothi College of Engineering, Kanjirappally

Department of Computer Applications

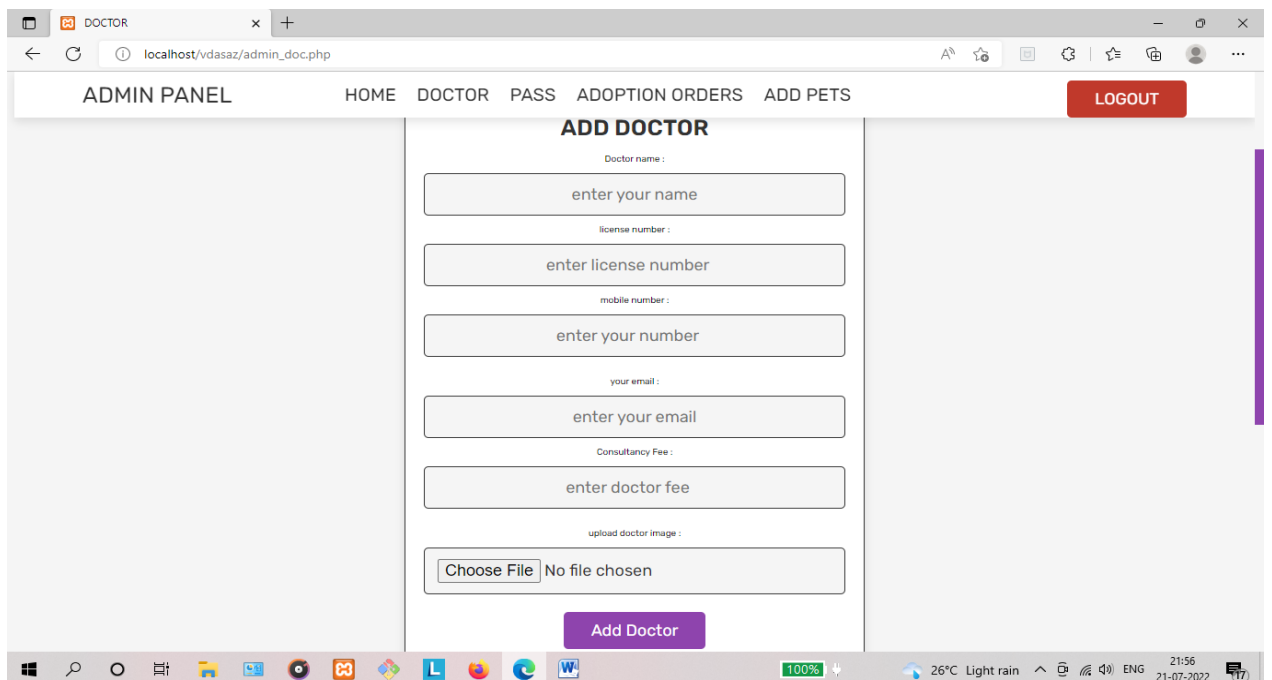
## Doctor Pages



The screenshot shows a web browser window with the title "DOCTOR PANEL". The address bar shows "localhost/vdasaz/doctor\_page.php". The navigation bar includes "DOCTOR PANEL", "HOME", "APPOINTMENTS", "AVAILABILITY UPDATION", and "LOGOUT". A user profile for "Dr Alex" with email "adarshs6681@gmail.com" is shown in the top right. The main content area is titled "DASHBOARD" and contains two cards. The first card displays a large number "0" and a button labeled "appointment placed". The second card displays a large number "1" and a button labeled "Doctor Name : Dr Alex". The Windows taskbar at the bottom shows various application icons, a 100% battery level, and system information: "26°C Light rain", "ENG", and the date "21-07-2022".

## Admin pages

### Doctor Add



The screenshot shows a web browser window with the title "DOCTOR". The address bar shows "localhost/vdasaz/admin\_doc.php". The navigation bar includes "ADMIN PANEL", "HOME", "DOCTOR", "PASS", "ADOPTION ORDERS", "ADD PETS", and a "LOGOUT" button. The main content area is titled "ADD DOCTOR" and contains a form with the following fields: "Doctor name :", "license number :", "mobile number :", "your email :", "Consultancy Fee :", and "upload doctor image :". Each field has a corresponding input box. The "upload doctor image :" field includes a "Choose File" button and the text "No file chosen". A purple "Add Doctor" button is located at the bottom of the form. The Windows taskbar at the bottom shows various application icons, a 100% battery level, and system information: "26°C Light rain", "ENG", and the date "21-07-2022".

## Pet Add




ADMIN PANEL HOME DOCTOR PASS ADOPTION ORDERS ADD PETS LOGOUT

### ADD PETS

Pet name :  
enter your name

upload pet image :  
Choose File No file chosen

Add Pet



21:58  
21-07-2022



| ●

|

| •

• |

| •

12/11/2015

11/11/2015

● |

● |

● |

● |

● |

● |

12/11/2015