

PETKART

Project Report Submitted by

UPAS NATH

Reg. No.:AJC21MCA-2107

In Partial fulfillment for the Award of the Degree Of

MASTER OF COMPUTER APPLICATIONS

(MCA TWO YEAR)

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]

2022-2023

DEPARTMENT OF COMPUTER APPLICATIONS
AMAL JYOTHI COLLEGE OF ENGINEERING
KANJIRAPPALLY



CERTIFICATE

This is to certify that the Project report, “**PETKART**” is the bonafide work of **UPAS NATH (Reg. No: AJC21MCA-2107)** 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 2022-23.

Mr. AJITH G.S

Internal Guide

Ms. Meera Rose Mathew

Coordinator

Rev. Fr. Dr. Rubin Thottupurathu Jose

Head of the Department

DECLARATION

I hereby declare that the project report “**PETKART**” is a bonafide 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 2022-2023.

Date: 26/11/2022

UPAS NATH

KANJIRAPPALLY

Reg: AJC21MCA-2107

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 coordinator **Ms. Meera Rose Mathew** for her valuable suggestions and for overwhelming concern and guidance from the beginning to the end of the project. I would also express sincere gratitude to my guide **Mr. AJITH G.S** 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.

UPAS NATH

ABSTRACT

Many peoples love and care their pets. Pets like, cat, rabbit, birds, dogs, etc. Those who love this kind of pets they don't know like where to buy and how to give care for pets, what kind of foods needed, vaccination period, suitable cave etc. In this project we discuss about an website application which help them to buy or sell these things. At present pet shops and accessories shops are available, but users cannot sell or buy above mentioned things directly. Users can buy pets or foods from shops only.to overcome this situation we discuss this project. Proposed system overcome the issues of existing system.in this application anyone can register and login. After login they can decide buy or sell. If you choose buy option you can choose category and it will displays all items available in that category. You can buy or book that item. Items like, pets, foods, caves etc. you can also know about pets breeding time, vaccination, necessary food details.

If you choose seller option then you can add your item in this application. Also you can see orders and payments. Reviews and ratings. Add complaints to admin about someone.

CONTENT

SL. NO	TOPIC	PAGE NO
1	INTRODUCTION	1
1.1	PROJECT OVERVIEW	2
1.2	PROJECT SPECIFICATION	3
2	SYSTEM STUDY	5
2.1	INTRODUCTION	6
2.2	EXISTING SYSTEM	6
2.3	DRAWBACKS OF EXISTING SYSTEM	6
2.4	PROPOSED SYSTEM	6
2.5	ADVANTAGES OF PROPOSED SYSTEM	6
3	REQUIREMENT ANALYSIS	7
3.1	FEASIBILITY STUDY	8
3.1.1	ECONOMICAL FEASIBILITY	8
3.1.2	TECHNICAL FEASIBILITY	8
3.1.3	BEHAVIORAL FEASIBILITY	9
3.1.4	FEASIBILITY STUDY QUESTIONNAIRE	9
3.2	SYSTEM SPECIFICATION	9
3.2.1	HARDWARE SPECIFICATION	9
3.2.2	SOFTWARE SPECIFICATION	9
3.3	SOFTWARE DESCRIPTION	10
3.3.1	PHP	10
3.3.2	MYSQL	10-12
4	SYSTEM DESIGN	13
4.1	INTRODUCTION	14
4.2	UML DIAGRAM	14
4.2.1	USE CASE DIAGRAM	15-16
4.2.2	SEQUENCE DIAGRAM	17-18
4.2.3	STATE CHART DIAGRAM	19
4.2.4	ACTIVITY DIAGRAM	20
4.2.5	CLASS DIAGRAM	21
4.2.6	OBJECT DIAGRAM	22
4.2.7	COMPONENT DIAGRAM	23

4.2.8	DEPLOYMENT DIAGRAM	23
4.3	USER INTERFACE DESIGN USING FIGMA	24
4.4	DATA BASE DESIGN	25
4.4.1	RELATIONAL DATABASE MANAGEMENT SYSTEM	26
4.4.2	NORMALIZATION	27-28
4.4.3	SANITIZATION	28
4.4.4	INDEXING	28
4.5	TABLE DESIGN	28-32
5	SYSTEM TESTING	33
5.1	INTRODUCTION	34
5.2	TEST PLAN	35
5.2.1	UNIT TESTING	35
5.2.2	INTEGRATION TESTING	36
5.2.3	VALIDATION TESTING	36
5.2.4	USER ACCEPTANCE TESTING	37
5.2.5	AUTOMATION TESTING	37
5.2.6	SELENIUM TESTING	37
6	IMPLEMENTATION	38
6.1	INTRODUCTION	39
6.2	IMPLEMENTATION PROCEDURE	39
6.2.1	USER TRAINING	40
6.2.2	TRAINING ON APPLICATION SOFTWARE	40
6.2.3	SYSTEM MAINTENANCE	40
7	CONCLUSION & FUTURE SCOPE	41
7.1	CONCLUSION	42
7.2	FUTURE SCOPE	42
8	BIBLIOGRAPHY	43-44
9	APPENDIX	45
9.1	SAMPLE CODE	46-55
9.2	SCREEN SHOTS	56-58

List of Abbreviation

IDE - Integrated Development Environment

HTML - Hyper Text Markup Language.

CSS - Cascading Style Sheet

SQL - Structured Query Language

UML - Unified Modeling Language

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

Here we done a website 'PETKART' that is a pet management system useful for the purpose of informing the people of different places about pets availability. Currently there are no such applications in our locality. So introducing such an application will be a fruitful thing for buyers and sellers to communicate about the need and availability of pets without any information loss. All the information relating to that concerned pets will also be mentioned in the post given by the seller . It is highly secured by allotting a unique password for all users in the start-up of the application. Only users need to be registered. An admin account is already settled for approving posts by checking its correctness and reliability. As website. We uses MySQL for creating database and storing details.

Each phases that we encountered during the development of this application has been mentioned here very accurately. Our problem domain was to create an online platform for buyers and sellers to communicate . The admin of the system will already be registered during the implementation of the website and the users are manually registered in the system. After the registration of users, each user's are provided an email and a unique password for login. The users can sign up their account by entering the details such as name, email, password. The user profile can be customized by changing the profile picture of the user, name, email, phone, etc..

1.2 PROJECT SPECIFICATION

At present pet shops and accessories shops are available, but users cannot sell or buy above mentioned things directly. Users can buy pets or foods from shops only. To overcome this situation we discuss this project. Proposed system overcomes the issues of existing system. In this application anyone can register and login. After login they can decide buy or sell. If you choose buy option you can choose category and it will display all items available in that category. You can buy or book that item. Items like, pets, foods, cages etc. you can also know about pets breeding time, vaccination, necessary food details.

If you choose seller option then you can add your item in this application. Also you can see orders and payments. Reviews and ratings. Add complaints to admin about someone.

The major modules included in the system are,

- Admin.
- Users
- Trainer

ADMIN

Admin module is the overall controlling module or the one important module of this system. This module is provided with a username and password as login credentials. The major feature of this module is it could control users views and delivery person. This module could view users, orders of the products, reviews and ratings given by the customers. Admin module can also see the complaints registered by user and could take actions upon it. Admin module adds the categories and details like breeding and vaccination time, pet foods etc. Admin adds the delivery person and assigns orders to the responsible delivery person. Admin can also view order status updated by the delivery person ..

USER

User module is another important module which only goes for a registration process. At the time of the registration the details such as name, contact info, bank account details are taken from the user. There will be a username and password for each user as login credentials. They can also choose buy or sell option from the home page itself. Users can add products in categories as pets

or pet food and can also edit or delete those products. Users can also view the available products in dashboard. Users are responsible for updating selling status and the complaints. users can know about pet foods, vaccination details etc. The payments can be done as online transactions, by the details provided at the registration time. For this feature the system ensures high security for storing the user information.

TRAINER

Trainer is a module in which it consists of an option to login as a trainer, allow him to post videos, photos and other information that are relevant to the process of caring pets. If these videos got captured any of the user's attention, he will be booked to train their pets by booking made by user. After booking, a specific amount (or training fee) will be presented with appropriate format to the panel of user. Having finished each booking made by a user, he or she can message him over a chat box. During this tiny period of chat, they both will arrive at a consistent decision concern moving forward to train the pets under trainer's consideration.

CHAPTER 2

SYSTEM STUDY

2.1 INTRODUCTION

This Pet Shop Online Shopping System using php was designed to manage the pet information such to produce the pet breed and the vaccine schedules. It is also performed in systematic through its function requirement. The management of pet shop includes the services offers such as cleaning, scissoring, bathing and blow-drying.

First, the pet must be registered before having any services. Any piece of information about the pets must be correct and organized to avoid any problem to be in countered like the bad feedback from the customer. The purpose of this system is to transact and deals with a nice and easier way by simply gathering information from the customer. It also helps to every individual by searching for the information needed. project image.

2.2.1 NATURAL SYSTEM STUDIED

At present pet shops and accessories shops are available, but users cannot sell or buy above mentioned things directly. Users can buy pets or foods from shops only. to overcome this situation we discuss this project.

2.2 DRAWBACKS OF EXISTING SYSTEM

- Not easy to access
- Time consuming process
- Don't Know the reviews and rating of each shop

2.2.2 DESIGNED SYSTEM STUDIED

Proposed system overcome the issues of existing system. in this application anyone can register and login. After login they can decide buy or sell. If you choose buy option you can choose category and it will displays all items available in that category. You can buy or book that item. Items like, pets, foods, caves etc. you can also know about pets breeding time, vaccination, necessary food details.

If you choose seller option then you can add your item in this application. Also you can see orders and payments. Reviews and ratings. Add complaints to admin about someone.

ADVANTAGES OF PROPOSED SYSTEM

- Less manual effort
- Time efficient system
- Interaction becomes more active through the system.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 FEASIBILITY STUDY

Planning, organizing, and managing resources to ensure the achievement of particular project goals and objectives is the process of project management. A feasibility study is a preliminary examination of a prospective project or end to determine its merits and viability. A feasibility study aims to provide an objective assessment of the technical, economic, financial, legal, and environmental elements of a proposed project. The information can then be used by decision-makers to decide whether to proceed with the project or not. The findings of the feasibility study can also be used to develop a practical project plan and budget. It cannot be simple to determine whether or not a proposed project is worthwhile pursuing without a feasibility study. The document provides the feasibility of the project that is being designed and lists. Various areas that were considered very carefully during the feasibility study of this project such as Technical, Economic and Operational feasibility. The following are its features:

3.1.1 Economical Feasibility

Cost and benefit analyses are required to support the developing system. criteria to make sure that focus is on the project that will yield the best results and return the earliest. The price that would be involved in developing a new system is one of the variables. Some significant financial queries raised during the initial investigation include the following:

- The costs conduct a full system investigation?
 - The proposed system is developed as part of project work, there is no manual cost to spend for the proposed system.
- The cost of the hardware and software?
 - Also all the resources are already available.

3.1.2 Technical Feasibility

The system needs to be assessed first from a technical standpoint. The outline design of the system requirement in terms of input, output, programs, and procedures must serve as the foundation for the assessment of this feasibility. After determining an outline investigation must continue to identify the necessary equipment kind. Once the system has been designed, there are several ways to run it.

- Is the project feasible within the limits of current technology.
 - YES

- Technical issues raised during the investigation are:
 - NOTHING
- Can the technology be easily applied to current problems?
 - YES
- Does the technology have the capacity to handle the solution?
 - YES
 -

3.1.3 Behavioral Feasibility

The proposed system includes the following questions:

- Is there sufficient support for the users
 - YES
- Will the proposed system cause harm?
 - NO

The project would be beneficial because it satisfies the objectives when developed and installed. All behavioral aspects are considered carefully and conclude that the project is behaviorally feasible

3.1 SYSTEM SPECIFICATION

3.2.1 Hardware Specification

Processor - Intel core i5

RAM - 8 GB

Hard disk - 1 TB HDD

3.2.2 Software Specification

Front End - HTML, CSS

Backend - MYSQL

Client on PC - Windows 7 and above.

Technologies used - JS, HTML5, J Query, PHP, CSS

3.3 SOFTWARE DESCRIPTION

3.3.1 PHP

PHP is a server side scripting language designed for web development but also used as a general purpose programming language. PHP is now installed on more than 244 million websites and 2.1 million web servers. Originally created by Rasmus Ledorf in 1995, the reference implementation of PHP is now produced by the PHP group. While PHP originally stood for personal Home page ,it now stands for PHP:HypertextPreprocessor, a recursive acronym.PHP code is interpreted by a web server with a PHP processor module which generates the resulting web page.PHP commands can be embedded directly into a HTML source document rather than calling an external file to process data. It has also evolved to include a command-line interface capability and can be used in standalone incompatible with the GNU General Public License (GPL) due to restrictions on the usage of the term PHP.PHP can be deployed on most web servers and also as a standalone shell on almost every operating system and platform, free of charge.

3.3.2 MySQL

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by OracleCorporation. The MySQLWeb site providesthe latest information about MySQL software.

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

A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

- **MySQL databases are relational.**

A relational database stores data in separate tables rather than putting all the data in one big storeroom. The database structures are organized into physical files optimized for speed. The logical model, with objects such as databases, tables, views, rows, and columns, offers a flexible programming environment. You setup rules governing the relationships between different data fields, such as one-to-one, one-to-many, unique, required or optional, and “pointers” between different tables. The database enforces these rules, so that with a well-designed database, your application never sees inconsistent, duplicate, orphan, out-of-date, or missing data. The SQL part of “MySQL” stands for “Structured Query Language”. SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax. SQL is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual, “SQL92” refers to the standard released in 1992, “SQL: 1999” refers to the standard released in 1999, and “SQL: 2003” refers to the current version of the standard. We use the phrase “the SQL standard” to mean the current version of the SQL Standard at any time.

- **MySQL software is Open Source.**

Open Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL software uses the GPL (GNU General Public License), to define what you may and may not do with the software in different situations. If you feel uncomfortable with the GPL or need to embed MySQL code into a commercial application, you can buy a commercially licensed version from us. See the MySQL Licensing Overview for more information.

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

If that is what you are looking for, you should give it a try. MySQL Server can run comfortably on a desktop or laptop, alongside your other applications, web servers, and so on, requiring little or no attention. If you dedicate an entire machine to MySQL, you can adjust the settings to take advantage of all the memory, CPU power, and I/O capacity available.

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

The MySQL Database Software is a client/server system that consists of a multithreaded SQL server that supports different backends, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs). We also provide MySQL Server as an embedded multi-threaded library that you can link into your application to get a smaller, faster, easier-to-manage standalone product.

CHAPTER 4

SYSTEM DESIGN

4.1 INTRODUCTION

Design is the first step into the development phase for any engineered product or system. Design is a creative process. A good design is the key to effective system. The term “design” is defined as “the process of applying various techniques and principles for the purpose of defining a process or a system in sufficient detail to permit its physical realization”. It may be defined as a process of applying various techniques and principles for the purpose of defining a device, a process or a system in sufficient detail to permit its physical realization. Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm that is used. The system design develops the architectural detail required to build a system or product. As in the case of any systematic approach, this software too has undergone the best possible design phase fine tuning all efficiency, performance and accuracy levels. The design phase is a transition from a user oriented document to a document to the programmers or database personnel. System design goes through two phases of development: Logical and Physical Design.

4.2 UML DIAGRAM

UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. UML was created by the Object Management Group (OMG) and UML 1.0 specification draft was proposed to the OMG in January 1997. UML stands for Unified Modeling Language. UML is different from the other common programming languages such as C++, Java, COBOL, etc. UML is a pictorial language used to make software blueprints. UML can be described as a general purpose visual modeling language to visualize, specify, construct, and document software system. Although UML is generally used to model software systems, it is not limited within this boundary. It is also used to model non-software systems as well. For example, the process flow in a manufacturing unit, etc. UML is not a programming language but tools can be used to generate code in various languages using UML diagrams. UML has a direct relation with 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
- State chart diagram
- Deployment diagram
- Component diagram

4.2.1 USE CASE DIAGRAM

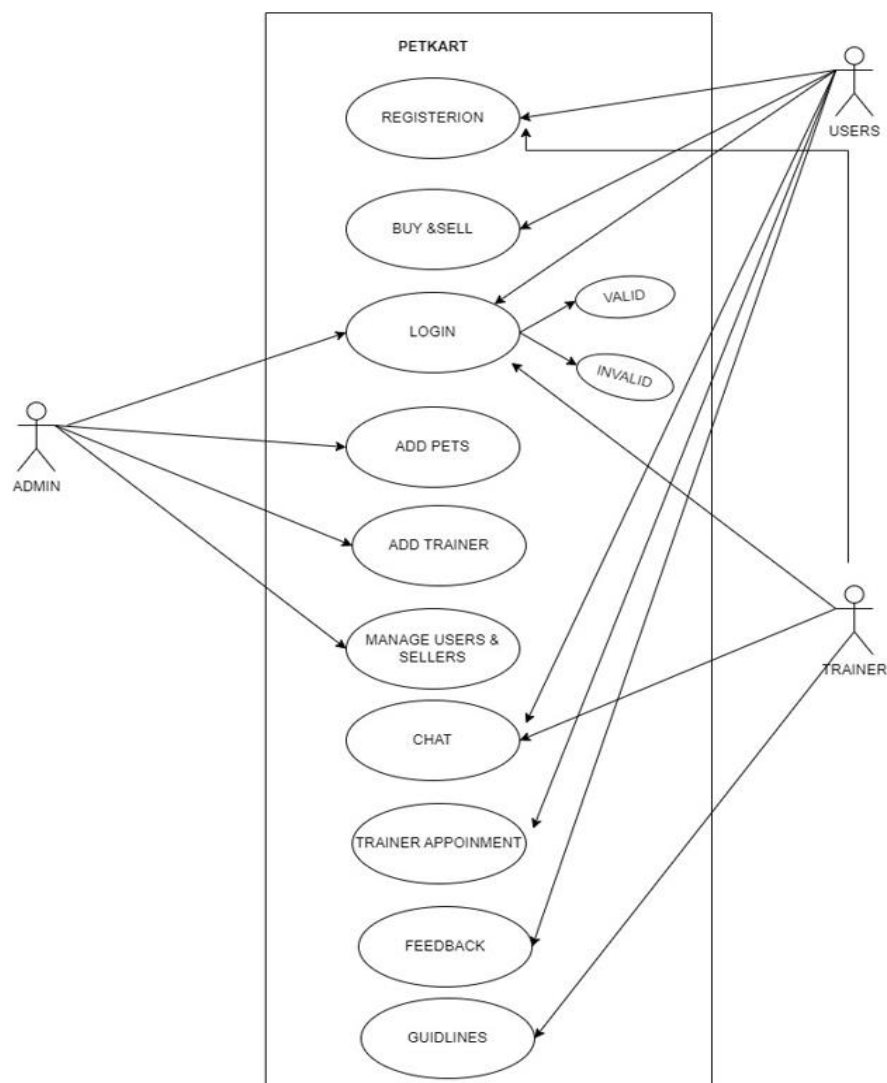
A use case diagram is a graphic depiction of the interactions among the elements of a system. A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. In this context, the term "system" refers to something being developed or operated, such as a mail-order product sales and service Web site. Use case diagrams are employed in UML (Unified Modeling Language), a standard notation for the modeling of real-world objects and systems. System objectives can include planning overall requirements, validating a hardware design, testing and debugging a software product under development, creating an online help reference, or performing a consumer-service-oriented task. For example, use cases in a product sales environment would include item ordering, catalog updating, payment processing, and customer relations. A use case diagram contains four components.

- The boundary, which defines the system of interest in relation to the world around it.
- 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 relationships between and among the actors and the use cases.

Use case diagrams are drawn to capture the functional requirements of a system. After identifying the above items, we have to use the following guidelines to draw an efficient use case diagram

- The name of a use case is very important. The name should be chosen in such a way so that it can identify the functionalities performed.
- Give a suitable name for actors.
- Show relationships and dependencies clearly in the diagram.
- Do not try to include all types of relationships, as the main purpose of the diagram is to identify the requirements.
- Use notes whenever required to clarify some important points.

DIAGRAM



4.2.1 SEQUENCE DIAGRAM

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

Sequence Diagram Notations

- i. **Actors** – An actor in a UML diagram represents a type of role where it interacts with the system and its objects. It is important to note here that an actor is always outside the scope of the system we aim to model using the UML diagram. We use actors to depict various roles including human users and other external subjects. We represent an actor in a UML diagram using a stick person notation. We can have multiple actors in a sequence diagram.
- ii. **Lifelines** – A lifeline is a named element which depicts an individual participant in a sequence diagram. So basically each instance in a sequence diagram is represented by a lifeline. Lifeline elements are located at the top in a sequence diagram.
- iii. **Messages** – Communication between objects is depicted using messages. The messages appear in a sequential order on the lifeline. We represent messages using arrows. Lifelines and messages form the core of a sequence diagram.

Messages can be broadly classified into the following categories:

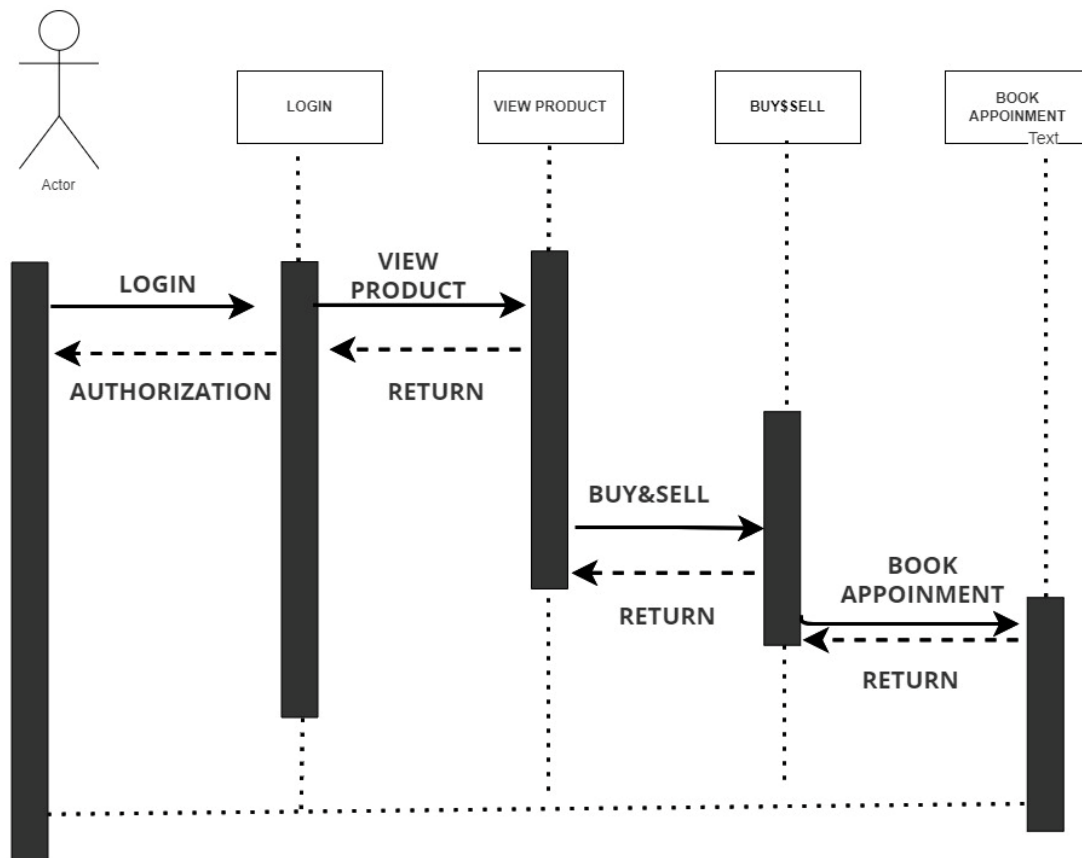
- Synchronous messages
- Asynchronous Messages
- Create message
- Delete Message
- Self-Message
- Reply Message
- Found Message
- Lost Message
-

- iv. **Guards** – To model conditions we use guards in UML. They are used when we need to restrict the flow of messages on the pretext of a condition being met. Guards play an important role in letting software developers know the constraints attached to a system or a particular process.

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.
- Visualize how messages and tasks move between objects or components in a system

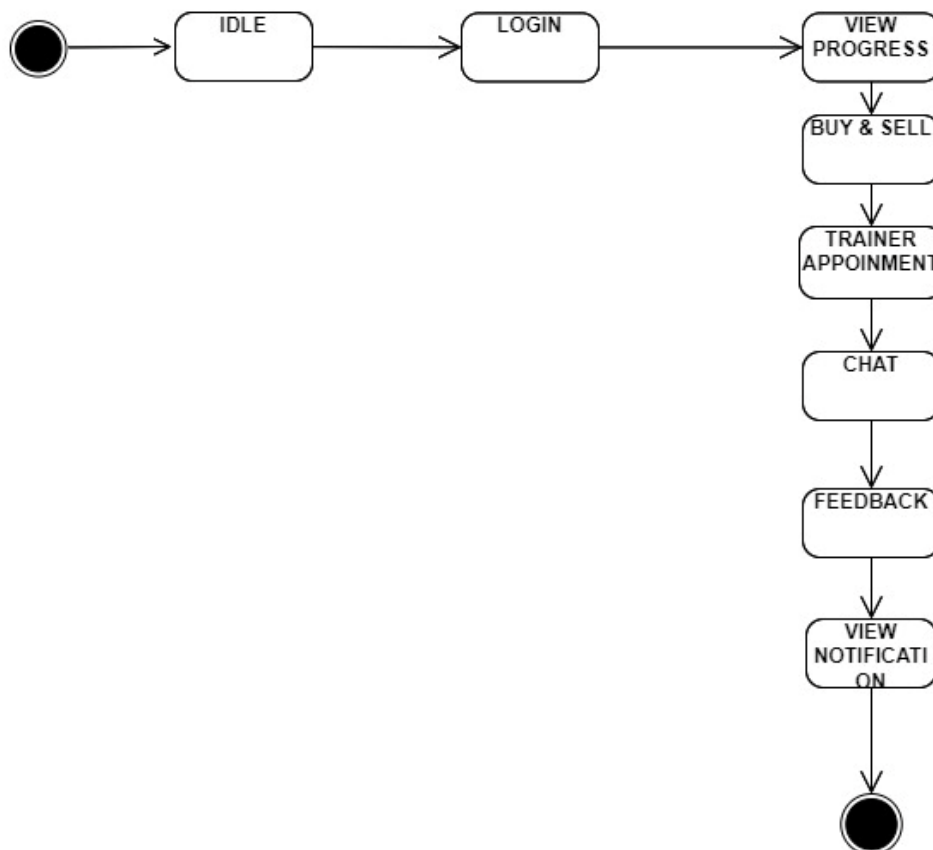
DIAGRAM



4.2.2 State Chart Diagram

A particular form of diagram used in computer science and related subjects to explain how systems behave is called a state diagram. State diagrams call for the system being represented to consist of a finite number of states; occasionally, this is the case, and other times, it's only an acceptable abstraction. State diagrams come in a variety of shapes and sizes, each with a distinct meaning. State diagrams are there to provide a system's behaviour an abstract explanation. In order to evaluate and demonstrate this behaviour, a sequence of events that could occur in one or more hypothetical states is employed. "Each diagram typically depicts objects of a single class and monitor the different states of its objects across the system," according to this.

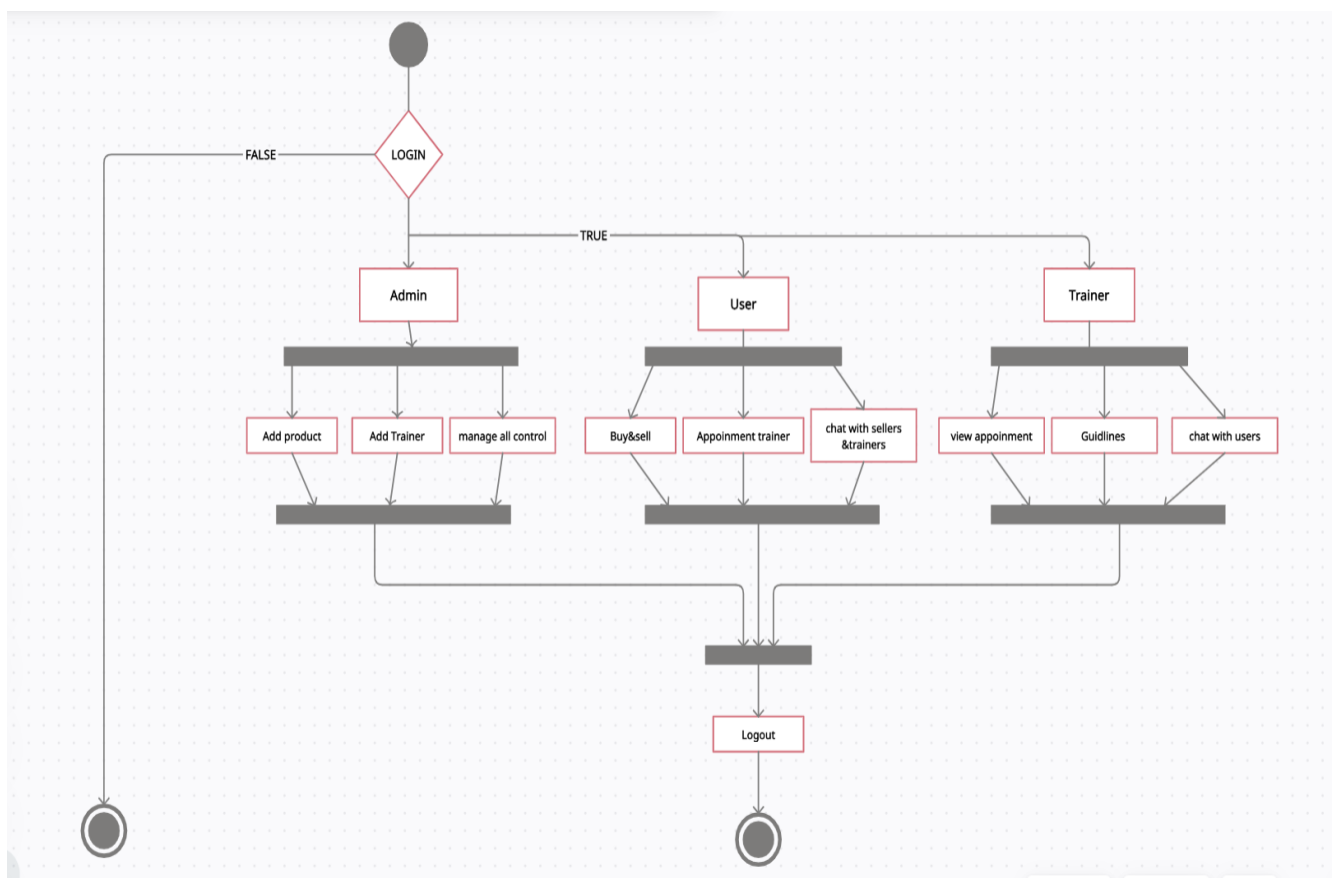
DIAGRAM



4.2.2 Activity Diagram

Activity diagrams depict how different levels of abstraction of activities are linked to provide a service. Typically, an event should be completed by some activities, particularly when the activity is intended to do multiple separate goals that need coordination. Another typical requirement is how the events in a single use case interact with one another, particularly in use cases where operations may overlap and require coordination. It may also be used to show how a collection of interrelated use cases interacts to reflect business operations.

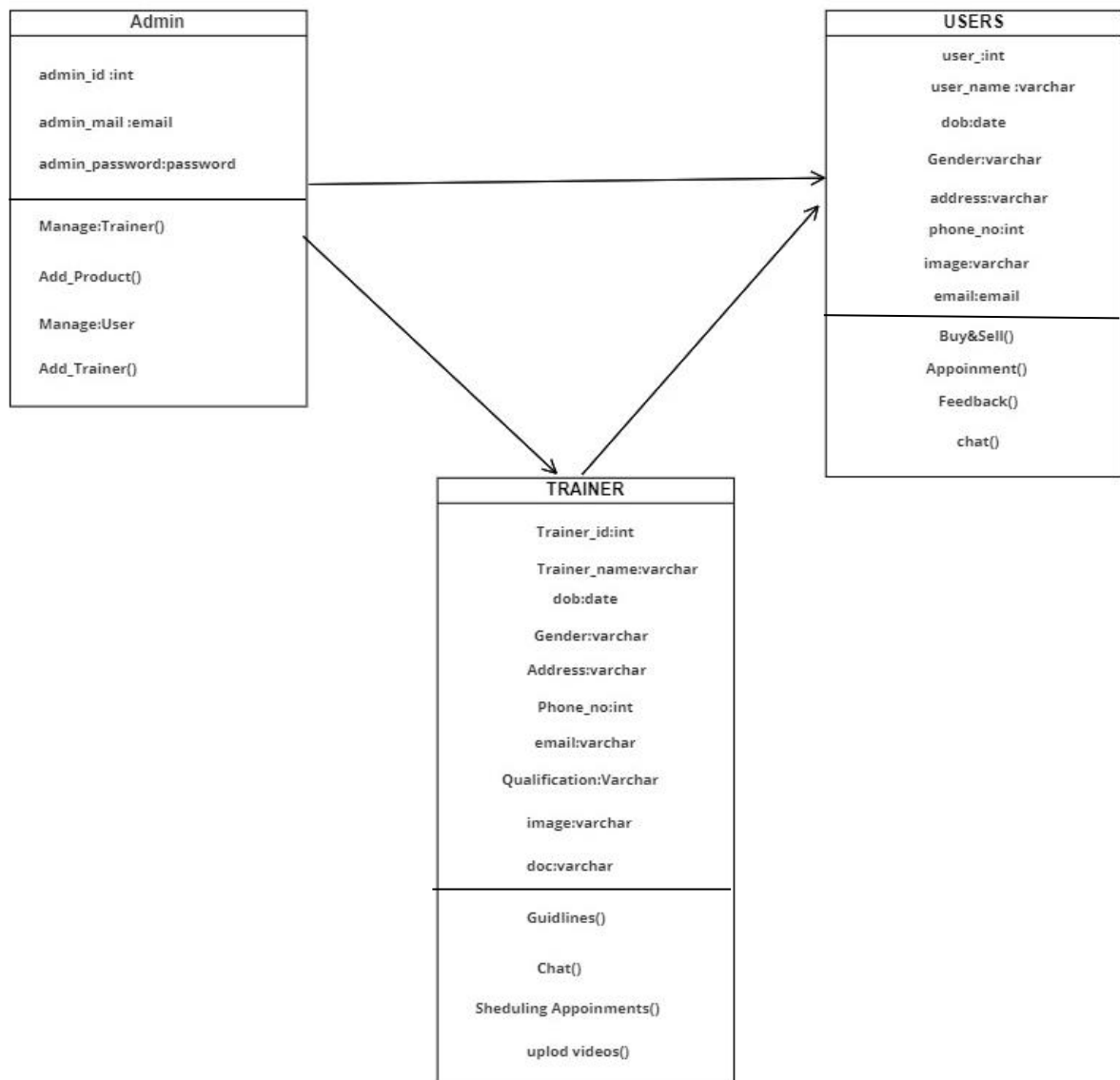
DIAGRAM



4.2.3 Class Diagram

Class diagram is a static diagram. It represents the static view of the application. Class diagrams are useful for visualising, describing, and documenting various system components as well as for writing executable code for software applications. A class diagram describes the constraints imposed on the system together with the properties and operations of a class. The only UML diagrams that can be directly converted into objectoriented languages are class diagrams, which are extensively utilised in the designing of object-oriented systems. An assortment of classes, interfaces, affiliations, partnerships, and limitations are displayed in a class diagram. It also goes by the name "structural diagram."

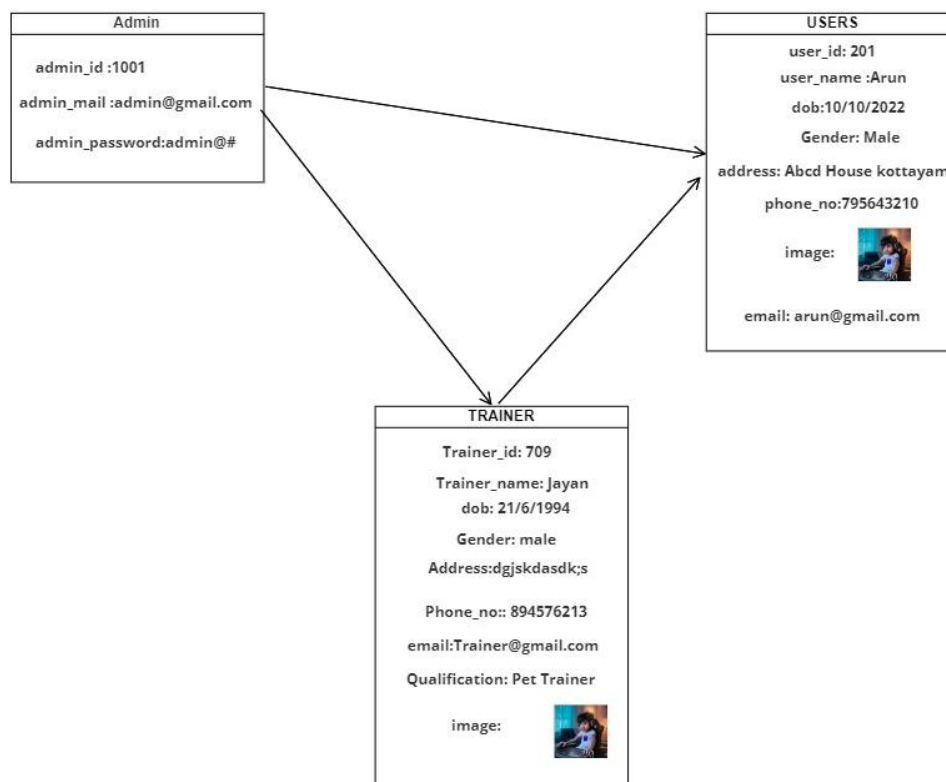
DIAGRAM



4.2.4 Object Diagram

Class diagrams are necessary before object diagrams can be created since they are the ancestor of object diagrams. An object diagram represents a particular instance of a class diagram. The underlying concepts used in class and object diagrams are the same. Object diagrams may also describe the static view of a system, although this static view only depicts a current state of the system. Object diagrams are used to show links between a set of things.

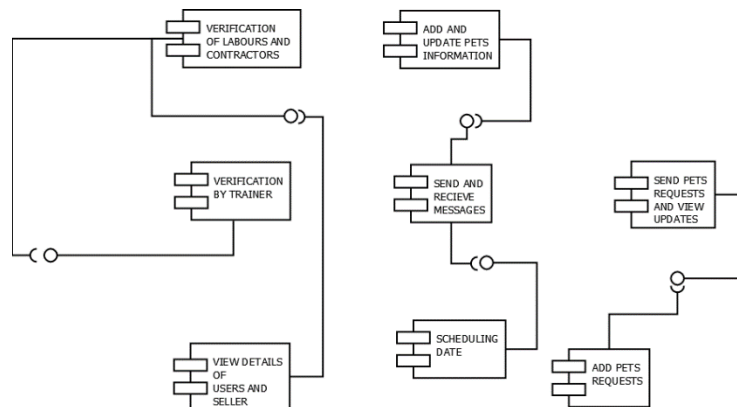
DIAGRAM



4.2.5 Component Diagram

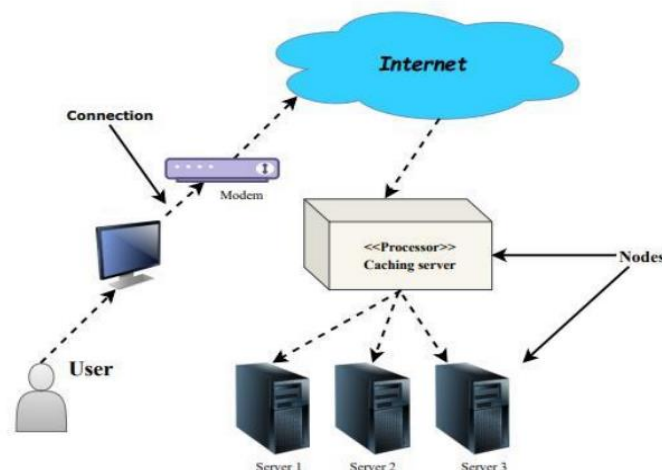
Component diagrams have different behaviour's and personalities. The physical parts of the system are represented using component diagrams. Executables, libraries, files, documents, and other items that are physically present in a node are just a few examples. Component diagrams are used to show how the components of a system are connected and arranged. These diagrams may also be used to construct systems that can be run.

DIAGRAM



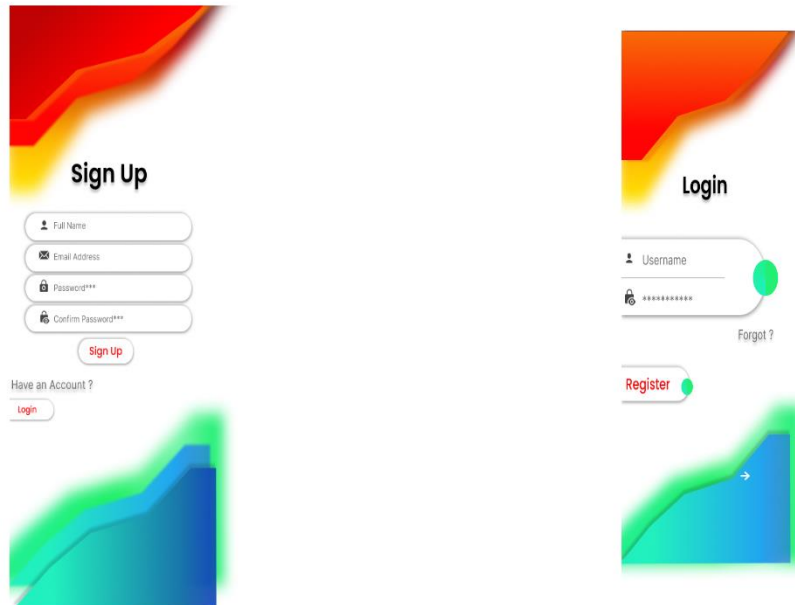
4.2.8 Deployment Diagram

An execution architecture of a system, containing nodes like hardware or software execution environments, and the middleware linking them, is shown in a deployment diagram, a form of UML diagram. Typically, deployment diagrams are used to represent the actual hardware and software of a system. By using it, you can comprehend how the hardware will physically deliver the system. In contrast to other UML diagram types, which primarily depict the logical components of a system, deployment diagrams assist describe the hardware structure of a system.

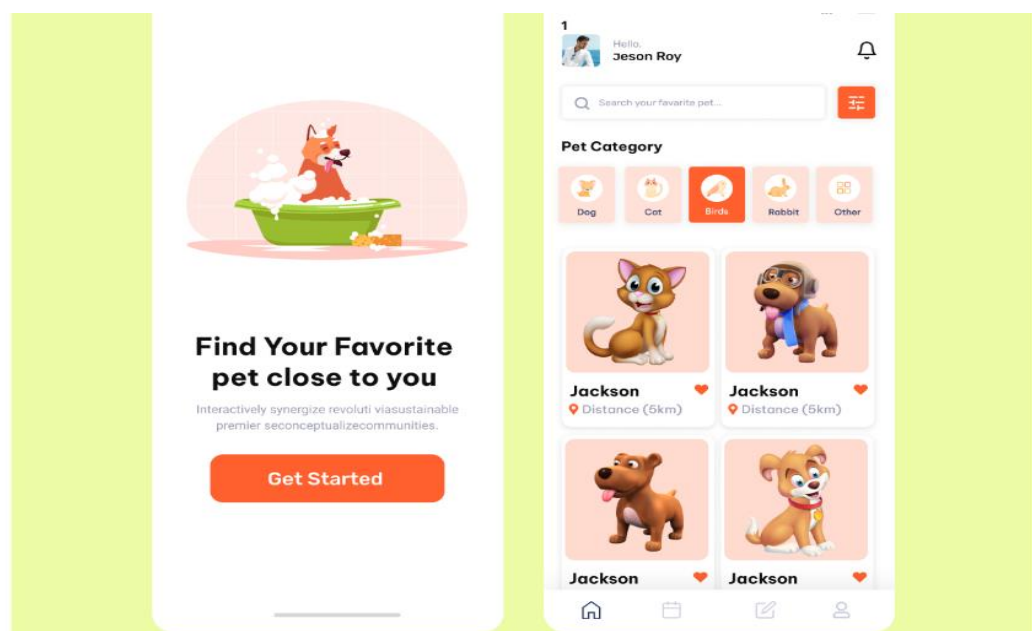


4.3 USER INTERFACE DESIGN USING FIGMA

Form Name: sign_up & sign_in



Form Name: Home page



4.3 DATABASE DESIGN

A database is an organized mechanism that has the capability of storing information through which a user can retrieve stored information in an effective and efficient manner. The data is the purpose of any database and must be protected.

The database design is a two level process. In the first step, user requirements are gathered together and a database is designed which will meet these requirements as clearly as possible. This step is called Information Level Design and it is taken independent of any individual DBMS.

In the second step, this Information level design is transferred into a design for the specific DBMS that will be used to implement the system in question. This step is called Physical Level Design, concerned with the characteristics of the specific DBMS that will be used. A database design runs parallel with the system design. The organization of the data in the database is aimed to achieve the following two major objectives.

- Data Integrity
- Data independence

A database is an organized mechanism that has the capability of storing information through which a user can retrieve stored information in an effective and efficient manner. The data is the purpose of any database and must be protected.

The database design is a two level process. In the first step, user requirements are gathered together and a database is designed which will meet these requirements as clearly as possible. This step is called Information Level Design and it is taken independent of any individual DBMS.

In the second step, this Information level design is transferred into a design for the specific DBMS that will be used to implement the system in question. This step is called Physical Level Design, concerned with the characteristics of the specific DBMS that will be used. A database design runs parallel with the system design. The organization of the data in the database is aimed to achieve the following two major objectives.

- Data Integrity
- Data independence

4.4.1 Relational Database Management System (RDBMS)

A relational model represents the database as a collection of relations. Each relation resembles a table of values or file of records. In formal relational model terminology, a row is called a tuple, a column header is called an attribute and the table is called a relation. A relational database consists of a collection of tables, each of which is assigned a unique name. A row in a table represents a set of related values.

Relations, Domains & Attributes

A table is a relation. The rows in a table are called tuples. A tuple is an ordered set of n elements. Columns are referred to as attributes. Relationships have been set between every table in the database. This ensures both Referential and Entity Relationship Integrity. A domain D is a set of atomic values. A common method of specifying a domain is to specify a data type from which the data values forming the domain are drawn. It is also useful to specify a name for the domain to help 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.

4.4.2 Normalization

Data are grouped together in the simplest way so that later changes can be made with minimum impact on data structures. Normalization is formal process of data structures in manners that eliminates redundancy and promotes integrity. Normalization is a technique of separating redundant fields and breaking up a large table into a smaller one. It is also used to avoid insertion, deletion, and updating anomalies. Normal form in data modelling use two concepts, keys and relationships. A key uniquely identifies a row in a table. There are two types of keys, primary key and foreign key. A primary key is an element or a combination of elements in a table whose purpose is to identify records from the same table. A foreign key is a column in a table that uniquely identifies record from a different table. All the tables have been normalized up to the third normal form. As the name implies, it denotes putting things in the normal form. The application developer via normalization tries to achieve a sensible organization of data into proper tables and columns and where names can be easily correlated to the data by the user. Normalization eliminates repeating groups at data and thereby avoids data redundancy which proves to be a great burden on the 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

The First Normal Form states that the domain of an attribute must include only atomic values and that the value of any attribute in a tuple must be a single value from the domain of that attribute. In other words 1NF disallows “relations within relations” or “relations as attribute values within tuples”. The only attribute values permitted by 1NF are single atomic or indivisible values. The first step is to put the data into First Normal Form. This can be done by moving data into separate tables where the data is of similar type in each table. Each table is given a Primary Key or Foreign Key as per requirement of the project. In this we form new relations for each non-atomic attribute or nested relation. This eliminated repeating groups of data. A relation is said to be in first normal form if only if it satisfies the constraints that contain the primary key only.

Second Normal Form

According to Second Normal Form, for relations where primary key contains multiple attributes, no non-key attribute should be functionally dependent on a part of the primary key. In this we decompose and setup a new relation for each partial key with its dependent attributes. Make sure to keep a relation with the original primary key and any attributes that are fully functionally dependent on it. This step helps in taking out data that is only dependent on a part of the key. A relation is said to be in second normal form if and only if it satisfies all the first normal form conditions for the primary key and every non-primary key attributes of the relation is fully dependent on its primary key alone.

Third Normal Form

According to Third Normal Form, Relation should not have a non-key attribute functionally determined by another non-key attribute or by a set of non-key attributes. That is, there should be no transitive dependency on the primary key. In this we decompose and set up relation that includes the non-key attributes that functionally determines other non-key attributes. This step is taken to get rid of anything that does not depend entirely on the Primary Key. A relation is said to be in third normal form if only if it is in second normal form and more over the non key attributes of the relation should not be depend on other non-key attribute.

4.4.3 Sanitization

An automated procedure called "sanitization" is used to get a value ready for use in a SQL query. This process typically involves checking the value for particular characters that have a special significance for the target database. To prevent a SQL injection attack, you must sanitise (filter) the input string while processing a SQL query based on user input. For instance, the user and password input is a typical scenario. In that particular scenario, the server response would provide access to the 'target user' account without requiring a password check

4.4.4 Indexing

By reducing the number of disk accesses needed when a query is completed, indexing helps a database perform better. It is a data structure method used to locate and access data in a database rapidly. Several database columns are used to generate indexes. The primary key or candidate key of the table is duplicated in the first column, which is the Search key. To make it easier to find the related data, these values are kept in sorted order. Recall that the information may or may not be kept in sorted order.

4.5 TABLE DESIGN

1. tbl_login

No.	Field Name	Data Type(size)	Constrains	Description
1.	login_id	int (10)	Primary key	Login id
2.	email	varchar(10)	Not null	Email
3.	phone	int (10)	Not null	Phone
4.	password	varchar (10)	Not null	password

2. tbl_user

No.	Field Name	Data Type(size)	Constrains	Description
1.	User_id	int (10)	Primary key	User id
2.	login_id	int (10)	Foreign key	Login id
3.	fname	varchar (10)	Not null	First name of user
4.	lname	varchar (10)	Not null	Last name of user
5.	address	varchar (10)	Not null	Address of user
6.	gender	varchar (10)	Not null	Gender
7.	city	varchar (10)	Not null	City

2. tbl_category

No.	Field Name	Data Type (size)	Constrains	Description
1.	category_id	int (10)	Primary key	Category id
2.	subcategory_id	int (10)	Foreign key	Subcategory id
3.	category_name	varchar (10)	Not null	Category name
4.	description	varchar (10)	Not null	Description

3. tbl_complent

No.	Field Name	Data Type (size)	Constrains
1.	complent_id	int (10)	Primary key
2.	user_id	varchar (10)	Foreign key
3.	img	varchar (10)	Not null
4.	complent	varchar(50)	Not null

5. tbl_product

No.	Field Name	Data Type(size)	Constrains	Description
1.	product_id	int (10)	Primary key	Product id
2.	pname	int (10)	Foreign key	Product name
3.	description	varchar (10)	Not null	Product description
4.	subcategory_id	int (10)	Foreign key	Subcategory id
5.	price	int (10)	Not null	Price of product
6.	image	varchar (10)	Not null	Image of product
7.	size_id	int (10)	Foreign key	Size of product

6. tbl_buyer

No.	Field Name	Data Type(size)	Constrains
1.	buy_id	int (10)	Primary key
2.	product_name	varchar (10)	Not null
3.	reg_id	Int (10)	Foreign key
4.	No_of items	Int 10)	Not Null

7. tbl_cart

No.	Field Name	Data Type(size)	Constrains	Description
1.	cart_id	int (10)	Primary key	Cart id
2.	product_id	int (10)	Foreign key	Product id
3.	quantity	int (10)	Not null	Quantity
4.	status	Varchar(50)	Not null	status

8. tbl_pet

No.	Field Name	Data Type(size)	Constrains
1.	pet_id	int (10)	Primary key
2.	category	varchar (50)	Not null
3.	p_name	varchar(50)	Not null
4.	color	varchar(50)	Not null
5.	price	int(10)	Not null
6.	age	int(10)	Not null
7.	image	varchar(50)	Not null
8.	gender	varchar(50)	Not null
9.	user_id	int(10)	

9. tbl_order

No.	Field Name	Data Type (size)	Constrains	Description
1.	order_id	int (10)	Primary key	Order id
2.	User_id	int (10)	Not null	User id
3.	product_id	int (10)	Foreign key	Product id
4.	total	int (10)	Not null	Total amount

10.tbl_review

No.	Field Name	Data Type(size)	Constrains	Description
1.	review_id	int (10)	Primary key	Wishlist Id
2.	user_id	int (10)	Foreign key	Product id
3.	description	varchar (50)	Not null	Review of customer

11.tbl_Seller

No.	Field Name	Data Type(size)	Constrains
1.	prod_id	int (10)	Primary key
2.	reg_id	int (10)	Foreign key
3.	Sell_id	int (10)	Not null
4.	product_name	Varchar(20)	Not null

CHAPTER 5

SYSTEM TESTING

5.1 INTRODUCTION

Software Testing is the process of executing software in a controlled manner, in order to answer the question - Does the software behave as specified? Software testing is often used in association with the terms verification and validation. Validation is the checking or testing of items, includes software, for conformance and consistency with an associated specification. Software testing is just one kind of verification, which also uses techniques such as reviews, analysis, inspections, and walkthroughs. Validation is the process of checking that what has been specified is what the user actually wanted.

Other activities which are often associated with software testing are static analysis and dynamic analysis. Static analysis investigates the source code of software, looking for problems and gathering metrics without actually executing the code. Dynamic analysis looks at the behavior of software while it is executing, to provide information such as execution traces, timing profiles, and test coverage information.

Testing is a set of activity that can be planned in advanced and conducted systematically. Testing begins at the module level and work towards the integration of entire computers based system. Nothing is complete without testing, as it vital success of the system testing objectives, there are several rules that can 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 implies a series of desired course of action to be followed in accomplishing various testing methods. The Test Plan acts as a blue print for the action that is to be followed. The software engineers create a computer program, its documentation and related data structures. The software developers is always responsible for testing the individual units of the programs, ensuring that each performs the function for which it was designed. There is an independent test group (ITG) which is to remove the inherent problems associated with letting the builder to test the thing that has been built. The specific objectives of testing should be stated in measurable terms. So that the mean time to failure, the cost to find and fix the defects, remaining defect density or frequency of occurrence and test work-hours per regression test all should be stated within the test plan.

The levels of testing include:

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

5.2.1 Unit Testing

Unit testing focuses verification effort on the smallest unit of software design – the software component or module. Using the component level design description as a guide, important control paths are tested to uncover errors within the boundary of the module. The relative complexity of tests and uncovered scope established for unit testing. The unit testing is white-box oriented, and step can be conducted in parallel for multiple components. The modular interface is tested to ensure that information properly flows into and out of the program unit under test. The local data structure is examined to ensure that data stored temporarily maintains its integrity during all steps in an algorithm's execution. Boundary conditions are tested to ensure that all statements in a module have been executed at least once. Finally, all error handling paths are tested.

Tests of data flow across a module interface are required before any other test is initiated. If data do not enter and exit properly, all other tests are moot. Selective testing of execution paths is an essential task during the unit test. Good design dictates that error conditions be anticipated and error handling paths set up to reroute or cleanly terminate processing when an error does occur. Boundary testing is the last task of unit testing step. Software often fails at its boundaries.

Unit testing was done in Sell-Soft System by treating each module as separate entity and testing each one of them with a wide spectrum of test inputs. Some flaws in the internal logic of the modules were found and were rectified. After coding each module is tested and run individually. All unnecessary code where removed and ensured that all modules are working, and gives the expected result.

5.2.2 Integration Testing

Integration testing is systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing. The objective is to take unit tested components and build a program structure that has been dictated by design. The entire program is tested as whole. Correction is difficult because isolation of causes is complicated by vast expanse of entire program. Once these errors are corrected, new ones appear and the process continues in a seemingly endless loop. After performing unit testing in the System all the modules were integrated to test for any inconsistencies in the interfaces. Moreover differences in program structures were removed and a unique program structure was evolved.

5.2.3 Validation Testing or System Testing

This is the final step in testing. In this the entire system was tested as a whole with all forms, code, modules and class modules. This form of testing is popularly known as Black Box testing or System tests.

Black Box testing method focuses on the functional requirements of the software. That is, Black Box testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements for a program.

Black Box testing attempts to find errors in the following categories; incorrect or missing functions, interface errors, errors in data structures or external data access, performance errors and initialization errors and termination errors.

5.2.4 Output Testing or User Acceptance Testing

The system considered is tested for user acceptance; here it should satisfy the firm's need. The software should keep in touch with perspective system; user at the time of developing and making changes whenever required. This done with respect to the following points:

- Input Screen Designs
- Output Screen Designs

The above testing is done taking various kinds of test data. Preparation of test data plays a vital role in the system testing. After preparing the test data, the system under study is tested using that test data. While testing the system by which test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

5.2.5 Automation Testing

A test case suite is executed using specialised automated testing software tools as part of the software testing technique known as automation testing. The test stages are meticulously carried out by a human performing manual testing while seated in front of a computer. Additionally, the automation testing software may generate thorough test reports, compare expected and actual findings, and enter test data into the System Under Test. Software test automation necessitates significant financial and material inputs. Repeated execution of the same test suite will be necessary during subsequent development cycles. This test suite can be recorded and replayed as needed using a test automation tool. No further human involvement is needed once the test suite has been automated.

5.2.6 Selenium Testing

Selenium is a free and open-source tool for testing web applications across multiple browsers and operating systems. Selenium Test Scripts can be written in different programming languages, including Java, C#, JavaScript, Python, etc. Automation performed using the Selenium framework is referred to as Selenium automation testing.

CHAPTER 6

IMPLEMENTATION

6.1 INTRODUCTION

This Pet Shop Online Shopping System using php was designed to manage the pet information such to produce the pet breed and the vaccine schedules. It is also performed in systematic through its function requirement. The management of pet shop includes the services offers such as cleaning, scissoring, bathing and blow-drying.

First, the pet must be registered before having any services. Any piece of information about the pets must be correct and organized to avoid any problem to be in countered like the bad feedback from the customer. The purpose of this system is to transact and deals with a nice and easier way by simply gathering information from the customer. It also helps to every individual by searching for the information needed. project image

6.2 IMPLEMENTATION PROCEDURES

Implementation is the stage in the project where the theoretical design is turned into a working system and is giving confidence on the system for the users that it will work efficiently and effectively. It involves careful planning, investigation of the current system and its constraints on implementation, design of methods to achieve the changeover, an evaluation of change over methods. The implementation process begins with preparing a plan for the implementation of the system. According to this, the activities are to be carried out, discussions made regarding the equipment and resources and the additional equipment has to be acquired to implement the new system. In PETKAR system no resources are needed.

6.2.1 User Training

User training is designed to prepare the user for testing and converting the system. To achieve the objective and benefits expected from computer based system, it is essential for the people who will be involved to be confident of their role in the new system. As system becomes more complex, the need for training is more important. By user training the user comes to know how to enter data, respond to error messages, interrogate the database and call up routine that will produce reports and perform other necessary functions.

6.2.2 Training on the Application Software

User training is designed to prepare the user for testing and converting the system. To achieve the objective and benefits expected from computer based system, it is essential for the people who will be involved to be confident of their role in the new system. As system becomes more complex, the need for training is more important. By user training the user comes to know how to enter data, respond to error messages, interrogate the database and call up routine that will produce reports and perform other necessary functions.

6.2.3 System Maintenance

Maintenance is the enigma of system development. The maintenance phase of the software cycle is the time in which a software product performs useful work. After a system is successfully implemented, it should be maintained in a proper manner. System maintenance is an important aspect in the software development life cycle. The need for system maintenance is for it to make adaptable to the changes in the system environment. Software maintenance is of course, far more than "Finding Mistakes".

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

All organisation where small or large computerize their areas that help to reduce manual work and also save the time the largest storage capacities of computer help to store data and their manipulation with a short period In this website anyone can be registered and the login after login they can't decide buy or sell In this website are used to make the members much more closer to the pet shop to sell and buy the products The system may allow members to search pets, food and vaccination and also able to find the pets or food status details and after considering the various feasible solution the most feasible one was selected for designing take into consideration the time and efficiency constants all the efforts have been put to make sure that the system handle the account processing efficiently the performance of the system was evaluated to determine whether the system achieved the results that were accepted and whether the predicted benefits of the system where being realized.

7.2 FUTURE SCOPE

Proposed system overcome the issues of existing system.in this website anyone can register and login. After login they can decide buy or sell. If you choose buy option you can choose category and it will display's all items available in that category. You can buy or sell that item. Items like, pets, foods, caves etc. you can also know about pets breeding time, vaccination, necessary food details. If you choose seller option then you can add your item in this application. Also you can see orders and payments. Reviews and ratings. Add complaints to admin about someone.

- A mobile application can be developed

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
- The Complete reference PHP by Steven Holzner
- The Complete reference MySQL by Vikram Vaswani
- CSS Cookbook by Christopher Schmitt

WEBSITES:

- www.w3schools.com
- www.jquery.com
- <http://homepages.dcc.ufmg.br/~rodolfo/es-1-03/IEEE-Std-830-1998.pdf>
- www.agilemodeling.com/artifacts/useCaseDiagram.html
- www.geeksforgeeks.com
- www.tutorialspoint.org

CHAPTER 9

APPENDIX

9.1 Sample Code

9.1.1. LOGIN PAGE & SIGNUP

```
<div class="modal fade" id="myModal" tabindex="-1" role="dialog">
<div class="modal-dialog">
<!-- Modal content-->
<div class="modal-content">
<div class="modal-header">
<button type="button" class="close" data-dismiss="modal">&times;</button>
</div>
<div class="modal-body modal-body-sub_agile">
<div class="col-md-8 modal_body_left modal_body_left1">
<h3 class="agileinfo_sign">Sign In <span>Now</span></h3>
<form action="#" method="post">
<div class="styled-input agile-styled-input-top">
<input type="text" name="email" required="">
<label>Email</label>
<span></span>
</div>
<div class="styled-input">
<input type="password" name="password" required="">
<label>Password</label>
<span></span>
</div>
<input type="submit" name="login" value="Sign In">
</form>
<ul class="social-nav model-3d-0 footer-social w3_agile_social top_agile_third">
<li><a href="#" class="facebook">
<div class="front"><i class="fa fa-facebook" aria-hidden="true"></i></div>
<div class="back"><i class="fa fa-facebook" aria-hidden="true"></i></div></a></li>
<li><a href="#" class="twitter">

<p><a href="#" data-toggle="modal" data-target="#myModal2" > Don't have an
account?</a></p>

</div>
<div class="col-md-4 modal_body_right modal_body_right1">

</div>
<div class="clearfix"></div>
</div>
</div>
<!-- //Modal content-->
</div>
</div>
<?php
```

```

if(isset($_REQUEST['login']))
{
extract($_REQUEST);
$query="select * from user_reg where email='$email' and password='$password'";

$login_data=select($query);
$n=mysqli_num_rows($login_data);
if($n==1)
{
while($data=mysqli_fetch_array($login_data))
{
extract($data);

}

$_SESSION['userid']=$user_reg_id;
$_SESSION['name']=$name;
//$_SESSION['image']=$image;
$_SESSION['login']="yes";

//echo "1";
echo '<script>alert("Login Successful")</script>';
}
else
{
echo "email or password is incorrect";
}
}

?>
<label>Email</label>
<span></span>
</div>
<div class="styled-input">
<input type="password" name="password" required="">
<label>Password</label>
<span></span>
</div>
<div class="styled-input">
<input type="password" name="Confirm_Password" required="">
<label>Confirm Password</label>
<span></span>
</div>
<input type="submit" name="signup" value="Sign Up">
</form>
<ul class="social-nav model-3d-0 footer-social w3_agile_social top_agile_third">
<li><a href="#" class="facebook">
<div class="front"><i class="fa fa-facebook" aria-hidden="true"></i></div>
<div class="back"><i class="fa fa-facebook" aria-hidden="true"></i></div></a></li>
<li><a href="#" class="twitter">

```

```

<div class="front"><i class="fa fa-twitter" aria-hidden="true"></i></div>
<div class="back"><i class="fa fa-twitter" aria-hidden="true"></i></div></a></li>
<li><a href="#" class="instagram">
<div class="front"><i class="fa fa-instagram" aria-hidden="true"></i></div>
<div class="back"><i class="fa fa-instagram" aria-hidden="true"></i></div></a></li>
<li><a href="#" class="pinterest">
<div class="front"><i class="fa fa-linkedin" aria-hidden="true"></i></div>
<div class="back"><i class="fa fa-linkedin" aria-hidden="true"></i></div></a></li>
</ul>
<div class="clearfix"></div>
<p><a href="#">By clicking register, I agree to your terms</a></p>

```

```

</div>
<div class="col-md-4 modal_body_right modal_body_right1">

</div>
<div class="clearfix"></div>
</div>
</div>
<!-- //Modal content-->
</div>
</div>

```

```

<?php
if(isset($_REQUEST['signup']))
{
$password=$_POST['password'];
$confirm_password=$_POST['Confirm_Password'];
if($password==$confirm_password){

extract($_REQUEST);
$n=iud("INSERT INTO `user_reg`(`name`,`email`,`password`) VALUES
('$Name','$Email','$password')");
if($n==1)
{
echo'<script>alert("Signup Successful")</script>';
}
else
{
echo'<script>alert("Something Wrong Try Again ")</script>';
}
}
else{
echo'<script>alert("password does not match")</script>';
}
}
?>

```


9.1.2. INDEX.PHP

```
</div>

<?php endif ?>

<?php require_once"dbconfig.php";
if(isset($_SESSION['login']))
{

}

else
{
header("location:login.php");
}

?>

<!DOCTYPE html>

<head>

<title>seller planel</title>

<script type="application/x-javascript"> addEventListener("load", function() {
setTimeout(hideURLbar, 0); }, false); function hideURLbar(){ window.scrollTo(0,1); }
</script>

<link rel="stylesheet" href="css/bootstrap.min.css" >

<link href="css/style.css" rel='stylesheet' type='text/css' />

<link href="css/style-responsive.css" rel="stylesheet"/>

<link
href="//fonts.googleapis.com/css?family=Roboto:400,100,100italic,300,300italic,400italic,5
00,500italic,700,700italic,900,900italic" rel='stylesheet' type='text/css'>

<link rel="stylesheet" href="css/font.css" type="text/css"/>

<link href="css/font-awesome.css" rel="stylesheet">

<link rel="stylesheet" href="css/morris.css" type="text/css"/>

<link rel="stylesheet" href="css/monthly.css">

<script src="js/jquery2.0.3.min.js"></script>

<script src="js/raphael-min.js"></script>

<script src="js/morris.js"></script>

</head>

<body>
```

```
<section id="container">
<header class="header fixed-top clearfix">
<div class="brand">
<a href="index.php" class="logo">
SELLER
</a>
<div class="sidebar-toggle-box">
<div class="fa fa-bars"></div>
</div>
</div>
<?php include"nav_top.php";?>

</header>
<?php include"sidebar.php";?>

<div class="clearfix"> </div>
</div>
</section>
</section>
<script src="js/bootstrap.js"></script>
<script src="js/jquery.dcjaccordion.2.7.js"></script>
<script src="js/scripts.js"></script>
<script src="js/jquery.slimscroll.js"></script>
<script src="js/jquery.nicescroll.js"></script>
<!--[if lte IE 8]><script language="javascript" type="text/javascript" src="js/flot-
chart/excanvas.min.js"></script><![endif]-->
<script src="js/jquery.scrollTo.js"></script>
</body>
</html>
```

9.1.2. SELLER INDEX_PAGE.PHP

```

<?php require_once"dbconfig.php";

?>
<!DOCTYPE html>
<head>
<script type="application/x-javascript"> addEventListener("load", function() {
setTimeout(hideURLbar, 0); }, false); function hideURLbar(){ window.scrollTo(0,1); } </script>
<link rel="stylesheet" href="css/bootstrap.min.css" >
<link href="css/style.css" rel='stylesheet' type='text/css' />
<link href="css/style-responsive.css" rel="stylesheet"/>
<link
href="//fonts.googleapis.com/css?family=Roboto:400,100,100italic,300,300italic,400italic,500,50
0italic,700,700italic,900,900italic" rel='stylesheet' type='text/css'>
<link rel="stylesheet" href="css/font.css" type="text/css"/>
<link href="css/font-awesome.css" rel="stylesheet">
<script src="js/jquery2.0.3.min.js"></script>
</head>
<body>
<div class="reg-w3">
<div class="w3layouts-main">
<h2>Register Now</h2>
<form method="post">
<input type="text" class="ggg" name="Name" placeholder="NAME" required="">
<input type="email" class="ggg" name="Email" placeholder="E-MAIL" required="">
<input type="text" class="ggg" name="Phone" placeholder="PHONE" required="">
<input type="password" class="ggg" name="Password" placeholder="PASSWORD"
required="">
<h4><input type="checkbox" />I agree to the Terms of Service and Privacy Policy</h4>

<div class="clearfix"></div>
<input type="submit" value="submit" name="register">
</form>

<?php
if(isset($_REQUEST['register']))
{
extract($_REQUEST);
$n=iud("INSERT INTO `registration`(`name`,`email`,`mobile`,`password`) VALUES
('$Name','$Email','$Phone','$Password')");
if($n==1)
{
echo'<script>alert("successful")</script>';
header("location:login.php");
}
}

```

```

}

?>
<p>Already Registered.<a href="login.php">Login</a></p>
</div>
</div>
<script src="js/bootstrap.js"></script>
<script src="js/jquery.dcjqaccordion.2.7.js"></script>
<script src="js/scripts.js"></script>
<script src="js/jquery.slimscroll.js"></script>
<script src="js/jquery.nicescroll.js"></script>
<!--[if lte IE 8]><script language="javascript" type="text/javascript" src="js/flot-
chart/excanvas.min.js"></script><![endif]-->
<script src="js/jquery.scrollTo.js"></script>
</body>
</html>

```

9.1.2. ADMIN_INDEX.PHP

```

<?php

include"dbconfig.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, shrink-to-
fit=no">
<meta name="description" content="">
<meta name="author" content="">

<title>Admin </title>

<!-- Custom fonts for this template-->
<link href="vendor/fontawesome-free/css/all.min.css" rel="stylesheet"
type="text/css">
<link
href="https://fonts.googleapis.com/css?family=Nunito:200,200i,300,300i,400,400i,60
0,600i,700,700i,800,800i,900,900i" rel="stylesheet">

<!-- Custom styles for this template-->
<link href="css/sb-admin-2.min.css" rel="stylesheet">

</head>

```

```

<body id="page-top">

<div id="wrapper">

<!-- Sidebar -->
<?php include"sidebar.php";?>

<div id="content-wrapper" class="d-flex flex-column">

<div id="content">
<nav class="navbar navbar-expand navbar-light bg-white topbar mb-4 static-top
shadow">
<button id="sidebarToggleTop" class="btn btn-link d-md-none rounded-circle mr-
3">
<i class="fa fa-bars"></i>
</button>

<!-- Topbar Search -->
<h2>Admin </h2>

<!-- Topbar Navbar -->
<ul class="navbar-nav ml-auto">
<div class="topbar-divider d-none d-sm-block"></div>

<li class="nav-item dropdown no-arrow">
<a class="nav-link dropdown-toggle" href="#" id="userDropdown" role="button"
data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
<span class="mr-2 d-none d-lg-inline text-gray-600 small">ADMINS</span>

</a>
<!-- Dropdown - User Information -->
<div class="dropdown-menu dropdown-menu-right shadow animated--grow-in" aria-
labelledby="userDropdown">
<a class="dropdown-item" href="#">
<i class="fas fa-user fa-sm fa-fw mr-2 text-gray-400"></i>
Profile
</a>
<a class="dropdown-item" href="#">
<i class="fas fa-cogs fa-sm fa-fw mr-2 text-gray-400"></i>
Settings
</a>
<a class="dropdown-item" href="#">
<i class="fas fa-list fa-sm fa-fw mr-2 text-gray-400"></i>
Activity Log
</a>
<div class="dropdown-divider"></div>
<a class="dropdown-item" href="#" data-toggle="modal" data-
<div class="form-group">
<div class="custom-control custom-checkbox small">

```

```

<input type="checkbox" class="custom-control-input" id="customCheck">
<label class="custom-control-label" for="customCheck">Remember Me</label>
</div>
</div>
<input type="submit" value="Login" name="login" class="btn btn-primary btn-user
btn-block">

<hr>

</form>
</div>
</div>
</div>
</div>

<!-- Pie Chart -->
<div class="col-xl-4 col-lg-5">
<?php
if(isset($_REQUEST['login']))
{

$email=trim($_REQUEST['email']);
$password=trim($_REQUEST['password']);
$query="select * from admin where email='$email' and password='$password'";
$login_data=select($query);
$n=mysqli_num_rows($login_data);
if($n==1)
{
while($data=mysqli_fetch_array($login_data))
{
extract($data);

}

//$_SESSION['id']=$id;
$_SESSION['login']="yes";

echo'<script>alert("login success")
window.location="add_cat.php"
</script>';
}
else
</div>
<a class="scroll-to-top rounded" href="#page-top">
<i class="fas fa-angle-up"></i>
</a>

<!-- Logout Modal-->
<div class="modal fade" id="logoutModal" tabindex="-1" role="dialog" aria-
labelledby="exampleModalLabel" aria-hidden="true">
<div class="modal-dialog" role="document">

```

```
<div class="modal-content">
<div class="modal-header">
<h5 class="modal-title" id="exampleModalLabel">Ready to Leave?</h5>
<button class="close" type="button" data-dismiss="modal" aria-label="Close">
<span aria-hidden="true">×</span>
</button>
</div>
<div class="modal-body">Select "Logout" below if you are ready to end your current
session.</div>
<div class="modal-footer">
<button class="btn btn-secondary" type="button" data-
dismiss="modal">Cancel</button>
<a class="btn btn-primary" href="login.php">Logout</a>
</div>
</div>
</div>
</div>
</div>

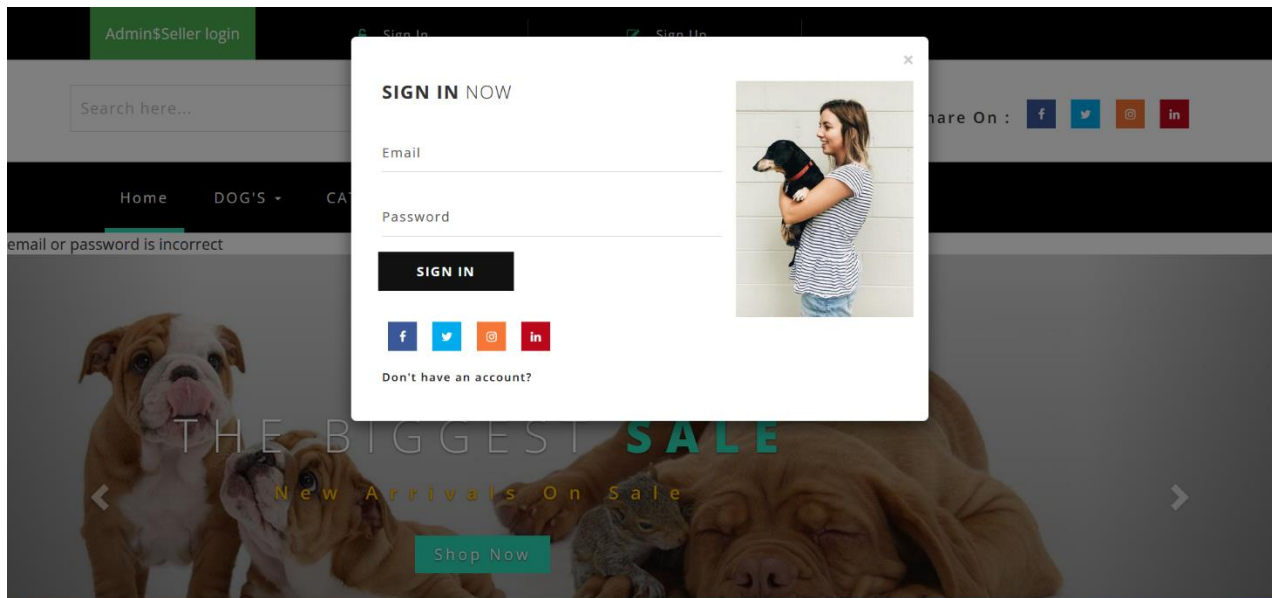
<script src="vendor/jquery/jquery.min.js"></script>
<script src="vendor/bootstrap/js/bootstrap.bundle.min.js"></script>

<script src="vendor/jquery-easing/jquery.easing.min.js"></script>
<script src="js/sb-admin-2.min.js"></script>
<script src="vendor/chart.js/Chart.min.js"></script>
<!-- Page level custom scripts -->
<script src="js/demo/chart-area-demo.js"></script>
<script src="js/demo/chart-pie-demo.js"></script>
</body>

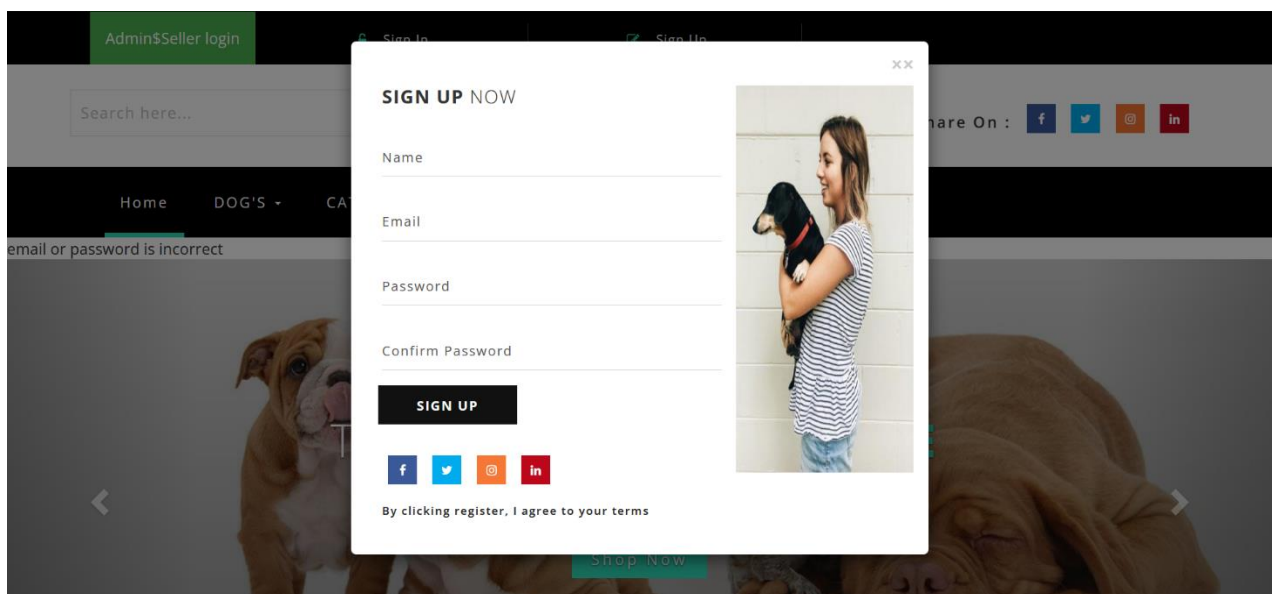
</html>
```

10.1. Screen Shots

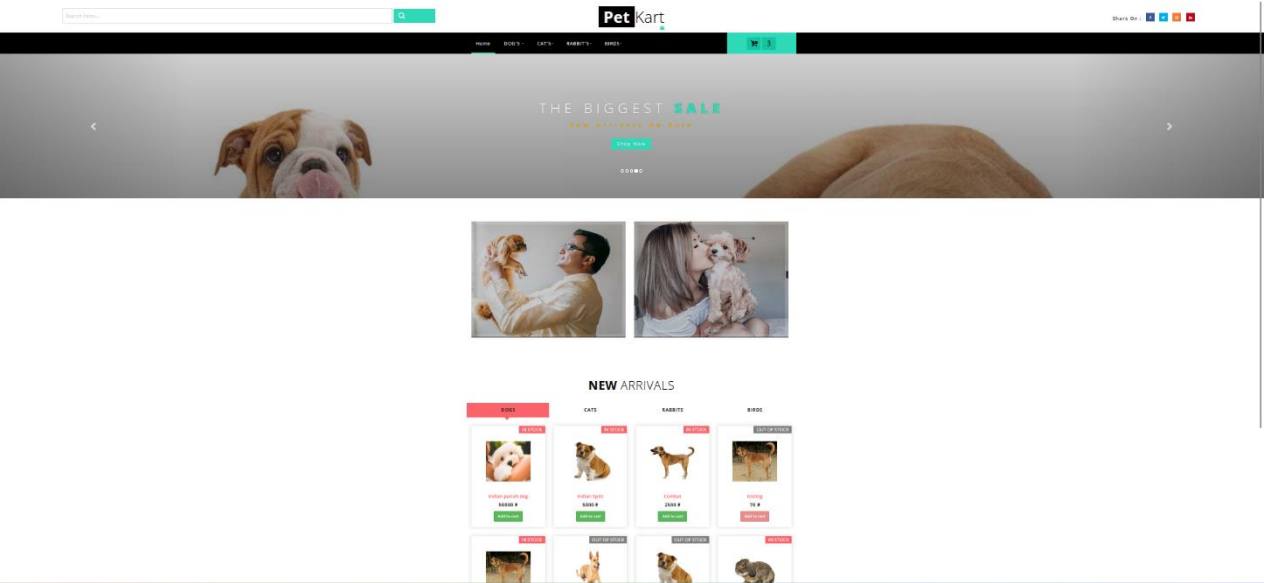
SIGNIN PAGE



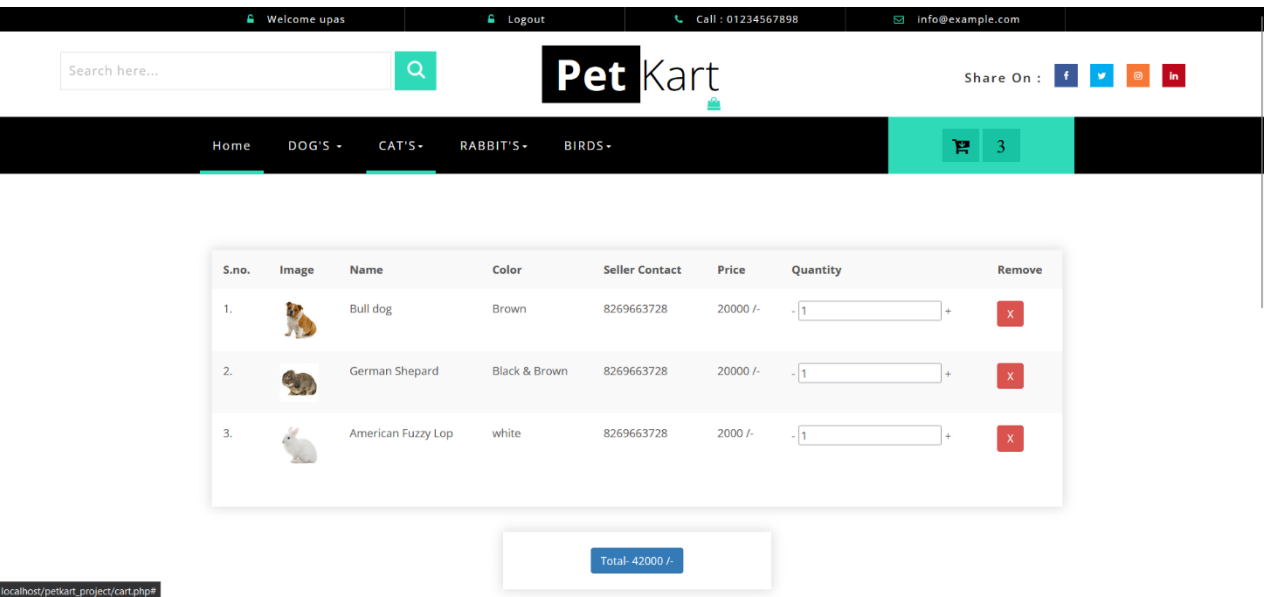
SIGNUP PAGE




HOME PAGE



CART PAGE



ADMIN DASHBOARD

 ADMIN

ADD PET CATEGORY

ADD SUB-CATEGORY

VIEW PET CATEGORY


VIEW SUB-CATEGORY

CART

VIEW SELLERS

VIEW USERS

LOGOUT





Dashboard

Category Name

Submit

SELLER DASHBOARD

SELLER



Dashboard

ADD PETS

VIEW PETS

PLACED ENQUIRY

Logout

CALENDAR WIDGET

<NOV 2022>

SUN	MON	TUE	WED	THU	FRI	SAT
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

