Chapter 1

Preamble

1.1 Introduction

A database is an organized collection of data. A relational database, more restrictively, is a collection of schemas, tables, queries, reports, views and other elements. A database management system(DBMS) is a computer software application that interacts with endusers, other applications, and the database itself to capture and analyse data. A general purpose DBMS allows the definition, creation, querying, update, and administration of databases. There is a need for an application to make it easy for ZOO to maintain their records and have track of tickets revenue. Zoo database management streamlines everything about animal details, zookeeper details, visitor details etc. It plays a important role in maintaining all these records.

1.1.1 Database Management System(DBMS)

Following the technology progress in the areas of processors, computer memory, computer storage, and computer networks, the sizes, capabilities, and performance of databases and their respective DBMSs have grown in orders of magnitude. The development of database technology can be divided into three eras based on data model or structure: navigational, SQL/relational, and post-relational. The two main early navigational data models were the hierarchical model, epitomized by IBM's IMS system, and the CODASYL model (network model), implemented in a number of products such as IDMS.

The relational model employees sets of ledger-style tables, each used for a different type of entity. Only in the mid-1980s did computing hardware become powerful enough to allow the wide deployment of relational systems (DBMSs plus applications). By the early 1990s, however, relational systems dominated in all large-scale data processing applications, and as of 2015 they remain dominant: IBM DB2, Oracle, MySQL, and Microsoft SQL Server are the top DBMS. The dominant database language, standardized SQL for the relational model, has influenced database languages for other data models.

1.1.2 Visual Studio Code

Visual Studio Code is a source-code editor made by Microsoft for Windows, Linex and macOS . Features include support for debugging , syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality.

Visual Studio Code is a source-code editor that can be used with a variety of programming languages, including Java, JavaScript, Go, Node.js, Python and C++. Visual Studio Code also ships with IntelliSense for TypeScript JSON,CSS, HTML and PHP.

1.1.3 MVC

MVC (Model-View-Controller) is a pattern in software design commonly used to implement user interfaces, data, and controlling logic. It emphasizes a separation between the software's business logic and display. This "separation of concerns" provides for a better division of labor and improved maintenance. Model will manage data and business logic. View handles layout and display. Controller routes commands to the model and view parts.

1.1.4 MySql

MySQL is a fast, easy-to-use RDBMS being used for many small and big businesses. It is developed, marketed and supported by MySQL AB, which is a Swedish company. MySQL is becoming so popular because of many good reasons −It is released under an open-source license. So, you have nothing to pay to use it, it is a very powerful program in its own right and handles a large subset of the functionality of the most expensive and powerful database packages. MySQL uses a standard form of the well-known SQL data language. It works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc and works very quickly and works well even with large data sets.

1.1.5 PHP

PHP Hypertext Pre-processor is a scripting language that helps people make web pages more interactive by allowing them to do more things. PHP code is run on the web server. A website programmed with PHP can have pages that are password protected. A website with no programming cannot do this without other complex things. Standard PHP file extensions are: .php .php3 or .phtml, but a web server can be set up to use any extension. Its structure was influenced by many languages like C, Perl, Java, C++, and even Python. It is considered to be free software by the Free Software Foundation.

1.1.6 CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a mark-up language like HTML.CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript. CSS is designed to enable the separation of presentation and content, including layout, colours, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate CSS file, and reduce complexity and repetition in the structural content. Separation of formatting and content also makes it feasible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices. CSS also has rules for alternate formatting if the content is accessed on a mobile device. The name cascading comes from the specified priority scheme to determine which style rule applies if more than one rule matches a element. This cascading priority scheme is predictable

1.1.7 HTML

Hypertext Mark-up Language (HTML) is the standard mark-up language for creating web pages and web applications. With Cascading Style Sheets (CSS) and Java Scripts, it forms a triad of cornerstone technologies for the World Wide Web. Web browser receives HTML documents from a web server or from local storage and renders the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document. HTML can embed programs written in a scripting

language such as JavaScript, which affects the behaviour and content of web pages. Inclusion of CSS defines the look and layout of content.

1.1.8 Bootstrap

Bootstrap is an HTML, CSS & JS Library that focuses on simplifying the development of informative web pages (as opposed to web apps). The primary purpose of adding it to a web project is to apply Bootstrap's choices of color, size, font and layout to that project. As such, the primary factor is whether the developers in charge find those choices to their liking. Once added to a project, Bootstrap provides basic style definitions for all HTML elements. The result is a uniform appearance for prose, tables and form elements across web browser. In addition, developers can take advantage of CSS classes defined in Bootstrap to further customize the appearance of their contents. For example, Bootstrap has provisioned for light- and dark-colored tables, page headings, more prominent pull quotes, and text with a highlight.

1.1.9 JavaScript

JavaScript is a text-based programming language used both on the client-side and server-side that allows you to make web pages interactive. Where HTML and CSS are languages that give structure and style to web pages. JavaScript gives web pages interactive elements that engage a user. A news recap video embedded on THE NEW YORK TIMES, or refreshing your Twitter Feed. Incorporating JavaScript improves the user experience of the web page by converting it from a static page into an interactive one. To recap, JavaScript adds behavior to web pages.

1.1.10 Laragon

Laragon is a portable, isolated, fast and powerful universal development environment for PHP, Node.js, Python, Java, Go, Ruby. It is fast, lightweight, easy-to-use and easy-to-extend.It is great for building and managing modern web applications. It is focused on performance and managing modern web applications. It is focused on performance-designed around stability, simplicity, flexibility and freedom. The core binary itself is less than 2MB and uses less than 4MB RAM when running. It has it's own service orchestraition which manages

services asynchronously and non-blocking so you'll find things run fast & smoothly.

1.1.11 Normalization

Normalization is a process of organizing the data in database to avoid data redundancy, insertion anomaly, update anomaly & deletion anomaly. To overcome these anomalies, we need to normalize the data. There are 4 basic types of normalization :

- First normal form(1NF)
- Second normal form(2NF)
- Third normal form(3NF)
- Boyce & codd normal form(BCNF)

First normal form(1NF) is defined as an attribute of a table cannot hold multiple values. It should hold only atomic values. This means that there shouldn't be repetition of data in the tables

Second normal form(2NF) it satisfy if two conditions are satisfied. The table is in first normal form and all the non-prime attribute are dependent on the proper subset of any candidate key of table. The attribute that is not part of any candidate key are known as non-prime attribute.

A table design is said to be in 3NF if the table is in 2NF and transitive functional dependency of non-prime attribute o any super key are removed

Boyce Codd normal form(BCNF) is the advance version of 3NF that's why it is also referred as 3.5NF, BCNF is stricter than 3NF. A table compiles with BCNF if it is in 3NF and for every functional dependency X->Y, X should be the super-key of the table.

1.2 Objectives

Objectives of the **ZOO DATABASE MANAGEMENT SYSTEM** are**:**

- Easy to determine current animal condition
- Record of tickets revenue
- Maintaining records of medication for animals

- Efficiency to keep track of all details
- To reduce manual effort and usage of papers.

1.3 Organization of report

Chapter 1 provides the information about the information of MySQL, HTML, CSS, PHP and Visual Studio Code etc. In Chapter 2, we discuss the software and hardware requirements to run the above applications. Chapter 3 gives the idea of the project and its actual implementation. Chapter 4 discusses about the results and discussions of the program. Chapter 5 concludes by giving the direction for future enhancement.

1.4 Summary

The chapter discussed before is an overview about the PHP, MVC, DBMS etc. The scope of study and objectives of the project are mentioned clearly. The organization of the report is been pictured to increase the readability. Further, coming up chapters depicts the use of various queries to implement various changes like insert, update, delete and also triggers to nperform varios functions.

Chapter 2

Requirement Specifications

2.1 Software Specification

Operating system:Windows 9,10,11

Front end:HTML,CSS,BOOTSTRAP

Back end:PHP(MVC) and MYSQL

2.2 Hardware Specification

Computer with a 1.1 GHz or faster processor

Minimum 2GB of RAM or more

2.5 GB of available hard-disk space

5400 RPM hard drive

1366 × 768 or higher-resolution display

2.3 User Characteristics

Every user:

-should be comfortable with basic working of computer

-Must have basic knowledge of english

-Must carry login ID and password used for authentication

Chapter 3

SYSTEM DESIGN AND IMPLEMENTATION

3.1 Introduction

System is a collection of an interrelated components that works together to achieve a purpose. System analysis is referred to the systematic examination or detailed study of a system in order to identify problems of the system, and using the information gathered in the analysis stage to recommend improvements or solution to the system.

System design is an abstract representation of a system component and their relationship and which describe the aggregated functionality and performance of the system. System design is also the overall plan or blueprint for how to obtain answer to the question being asked. The design specifies which of the various type of approach.

System design is the process or art of defining the architecture, components, modules, interfaces and data for a system to satisfy specified requirements. One could see it as the application of system theory to product development.

3.2 ER Diagram

An entity-relationship model or the ER Diagram describes interrelated things of interest in a specific domain of knowledge. An ER model is composed of entity types and specifies relationships that can exist between instances of those entity types.

In software engineering ER model is commonly formed to represent things that a business needs remember in order to perform business processes. Consequently, the ER model becomes an abstract model that defines a data or informational structure that can be implemented in a database, typically a relational database.

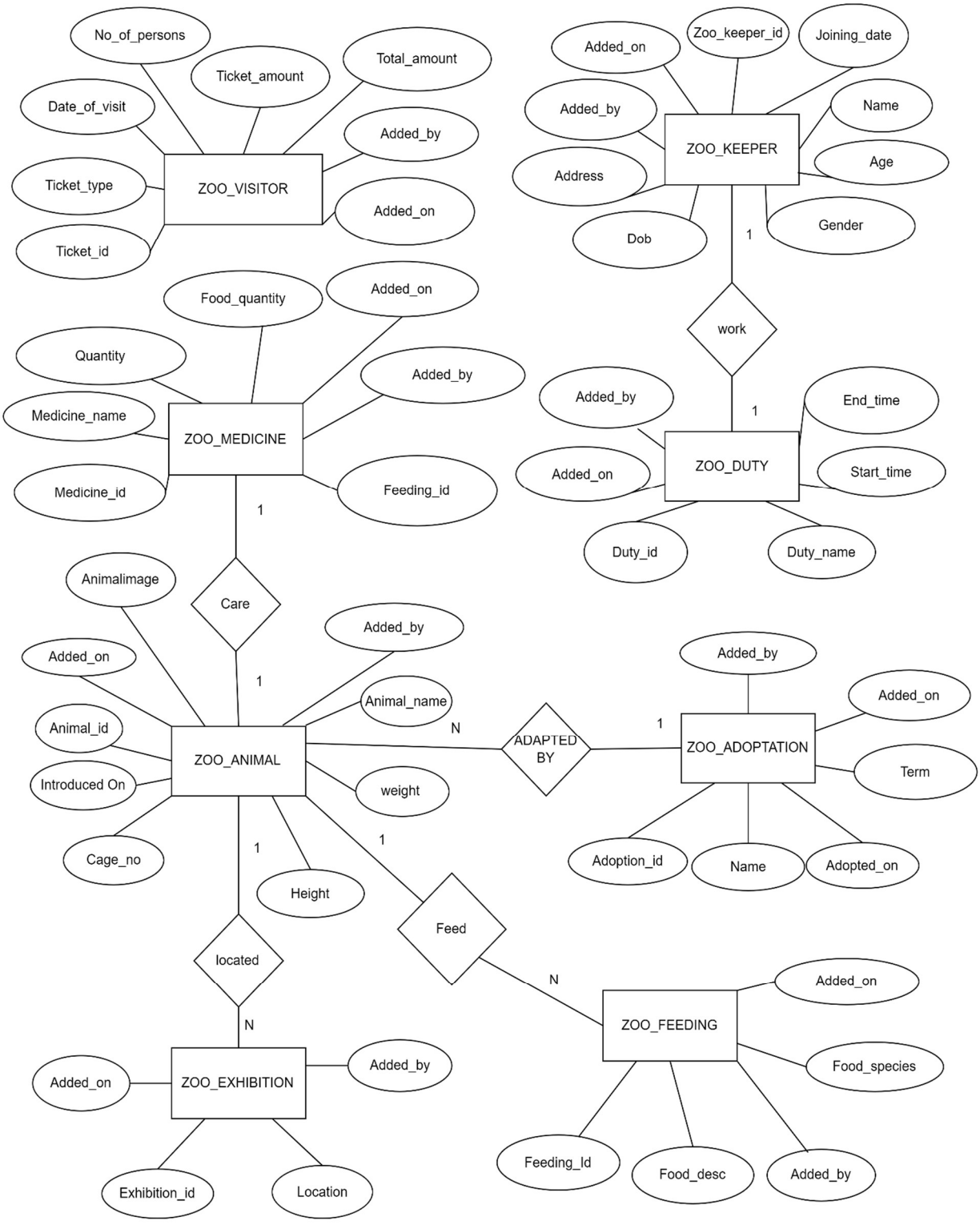


Figure 3.2: ER Diagram

3.3 Schema Diagram

The schema diagram of a database system is it's structure described in a formal language supported by the Database Management System (DBMS). The formal definition of a database schema is a set of formulas called integrity constraints imposed on a database.

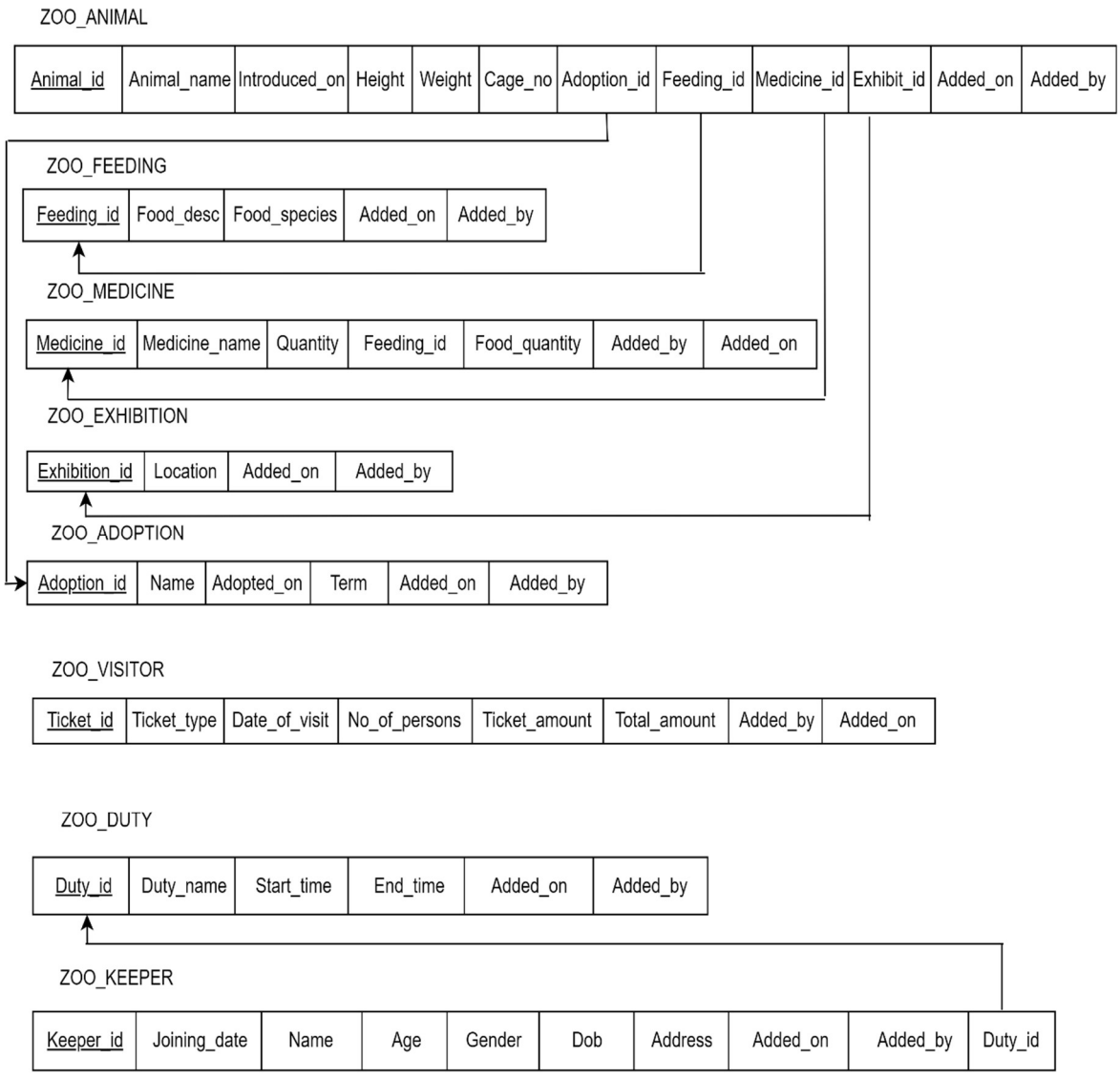


Figure 3.3: Schema Diagram

The term “Schema” refers to the organization of data as blueprint of how the database is constructed. These integrity constraints ensure compatibility between parts of the schema. All constraints are expressible in the same language. A database can be considered a structure in realization of the database language. The states of a created conceptual schema are transformed into an explicit mapping, the database schema. This describes how real world entities are modelled in the database.

3.4 Queries

The below mentioned are all the queries used to perform various tasks in MYSQL such as insert, delete, update. A short description of the query is also provided.

```
CREATE TABLE zoo_animal (
  animal_id int(25) NOT NULL,
  animal_name varchar(250) DEFAULT NULL,
  introduced_on varchar(250) DEFAULT NULL,
  height varchar(250) DEFAULT NULL,
  weight varchar(250) DEFAULT NULL,
  cage_no varchar(250) DEFAULT NULL,
  adopted_on varchar(250) DEFAULT NULL,
  adoption_id int(25) DEFAULT NULL,
  feeding_id int(25) DEFAULT NULL,
  medicine_id int(20) DEFAULT NULL,
  exibhit_id int(11) DEFAULT NULL,
  added_on datetime NOT NULL DEFAULT CURRENT_TIMESTAMP,
  added_by varchar(250) DEFAULT NULL,
  image_url varchar(250) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Name	Type	Collation	Attributes	Null	Default	Comments	Extra
animal_id	int(25)			No	*None*		AUTO_INCREMENT
animal_name	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
introduced_on	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
height	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
weight	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
cage_no	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
adopted_on	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
adoption_id	int(25)			Yes	*NULL*		
feeding_id	int(25)			Yes	*NULL*		
medicine_id	int(20)			Yes	*NULL*		
exibhit_id	int(11)			Yes	*NULL*		
added_on	datetime			No	CURRENT_TIMESTAMP		
added_by	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
image_url	varchar(250)	latin1_swedish_ci		Yes	*NULL*		

```
CREATE TABLE zoo_feeding (
  feeding_id int(25) NOT NULL,
  food_desc varchar(250) DEFAULT NULL,
  food_species varchar(250) DEFAULT NULL,
  added_on datetime NOT NULL DEFAULT CURRENT_TIMESTAMP,
  added_by varchar(250) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Name	Type	Collation	Attributes	Null	Default	Comments	Extra
feeding_id	int(25)			No	*None*		AUTO_INCREMENT
food_desc	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
food_species	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
added_on	datetime			No	CURRENT_TIMESTAMP		
added_by	varchar(250)	latin1_swedish_ci		Yes	*NULL*		

```
CREATE TABLE zoo_medicine (
  medicine_id int(20) NOT NULL,
  medicine_name varchar(250) DEFAULT NULL,
```

```
  quantity varchar(250) DEFAULT NULL,
  feeding_id int(25) DEFAULT NULL,
  food_quantity varchar(250) DEFAULT NULL,
  added_by varchar(250) DEFAULT NULL,
  added_on datetime NOT NULL DEFAULT CURRENT_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Name	Type	Collation	Attributes	Null	Default	Comments	Extra
medicine_id	int(20)			No	*None*		AUTO_INCREMENT
medicine_name	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
quantity	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
feeding_id	int(25)			Yes	*NULL*		
food_quantity	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
added_by	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
added_on	datetime			No	CURRENT_TIMESTAMP		

```
CREATE TABLE zoo_exibition (
  exibhition_id int(11) NOT NULL,
  location varchar(250) DEFAULT NULL,
  added_on datetime NOT NULL DEFAULT CURRENT_TIMESTAMP,
  added_by varchar(250) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Name	Type	Collation	Attributes	Null	Default	Comments	Extra
exibhition_id	int(11)			No	*None*		AUTO_INCREMENT
location	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
added_on	datetime			No	CURRENT_TIMESTAMP		
added_by	varchar(250)	latin1_swedish_ci		Yes	*NULL*		

```
CREATE TABLE zoo_adoption (
  adoption_id int(25) NOT NULL,
  name varchar(250) DEFAULT NULL,
  term varchar(250) DEFAULT NULL,
  added_on datetime NOT NULL DEFAULT CURRENT_TIMESTAMP,
  added_by varchar(250) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Name	Type	Collation	Attributes	Null	Default	Comments	Extra
adoption_id	int(25)			No	*None*		AUTO_INCREMENT
name	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
term	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
added_on	datetime			No	CURRENT_TIMESTAMP		
added_by	varchar(250)	latin1_swedish_ci		Yes	*NULL*		

```
CREATE TABLE zoo_visitor (
  ticket_id int(100) NOT NULL,
  ticket_type varchar(250) DEFAULT NULL,
  date_of_visit varchar(250) DEFAULT NULL,
  no_of_persons varchar(250) DEFAULT NULL,
```

```
  ticket_amount varchar(250) DEFAULT NULL,
  total_amount varchar(250) DEFAULT NULL,
  added_by varchar(250) DEFAULT NULL,
  added_on datetime NOT NULL DEFAULT CURRENT_TIMESTAMP,
  token_id varchar(250) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Name	Type	Collation	Attributes	Null	Default	Comments	Extra
ticket_id	int(100)			No	*None*		AUTO_INCREMENT
ticket_type	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
date_of_visit	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
no_of_persons	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
ticket_amount	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
total_amount	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
added_by	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
added_on	datetime			No	CURRENT_TIMESTAMP		
token_id	varchar(250)	latin1_swedish_ci		Yes	*NULL*		

```
CREATE TABLE zoo_duty (
  duty_id int(20) NOT NULL,
  duty_name varchar(250) DEFAULT NULL,
  start_time varchar(100) DEFAULT NULL,
  end_time varchar(100) DEFAULT NULL,
  added_on datetime NOT NULL DEFAULT CURRENT_TIMESTAMP,
  added_by varchar(100) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Name	Type	Collation	Attributes	Null	Default	Comments	Extra
duty_id	int(20)			No	*None*		AUTO_INCREMENT
duty_name	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
start_time	varchar(100)	latin1_swedish_ci		Yes	*NULL*		
end_time	varchar(100)	latin1_swedish_ci		Yes	*NULL*		
added_on	datetime			No	CURRENT_TIMESTAMP		
added_by	varchar(100)	latin1_swedish_ci		Yes	*NULL*		

```
CREATE TABLE zoo_keeper (
  zoo_keeper_id int(20) NOT NULL,
  joining_date varchar(250) DEFAULT NULL,
  name varchar(250) DEFAULT NULL,
  age int(25) DEFAULT NULL,
  gender varchar(250) DEFAULT NULL,
  dob varchar(250) DEFAULT NULL,
  address varchar(250) DEFAULT NULL,
  added_by varchar(250) DEFAULT NULL,
  added_on datetime NOT NULL DEFAULT CURRENT_TIMESTAMP,
  duty_id int(20) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Name	Type	Collation	Attributes	Null	Default	Comments	Extra
zoo_keeper_id	int(20)			No	*None*		AUTO_INCREMENT
joining_date	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
name	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
age	int(25)			Yes	*NULL*		
gender	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
dob	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
address	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
added_by	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
added_on	datetime			No	CURRENT_TIMESTAMP		
duty_id	int(20)			Yes	*NULL*		

```
CREATE TABLE users (
  id int(6) UNSIGNED NOT NULL,
  name varchar(250) DEFAULT NULL,
  username varchar(200) NOT NULL,
  password varchar(200) NOT NULL,
  email varchar(200) DEFAULT NULL,
  img_url varchar(250) DEFAULT NULL,
  created_on datetime NOT NULL DEFAULT CURRENT_TIMESTAMP,
  ip_address varchar(200) DEFAULT NULL,
  user_type varchar(250) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Name	Type	Collation	Attributes	Null	Default	Comments	Extra
id	int(6)		UNSIGNED	No	*None*		AUTO_INCREMENT
name	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
username	varchar(200)	latin1_swedish_ci		No	*None*		
password	varchar(200)	latin1_swedish_ci		No	*None*		
email	varchar(200)	latin1_swedish_ci		Yes	*NULL*		
img_url	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
created_on	datetime			No	CURRENT_TIMESTAMP		
ip_address	varchar(200)	latin1_swedish_ci		Yes	*NULL*		
user_type	varchar(250)	latin1_swedish_ci		Yes	*NULL*		

Inserting values into Tables

Actual data stored in the database

zoo_animal Table

INSERT INTO zoo_animal (animal_id, animal_name, introduced_on, height, weight, cage_no, added_on, added_by, image_url) VALUES (1, 'LION','2021-12-27','1.2m','350kg','3A-1','2022-01-12 23:53:00','admin','jde_marks.jpg'),

(2,'TIGER','2020-02-27','1.3m','320kg','3A-2','2021-11-23 12:21:32','admin','kjk_marks.jpg'),

(3,'GORILLA','2021-09-12','2m','80kg','3A-3','2021-10-12 18:12:21','admin','bjp_animal.jpg'),

(4,'PEACOCK','2019-12-21','1.5m','30kg','3A-4','2021-05-29 22:12:20','admin','con_animmal.jpg');

zoo_feeding Table

INSERT INTO zoo_feeding (feeding_id, food desc, food_species, added_on,added_by) VALUES (1,'Dry grass','Pavo cristatus','2022-01-12 18:25:10','admin),

(2,'meat','panthera leo','2021-12-12 19:10:01','admin'),

(3,'meat','panthera tigris','2021-01-31 20:12:23','admin'),

(4,'grains','psittaciformes',2020-12-13 04:20:21','admin');

zoo_medicine Table

INSERT INTO zoo_medicine (medicine_id, medicine_name, quantity, feeding_id, food_quantity, added_by, added_on) VALUES (1,'Azithromine zp 10mg','5 pills',2,'25kg','admin', '2022-01-14 00:28:28'),

(2,' Medicinal Rhuburb','3pills','3','8kg','admin','2021-10-12 12:12:32'),

(3,' Sulfamethoxazole','5pills','6','5kg','admin','2022-01-12 20:21:42'),

(4,’ Sulfadiazine‘,’8pills’,’1’,’9kg’,’admin’,’2021-12-23 17:13:22’);

zoo_exhibition Table

INSERT INTO zoo_exhibition(exhibition_id,location, ,added_on, added_by) VALUES (1,’mysuru’,’2021-02-12 23:12:14’,’admin’),

(2,’bengaluru’,’2022-01-20 19:12:23’,’admin’),

(3,’mangaluru’,’2021-12-03 17:12:18’,’admin’),

(4,’kolar’,’2020-12-13 23:12:18,’admin’);

zoo_adoption Table

INSERT INTO zoo_adoption(adoption_id,name,term,added_on,added_by) VALUES (2, ‘pramod, ‘5’, ‘2022-01-12 20:54:51’, ‘admin’),

(4, ‘anoop’, ‘11’, ‘2022-01-12 20:56:04’, ‘admin’),

(6,’keshav’,’16’,’2021-12-14 19:13:13’,’admin’),

(7,’ram’,’9’,’2022-01-20 19:13:12’,’admin’);

Update values into zoo_animal table

UPDATE zoo_animal SET adoption_id=2, feeding_id=1, medicine_id=2, exhibit_id=1 WHERE animal_id=1;

UPDATE zoo_animal SET adoption_id=3, feeding_id=2, medicine_id=3, exhibit_id=2 WHERE animal_id=2;

UPDATE zoo_animal SET adoption_id=4, feeding_id=3, medicine_id=4, exhibit_id=3 WHERE animal_id=3;

UPDATE zoo_animal SET adoption_id=5, feeding_id=4, medicine_id=5, exhibit_id=4 WHERE animal_id=4;

UPDATE zoo_animal SET adoption_id=6, feeding_id=5, medicine_id=6, exhibit_id=5 WHERE animal_id=5;

zoo_duty Table

INSERT INTO zoo_duty (duty_id, duty_name, start_time, end_time, added_on, added_by)VALUES(1,'night_shift','10:00 pm,'06:00 am','2021-12-14 02:13:00','admin'),

(3,'morning_shift','07:00 am','1:00 pm','2022-01-12 16:40:36','admin'),

(4,'afternoon_shift','02:00 pm','06:00 pm','2022-01-15 18:12:14','admin'),

(5,'evening_shift','05:00 pm','08:00 pm','2021-12-15 19:19:10','admin');

zoo_keeper Table

INSERT INTO zoo_keeper (keeper_id, joining_date, name,age, gender, dob, address, added_by, added_on, duty_id) VALUES (1,'2021-12-31','shiva raman', 45,'male','1975-12-09','Mysuru', 'admin','2022-01-12 15:09:42',3),

(3,'2021-01-05','Vijay Kumar',20,'male','2000-09-07','Chikkaballapura', 'admin', '2022-01-12 15:45:46',1);

(5,'2021-05-29','kruti',25,'male','1997-01-12','Hubli','admin','2021-04-13 19:23:13',3),

(7,'2022-01-13','Seema',20,'female','2001-05-29','kolar','admin',2022-01-02 17:22:12',6);

3.4.1 Stored Procedures

A stored procedure is a prepared SQL code that you can save, so the code can be reused over and over again. So if you have an SQL query that write over and over again, save it as a stored procedure, and then just call to execute it. You can also pass parameters to a stored procedures, so that the stored procedure can act based on the parameter values that is passed.

DELIMITER//

CREATE PROCEDURE zookeeperdetails()

BEGIN

SELECT * from zoo_keeper;

END//

DELIMETER;

DESCRIPTION: The procedure used here is zookeeperdetails, which displays all details of the zoo_keeper table. Here all data of zookeepers who joined in the zoo, their details saved into the zookeeperdetails().But it's even stored into the zoo_keeper table which is the actual table created for the zookeepers data to be stored.

3.4.2 Triggers

Triggers are stored programs, which are automatically executed or fired when some events occur. Triggers are stored into database and invoked repeatedly, when specific condition match.

Triggers used:

1: Trigger name: insertLog

Table: logs

Time: after

Event: insert

INSERT INTO logs VALUES (null,New.animal_id,'Inserted',NOW())

2: Trigger name: updateLog

Table: logs

Time: after

Event: update

INSERT INTO logs VALUES (null,New.animal_id,'Updated',NOW())

id	animal_id	action	cdate
1	6	Inserted	2022-01-30 17:33:27
2	6	updated	2022-01-30 17:37:26
3	7	Inserted	2022-01-30 17:40:21
4	6	updated	2022-01-30 17:41:08
5	8	Inserted	2022-01-30 17:46:01
6	9	Inserted	2022-01-30 17:49:03
7	10	Inserted	2022-01-30 17:51:26
8	11	Inserted	2022-01-30 17:53:25
9	12	Inserted	2022-01-30 17:56:40
10	1	updated	2022-01-31 15:52:21

Fig: Represents the records of logs table

3.5 Pseudo Code

Pseudocode is an informal high-level description of the operating principle of a computer program or other algorithm. It uses the structural conventions of a normal programming language, but is intended for human reading rather than machine reading.

3.5.1 Algorithm for login

Step 1: BEGIN

Step 2: Enter username and password

Step 3: Verify the credentials entered with that in the login table

Step 4: If credentials match then proceed to the entrance

Else show login failed

Step 5: End if

Step 6: END

3.5.2 Algorithm for Table Display

Step1: BEGIN

Step 2: Establish connection with the database using the username and password ofthe database.

Step 3: Define the select query to retrieve all the values from the DBMS

Step 4: Define desc table_name to display structure of the table

Step 5: END

3.5.3 Algorithm for Insert

Step 1: BEGIN

Step 2: Get all the necessary values required for insertion into variable defined in the above insertion of values into table as shown in above content.

Step 3: Define the query for insertion as stated above.

Step 4: Execute the Query using the **(Select * from)** the required table to see inserted values

Step 5: END

3.5.4 Algorithm for Update

Step 1: BEGIN

Step 2: Get all the necessary values required for updating the values into the variable defined in the above update of values into table as shown in above content

Step 3: UPDATE table name
SET column1 = value1, column2= value2….
WHERE condition;

Step 4: Define the Query for Updating as stated above

Step 5: Execute the Query using the (Select * from) the required table to see updated values

Step 6: END

3.5.5 Algorithm for Delete

Step 1: BEGIN

Step 2: Get the animal_id of the zoo_animal which is to be deleted

Step 3: Delete from table_name where condition;

Step 4: Define the Query for deleting as stated in step 3

Step 5: Execute the Query using the (Select * from) the required table to see whether the values are deleted or not

Step 6: END

Chapter 4

Results and Discussions

The project is compiled and executed using **laragon software** and **MySQL**. We have put infew screen shots here to show the working of our Application.

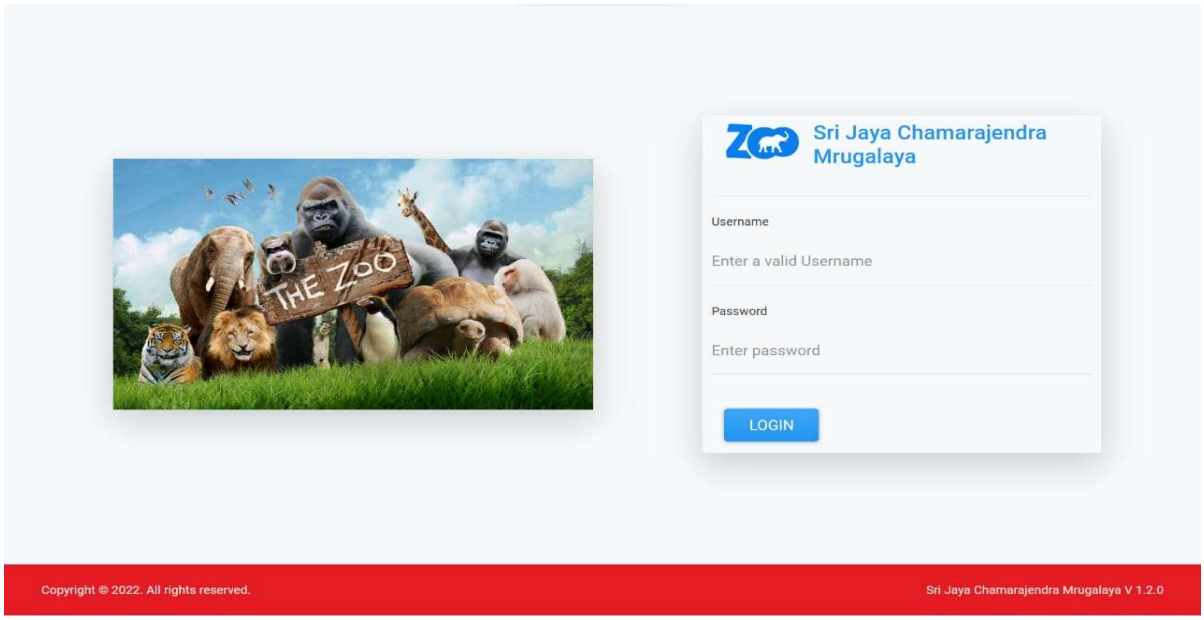


Figure 4.1: Login Page

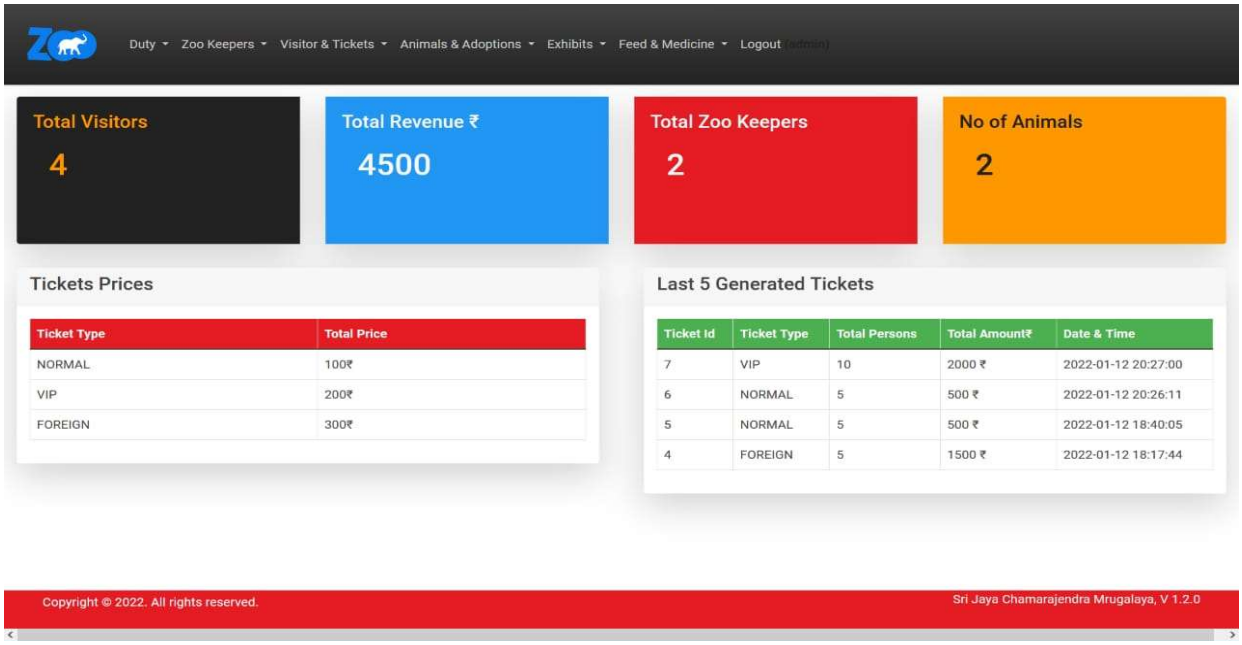


Figure 4.2: Dashboard Page

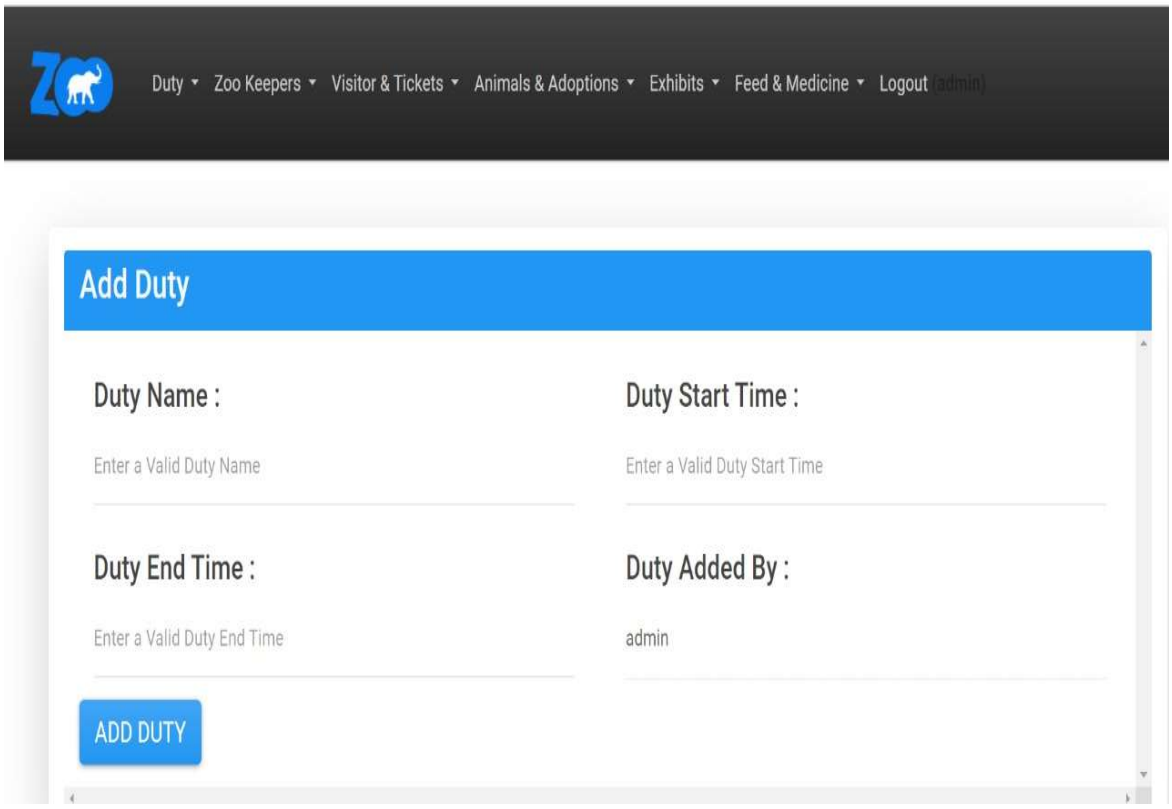


Figure 4.3: Add Duty Page

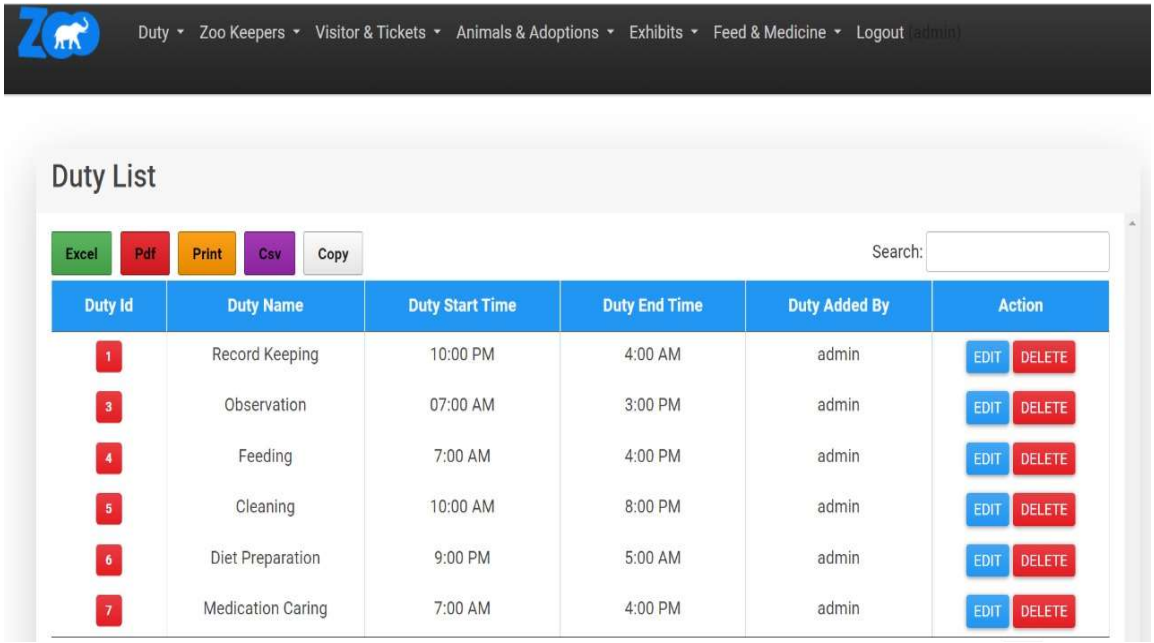


Figure 4.4: Duty List Page

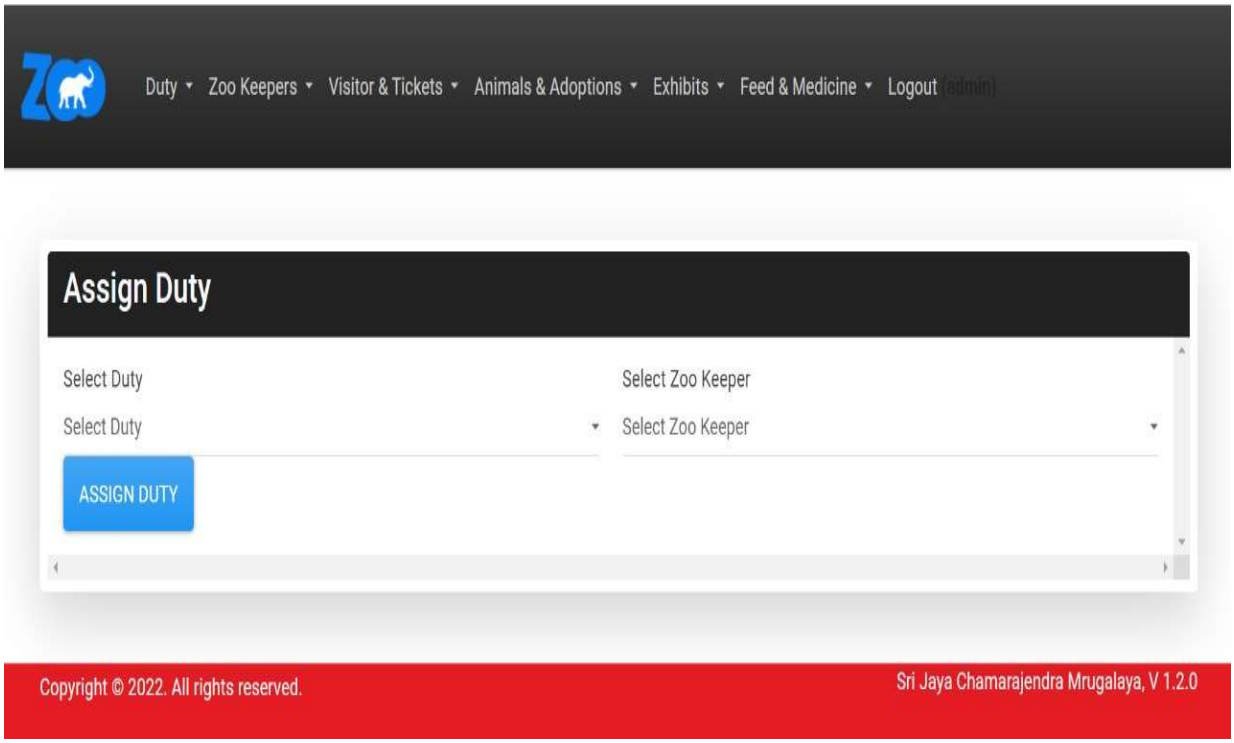


Figure 4.5: Assign Duty Page

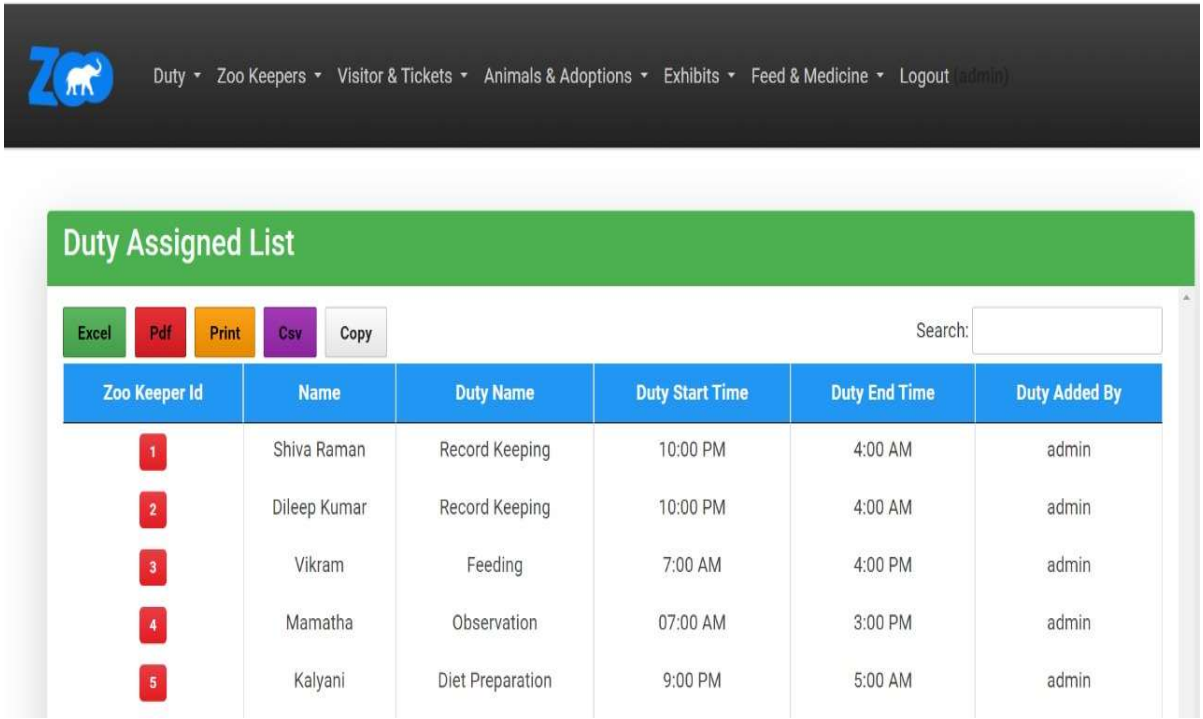


Figure 4.6: Duty Assigned List Page

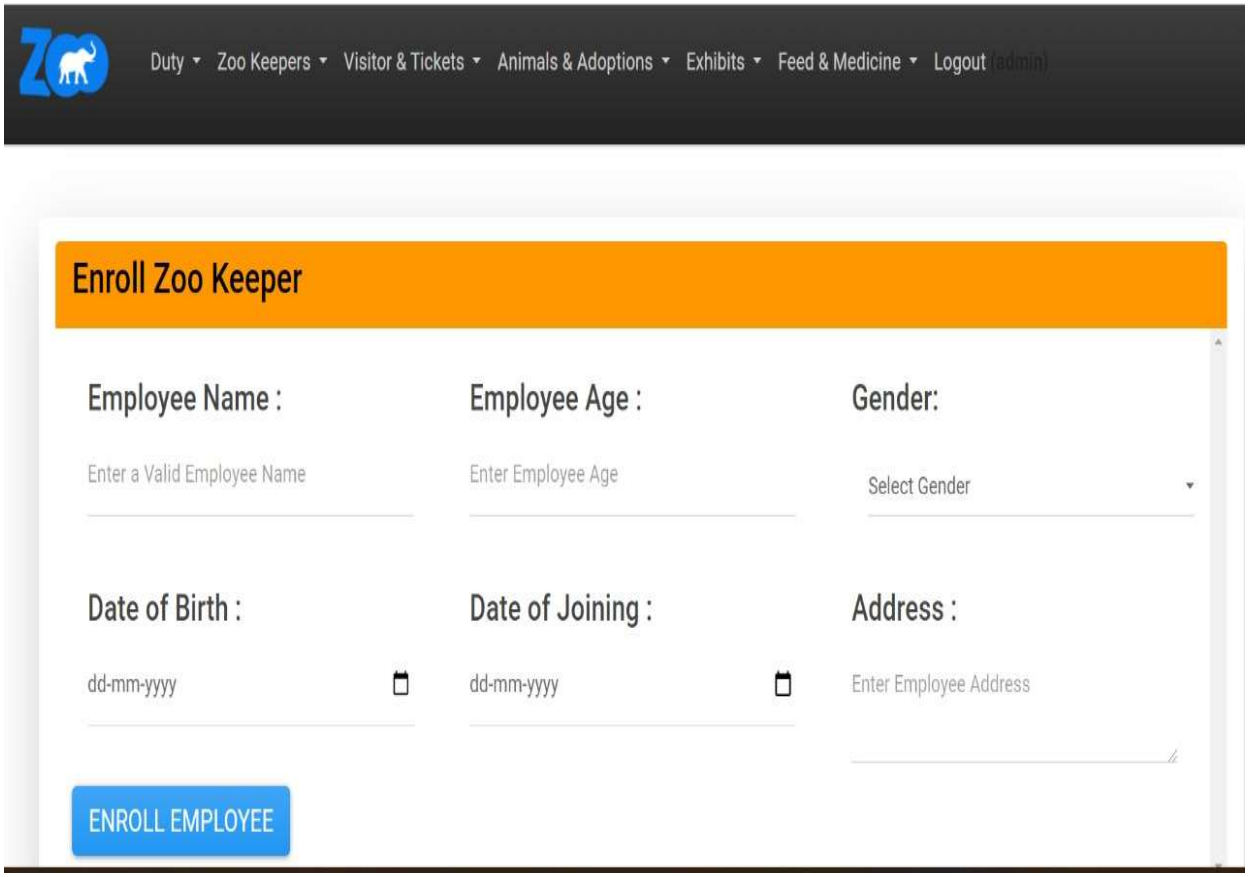


Figure 4.7: Enroll Zoo Keeper Page

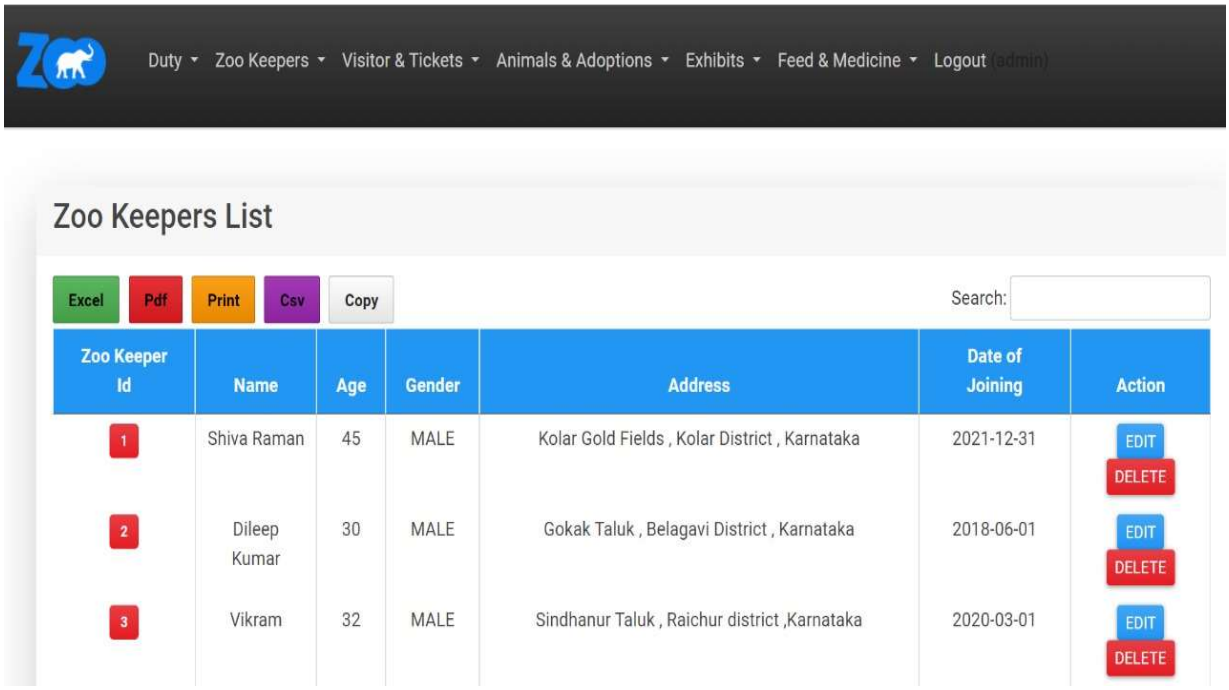


Figure 4.8: Zoo Keepers List Page

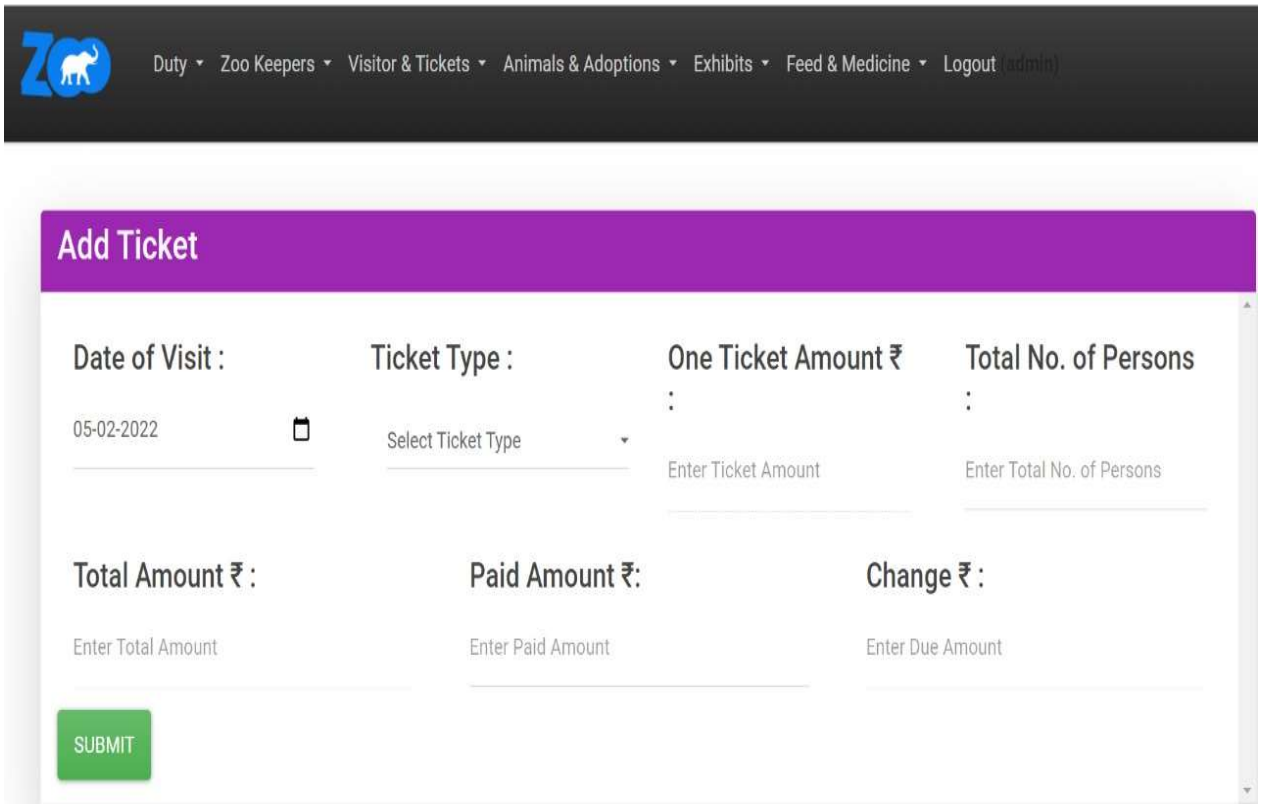


Figure 4.9: Add Ticket Page

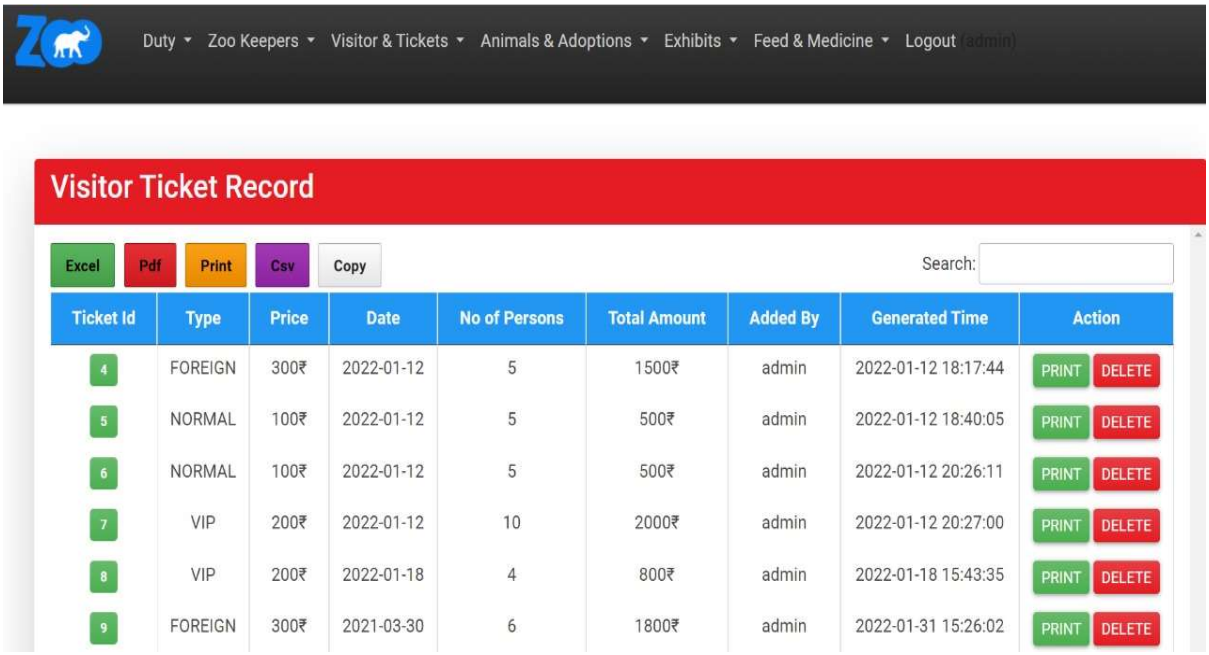


Figure 4.10: Visitor Ticket Record Page

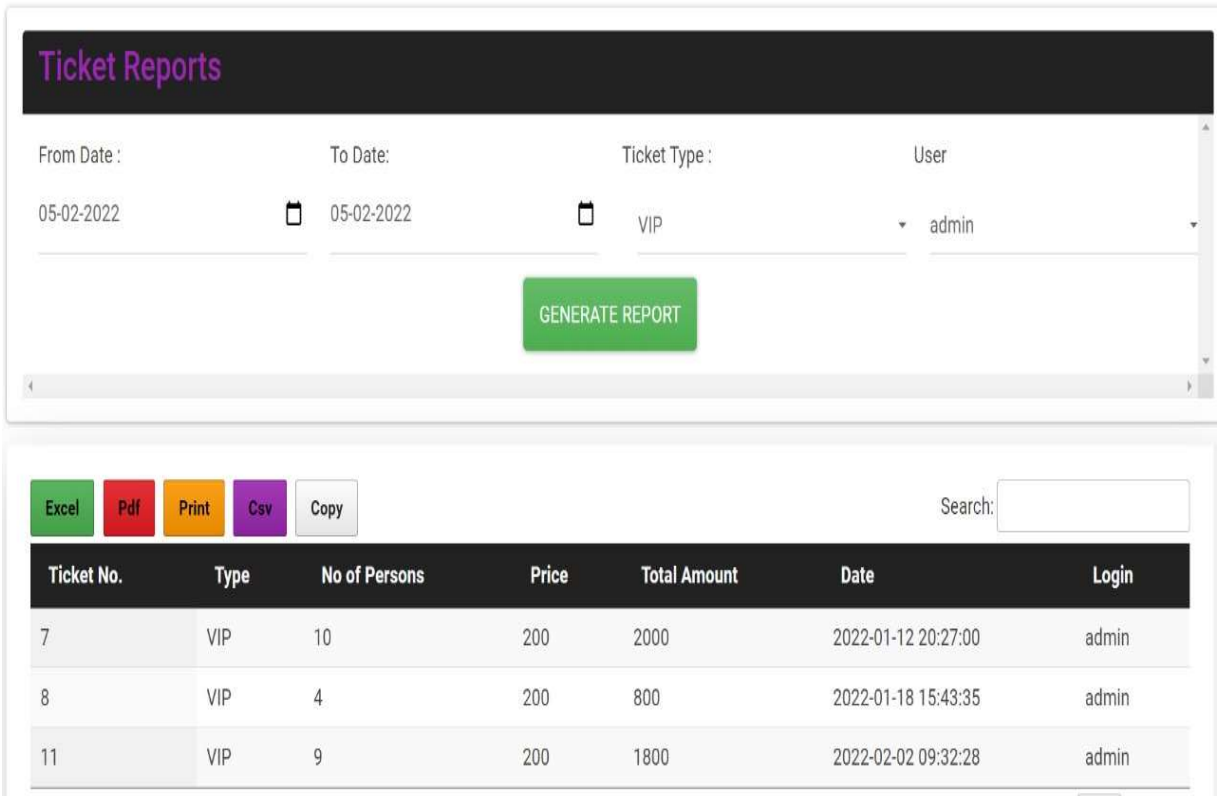


Figure 4.11: Ticket Reports Page

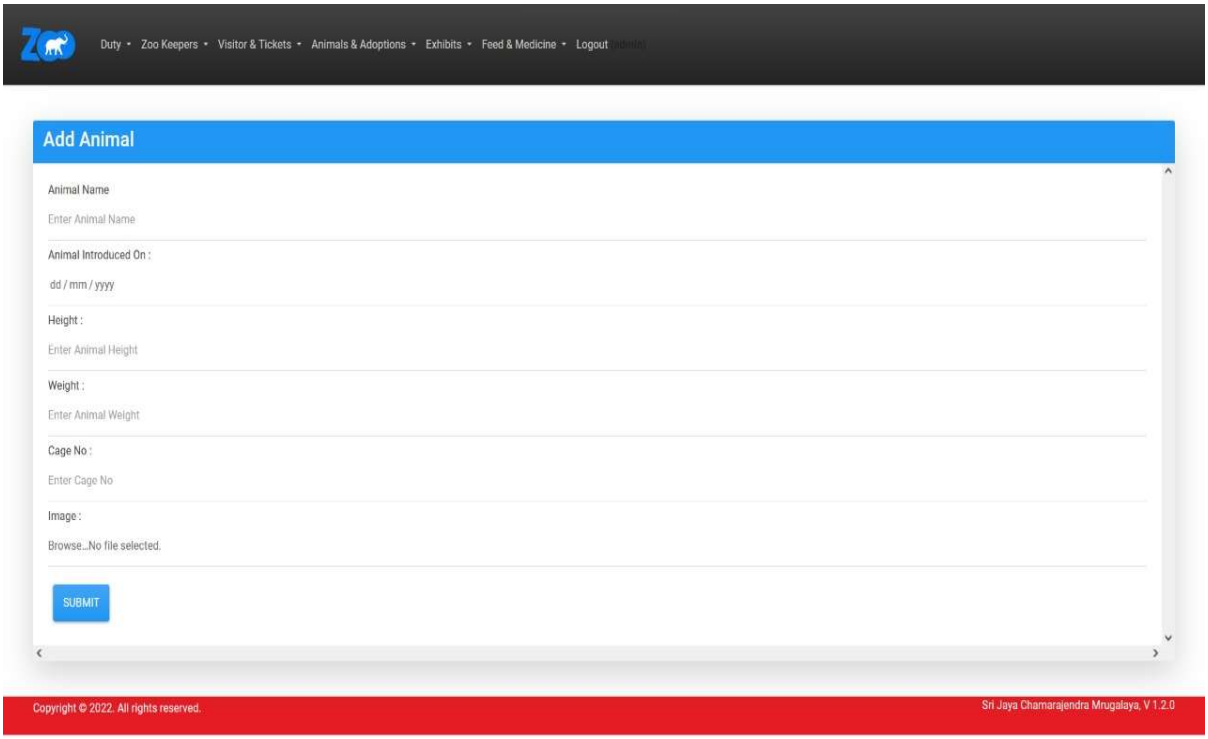


Figure 4.12: Add Animal Page

Figure 4.13: Animals List Page

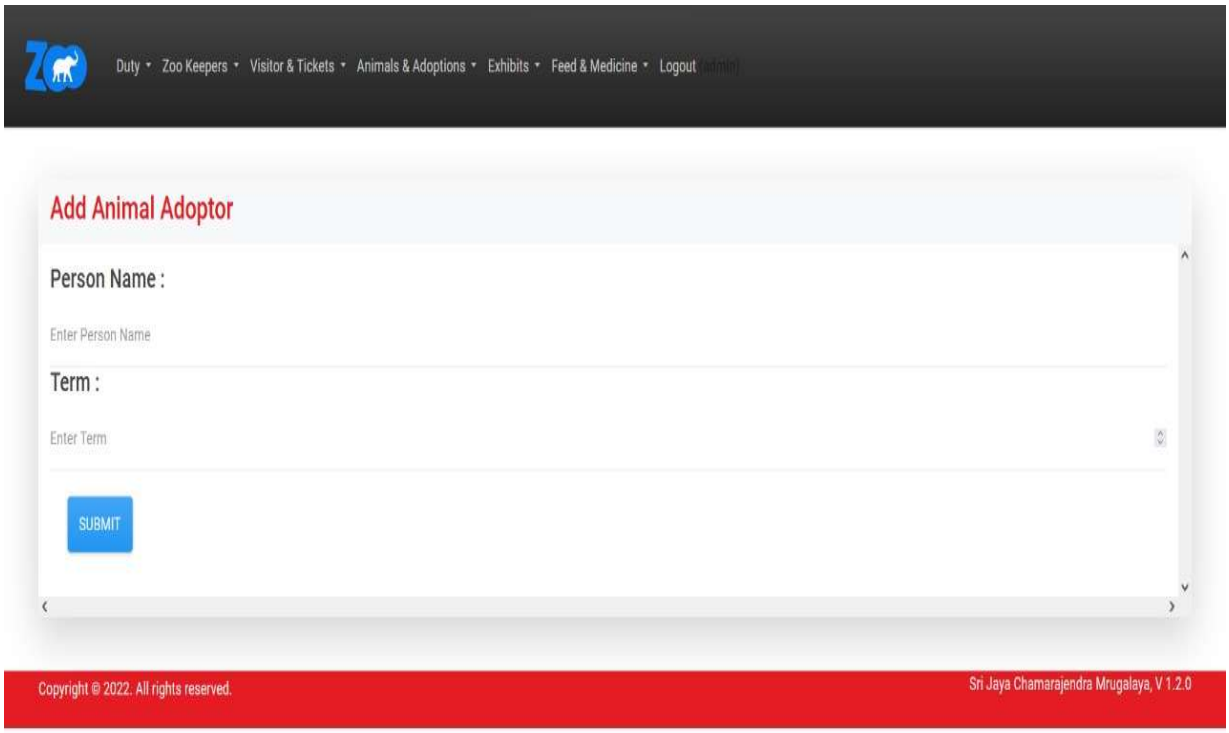


Figure 4.14: Add Animal Adopter Page

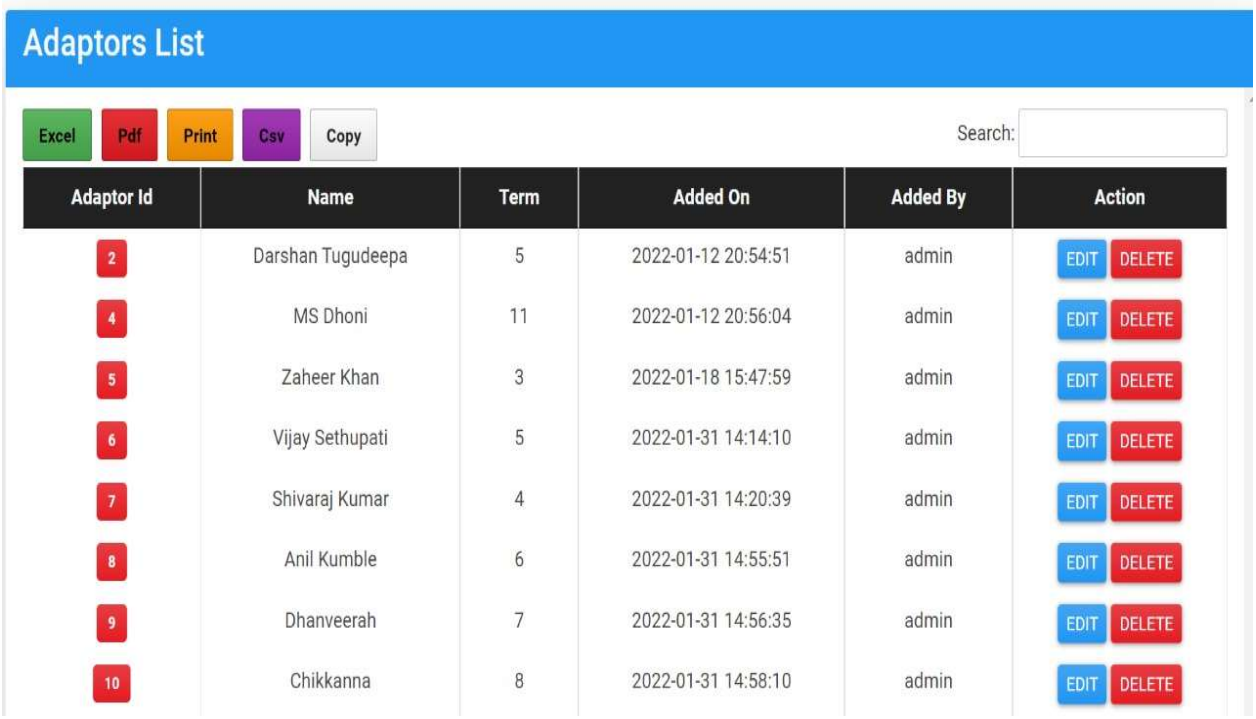


Figure 4.15: Adopters List Page

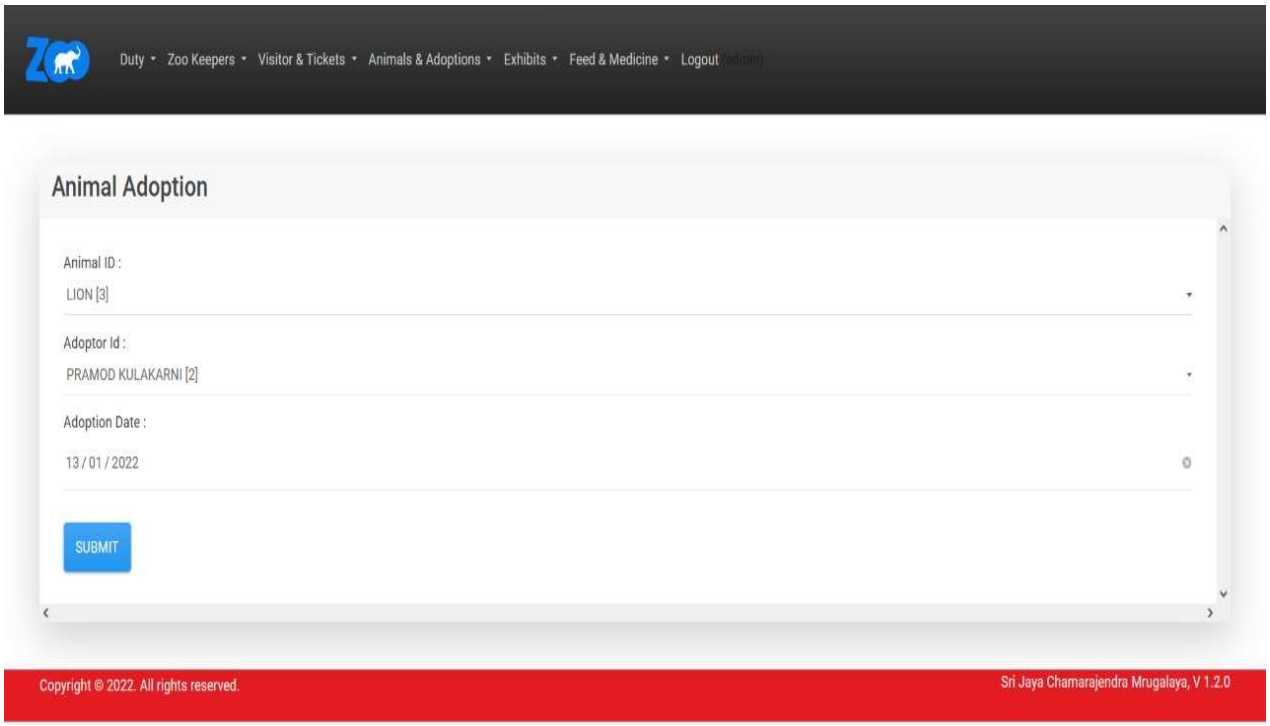
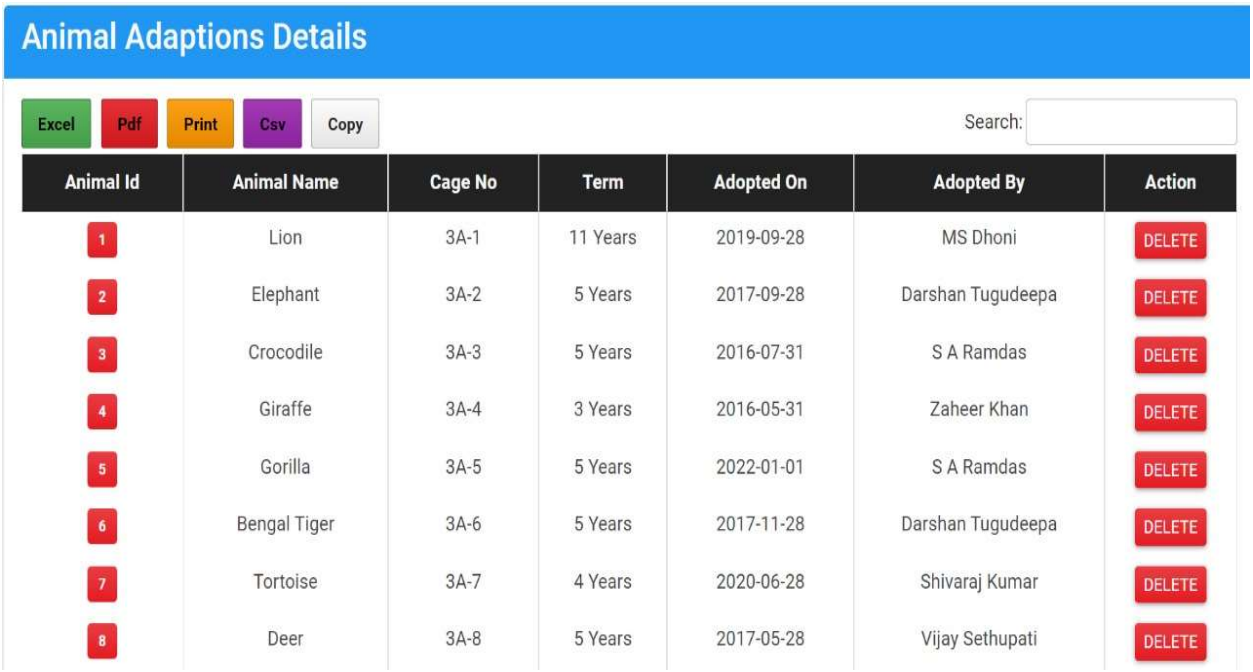


Figure 4.16: Animal Adoption Page



Animal Adaptions Details

Excel Pdf Print Csv Copy Search:

Animal Id	Animal Name	Cage No	Term	Adopted On	Adopted By	Action
1	Lion	3A-1	11 Years	2019-09-28	MS Dhoni	DELETE
2	Elephant	3A-2	5 Years	2017-09-28	Darshan Tugudeepa	DELETE
3	Crocodile	3A-3	5 Years	2016-07-31	S A Ramdas	DELETE
4	Giraffe	3A-4	3 Years	2016-05-31	Zaheer Khan	DELETE
5	Gorilla	3A-5	5 Years	2022-01-01	S A Ramdas	DELETE
6	Bengal Tiger	3A-6	5 Years	2017-11-28	Darshan Tugudeepa	DELETE
7	Tortoise	3A-7	4 Years	2020-06-28	Shivaraj Kumar	DELETE
8	Deer	3A-8	5 Years	2017-05-28	Vijay Sethupati	DELETE

Figure 4.17: Animal Adaption Details Page

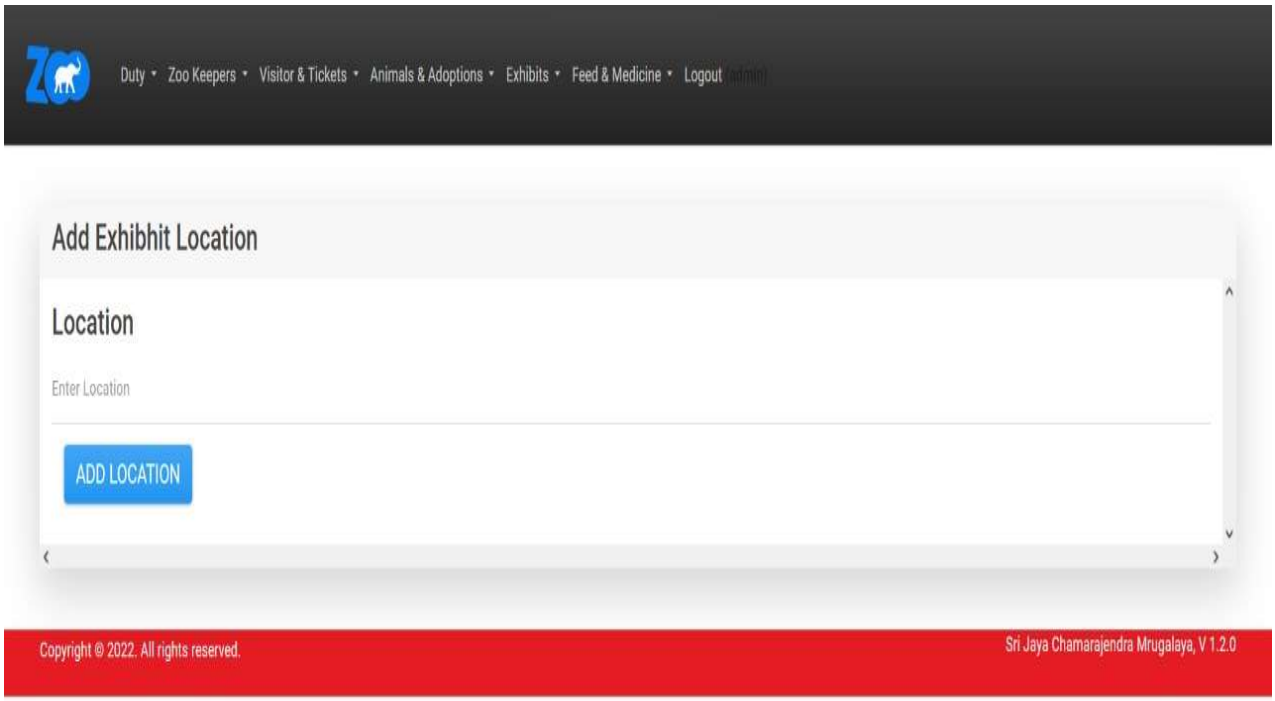


Figure 4.18: Add Exhibit Location Page

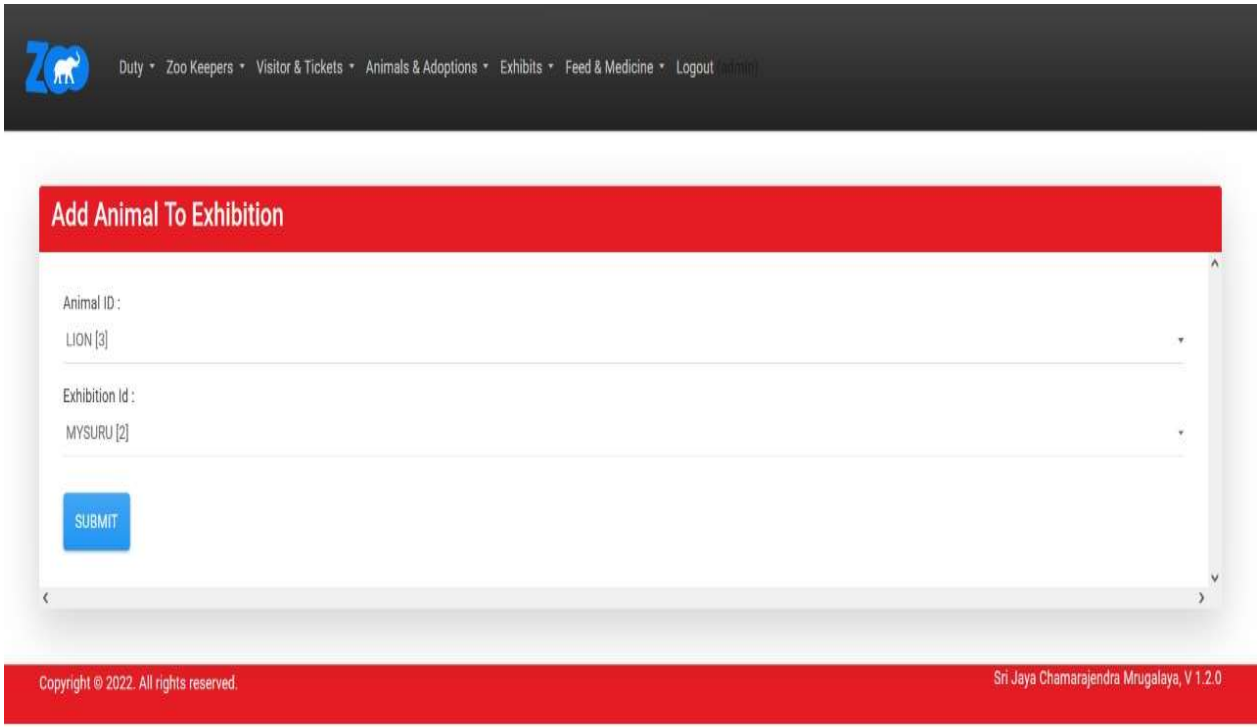


Figure 4.19: Add Animal to Exhibit Page

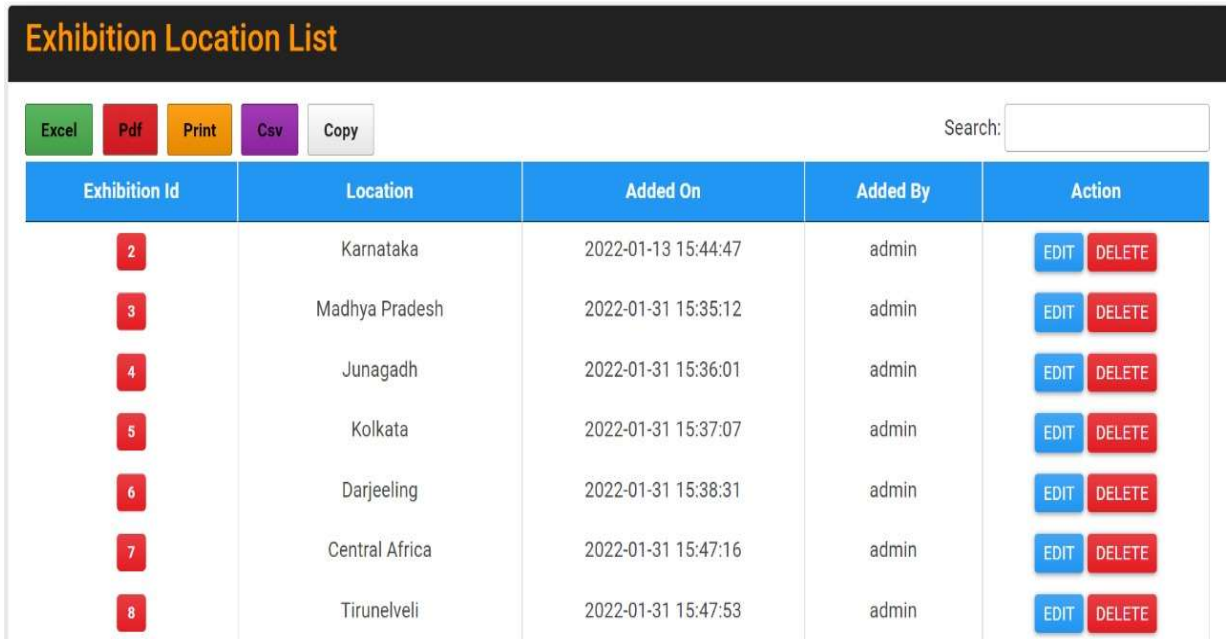


Figure 4.20: Exhibition Location List Page

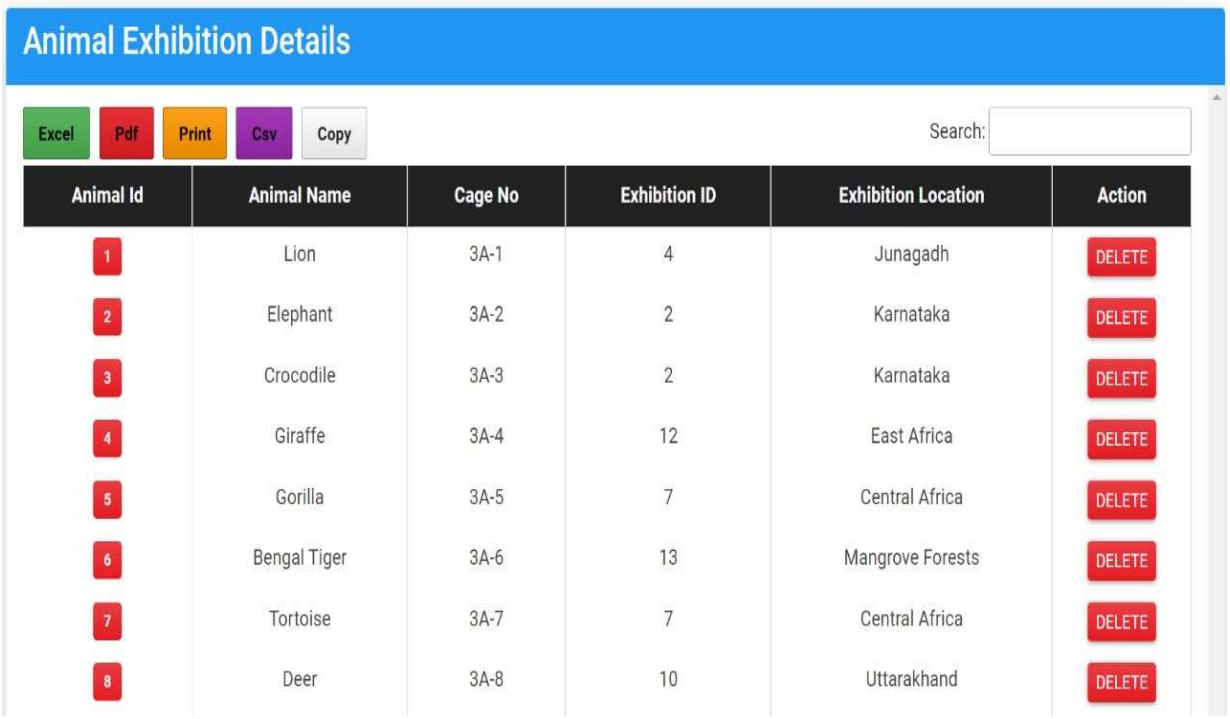


Figure 4.21: Animal Exhibition Details Page

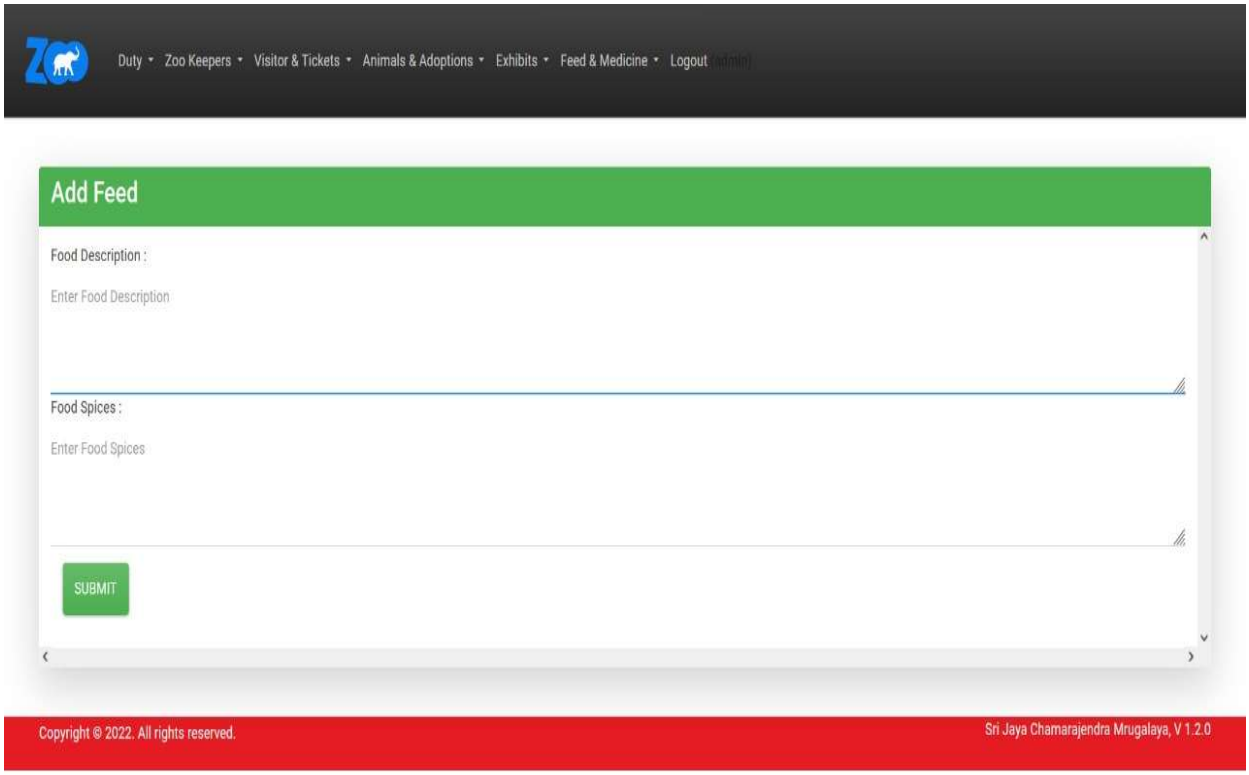


Figure 4.22: Add Feed Page

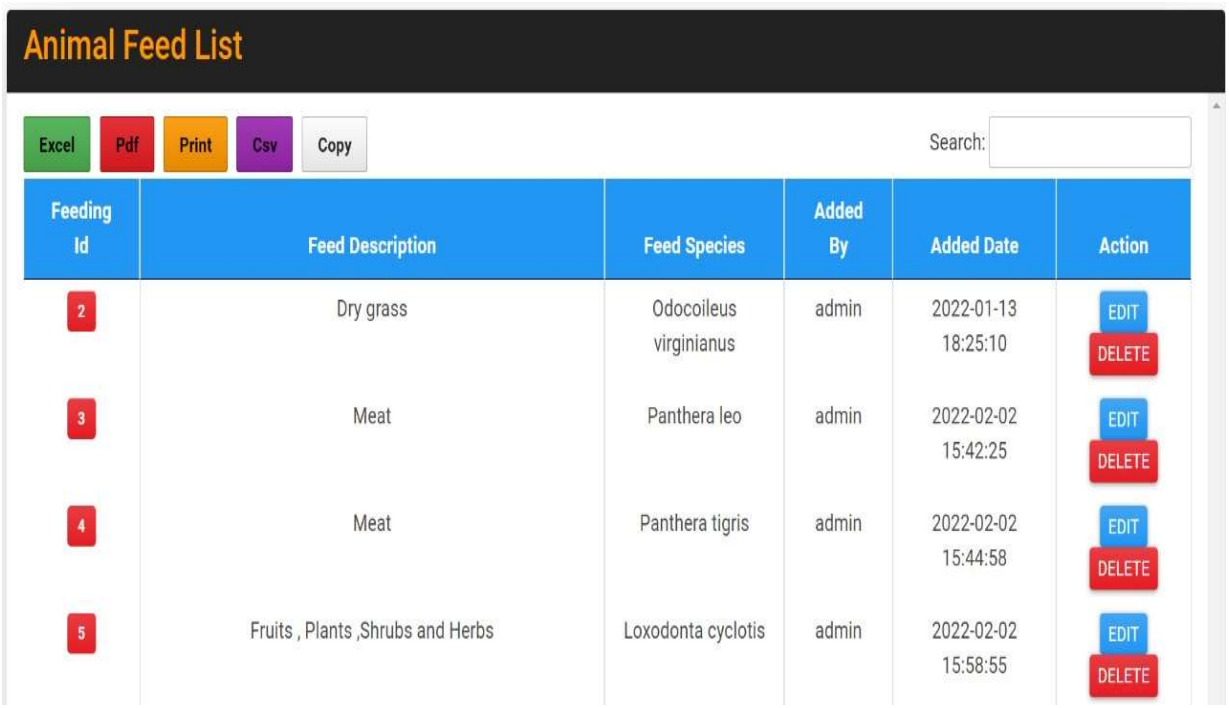


Figure 4.23: Animal Feed List Page

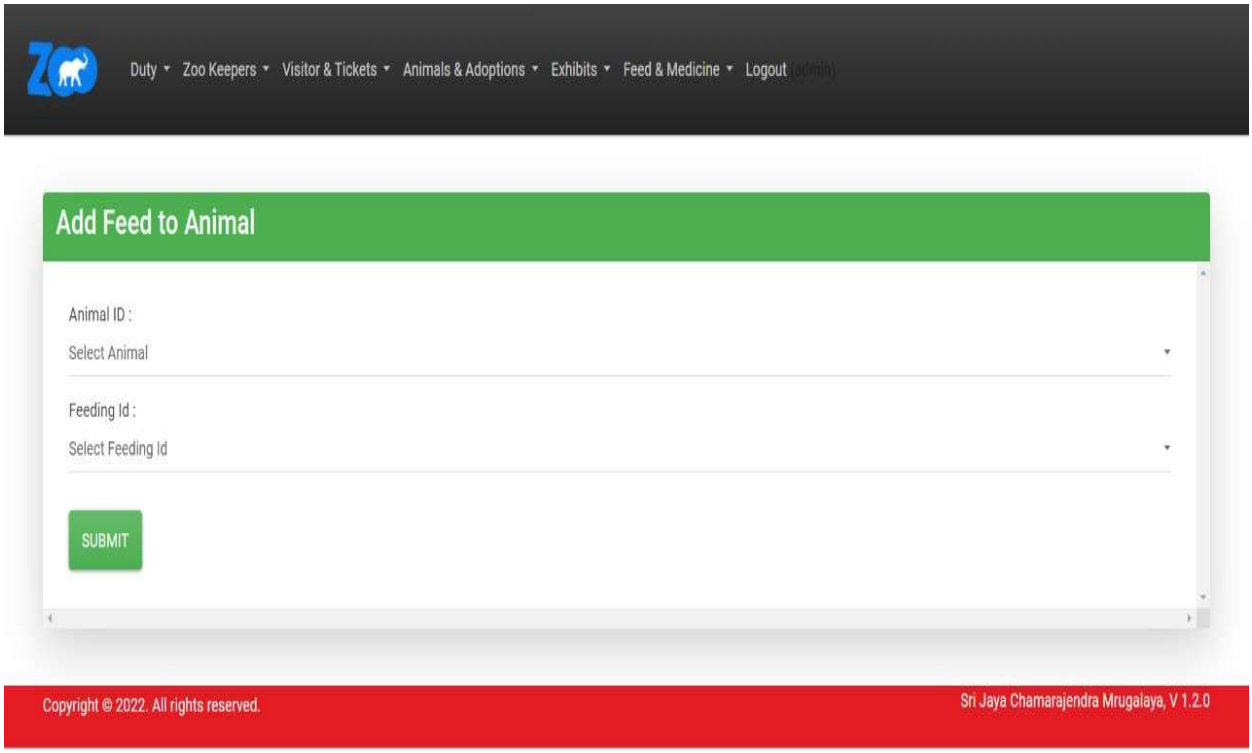


Figure 4.24: Add Feed to Animal Page

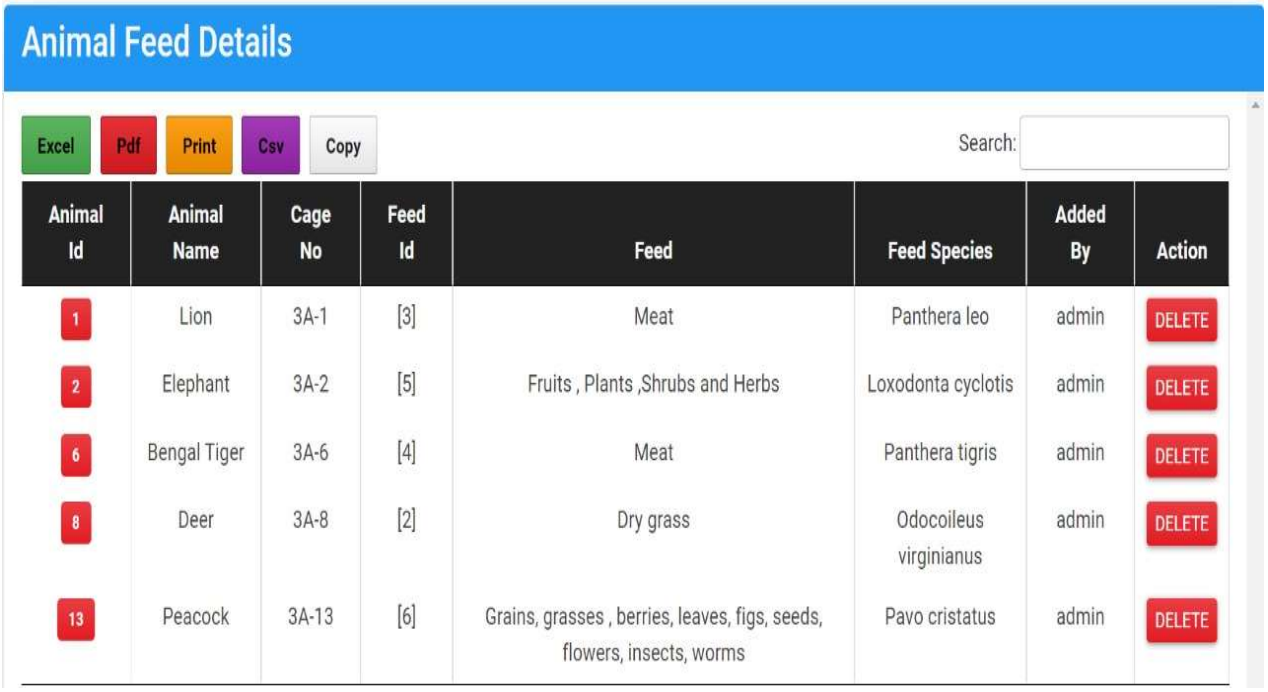


Figure 4.25: Animal Feed Details Page

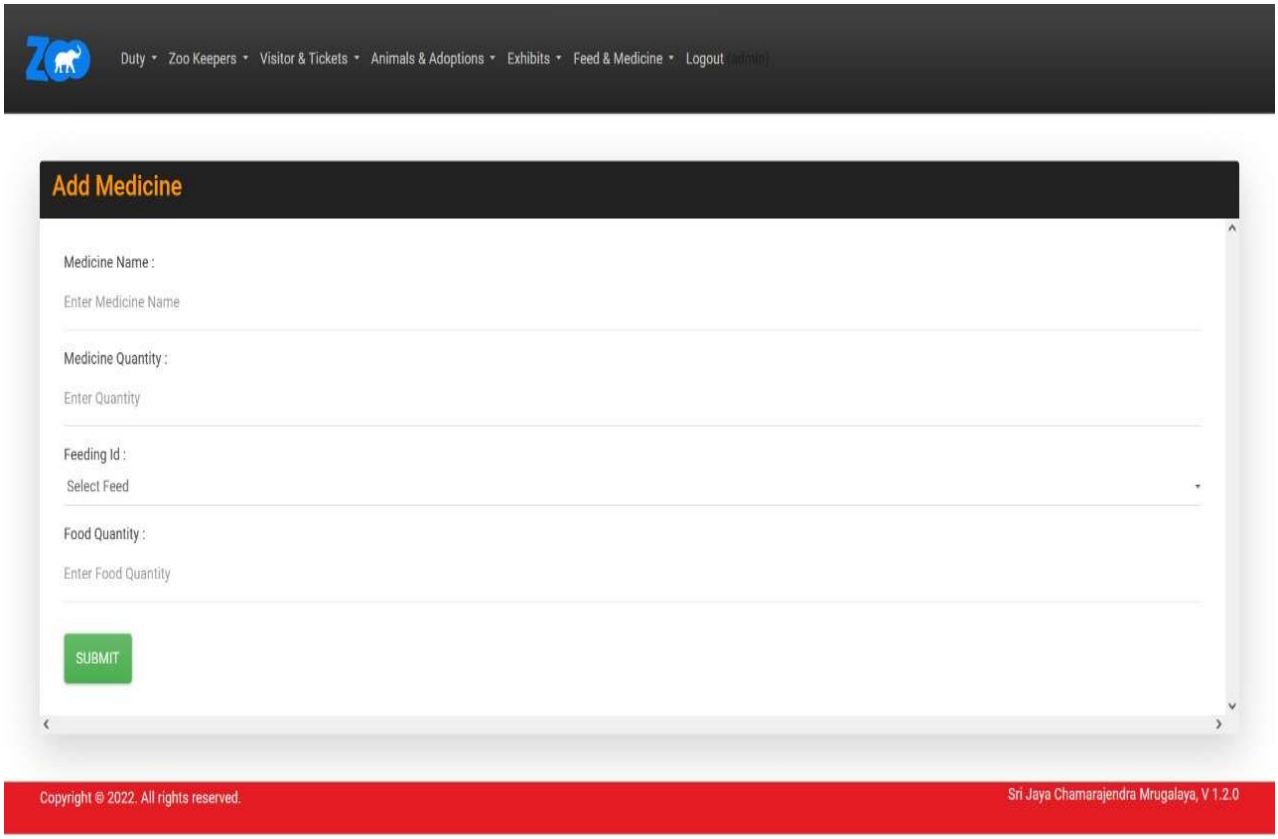


Figure 4.26: Add Medicine Page

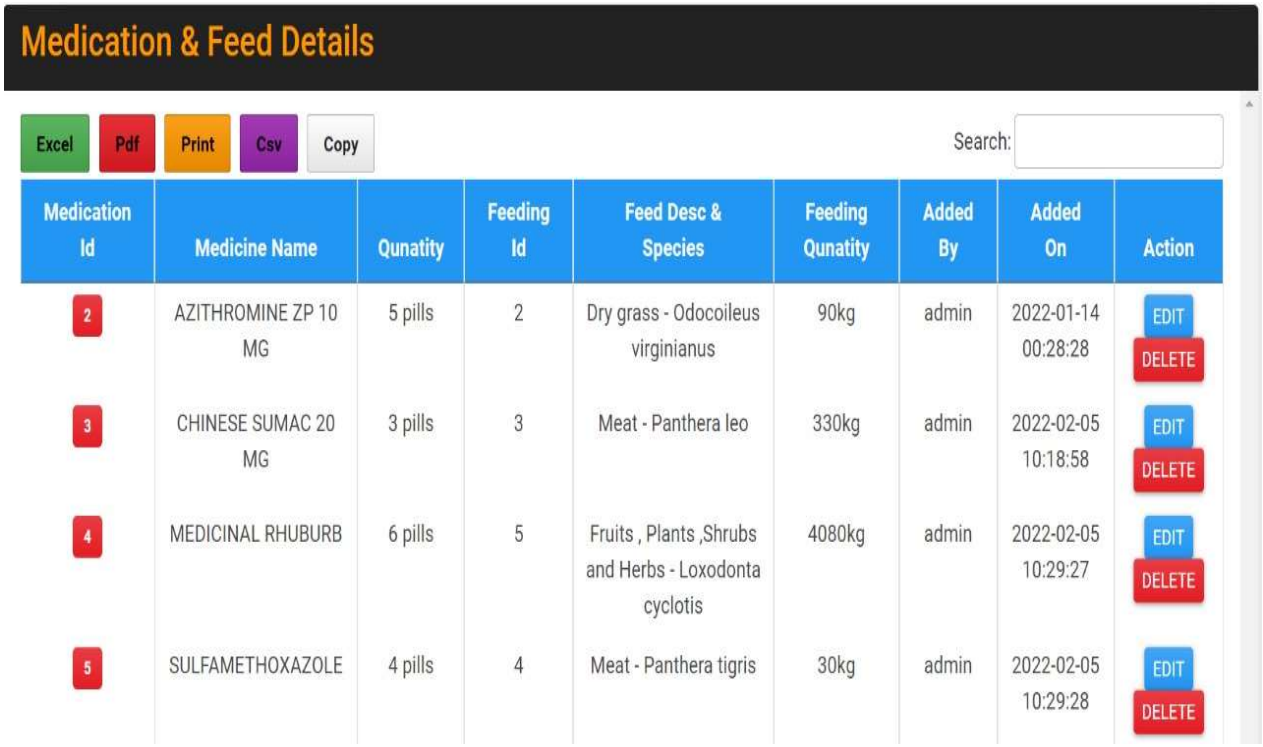


Figure 4.27: Medication and Feed Details Page

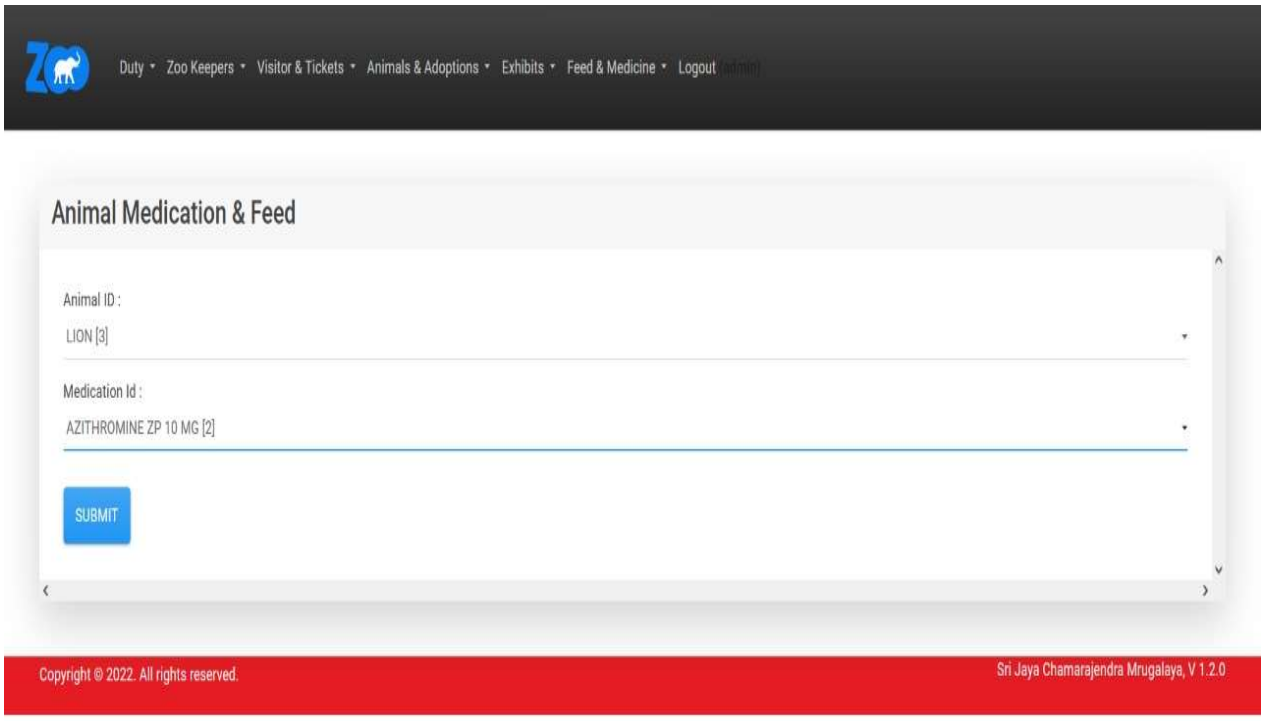


Figure 4.28: Animal Medication & Feed Page

Animal Medication Details

Excel Pdf Print Csv Copy Search:

Animal Id	Animal Name	Cage No	Medicine Id & Name	Dosage	Feed Id & Desc	Added By	Action
1	Lion	3A-1	[3] CHINESE SUMAC 20 MG	3 pills	[3] Meat	admin	DELETE
2	Elephant	3A-2	[6] SULFADIAZINE	4 pills	[4] Meat	admin	DELETE
6	Bengal Tiger	3A-6	[4] MEDICINAL RHUBURB	6 pills	[5] Fruits , Plants ,Shrubs and Herbs	admin	DELETE
8	Deer	3A-8	[2] AZITHROMINE ZP 10 MG	5 pills	[2] Dry grass	admin	DELETE
13	Peacock	3A-13	[5] SULFAMETHOXAZOLE	4 pills	[4] Meat	admin	DELETE

Figure 4.29: Animal Medication Details Page

Sri Jaya Chamarajendra Mrugalaya
Mysuru, Karnataka
080-0000000
sjcm@gmail.com
www.sjcm.mysuru.in

Visitor Ticket

Type	Price	Qnt	Sub Total
FOREIGN	300₹	5	1500₹
	Total		1500₹

Generated on: 2022-01-12 18:17:44

93457838

Thank you for your Visiting Sri Jaya Chamarajendra Mrugalaya!

Figure 4.30: Visitor Ticket

Conclusion and Future Enhancements

Conclusion

Zoo Database Management System(ZDMS) is constructed to store massive data related to animals and also produce reports according to the requirement. ZDMS successfully controls the data in a user-accessible manner. Furthermore permits the user to access, update and remove the data in a flexible mode.

ZDMS is a web based application which manages and handles the people ticket who visited in the zoo. Compared to the usual traditional method, queuing method, the web based management system could significantly increase visitor satisfaction and reduce total waiting time and efforts effectively of visitors as well as the employees.

Future Enhancement

The future scope of our project is vast and can be used in extensive ways:
Zoo Database Management System in streamlining all the activities of animals and time management of visitors ticket which will reduce time and give more profit. This will help to make data manipulation so easy.