# CMPG223 Project Documentation

# Members:

* Angelina Ramsunar –41081269
* Riya Patel-41914228
* Brian Anderson-40851842
* Christian Coetzee- 40513262

# Time Diary

|  |  |
| --- | --- |
| Angelina Ramsunar | Login Form, Physical Data Model, Databases |
| Riya Patel | HomePage Form, Physical Process Model |
| Brian Anderson | Staff Form(visitors, tickets), Receipt Form |
| Christian Coetzee | Admin Form(visitors, animals, reports) |

Each member spent about 4 hours each week for 3 weeks.

Which results in 12 hours for 3 weeks.

## Meetings

Meeting 1: Duration: 2 Hour

Place: Library

Discussion and the creation of GUI( blueprint)

Meeting 2 : Duration: 1 hour

Place: Library

Finalising documentation

Meeting 3 : Duration: 3 hours

Place: Library

Discussion and merging of the code.

# Screen print of GitHub

A screenshot of a computer

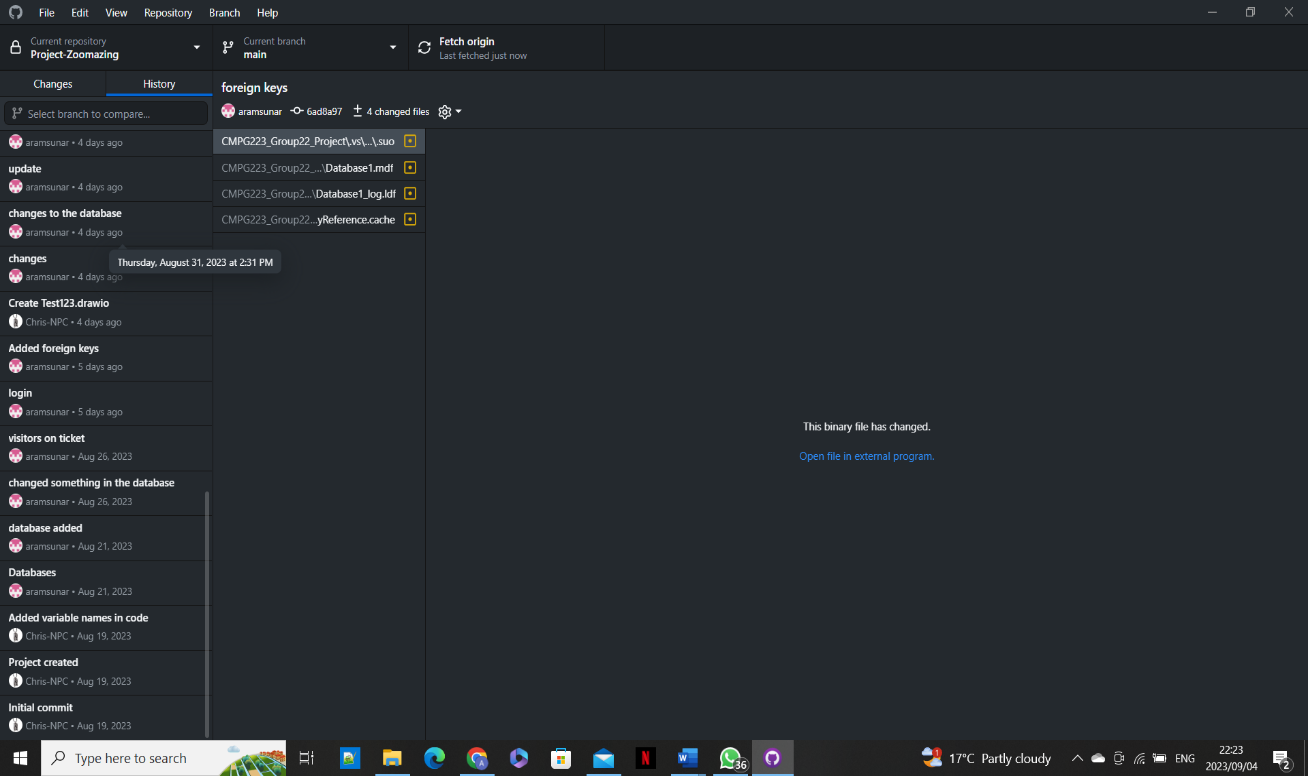
Description automatically generated

A screenshot of a computer

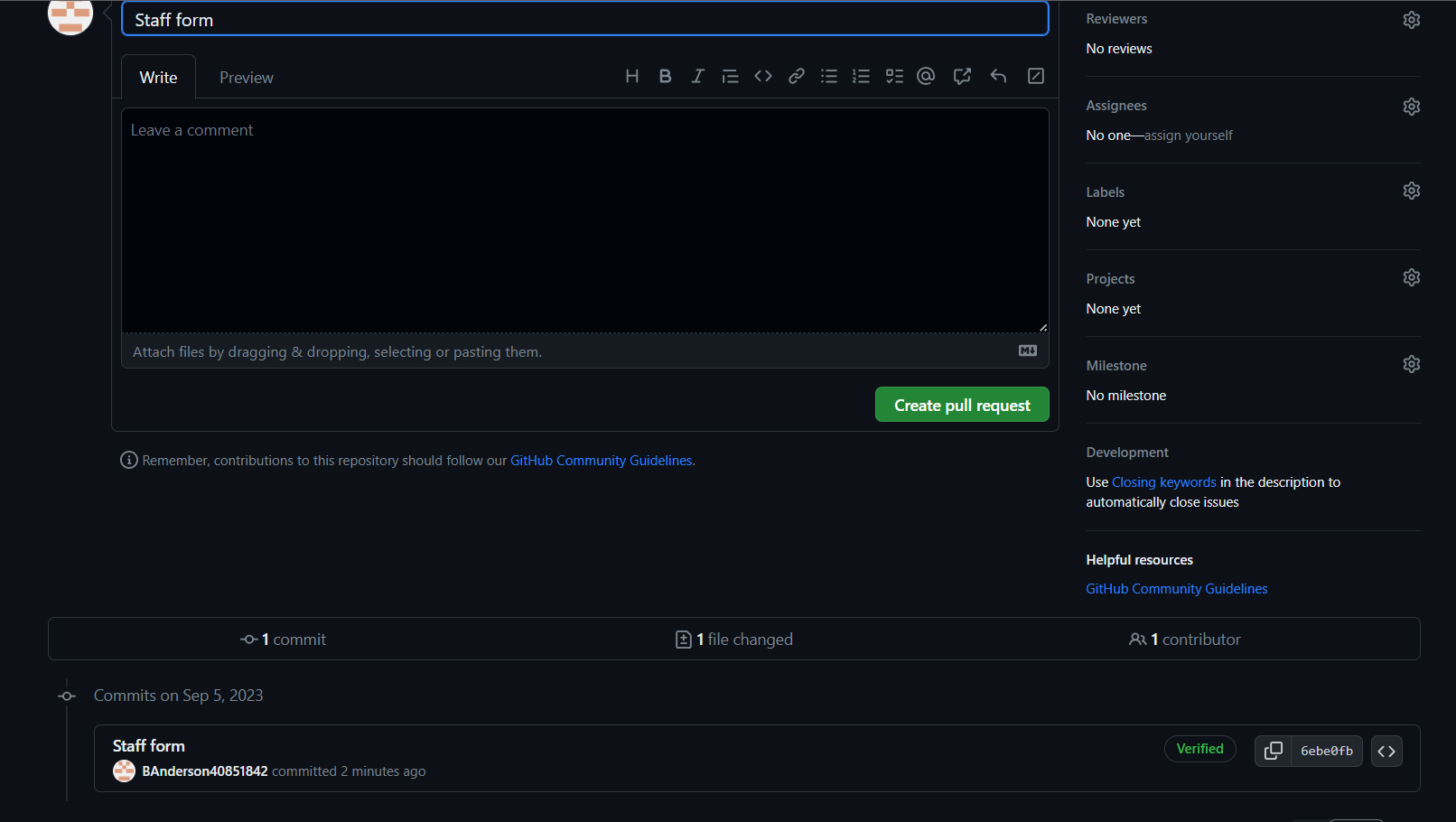
Description automatically generated

A screenshot of a computer

Description automatically generated



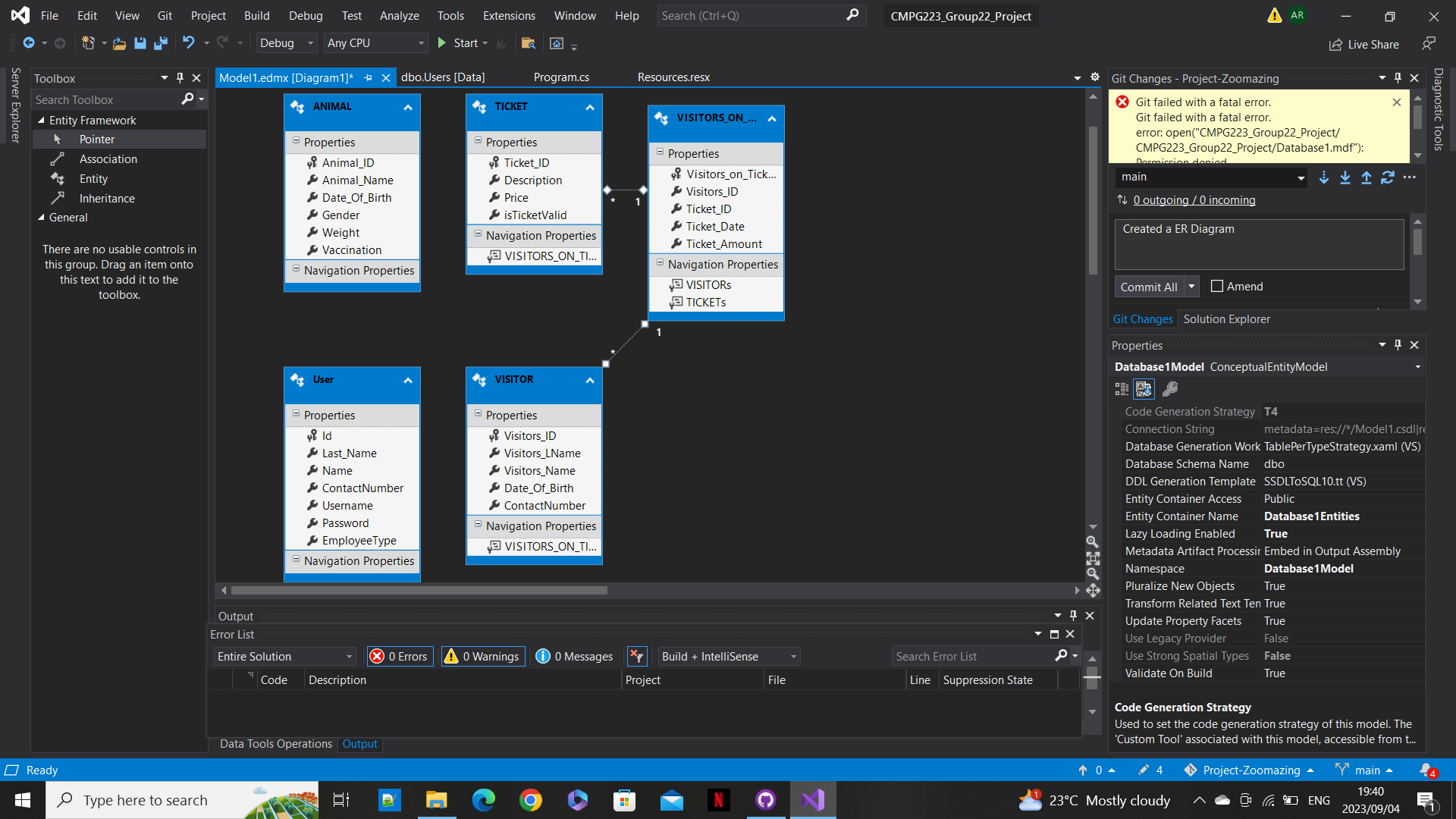


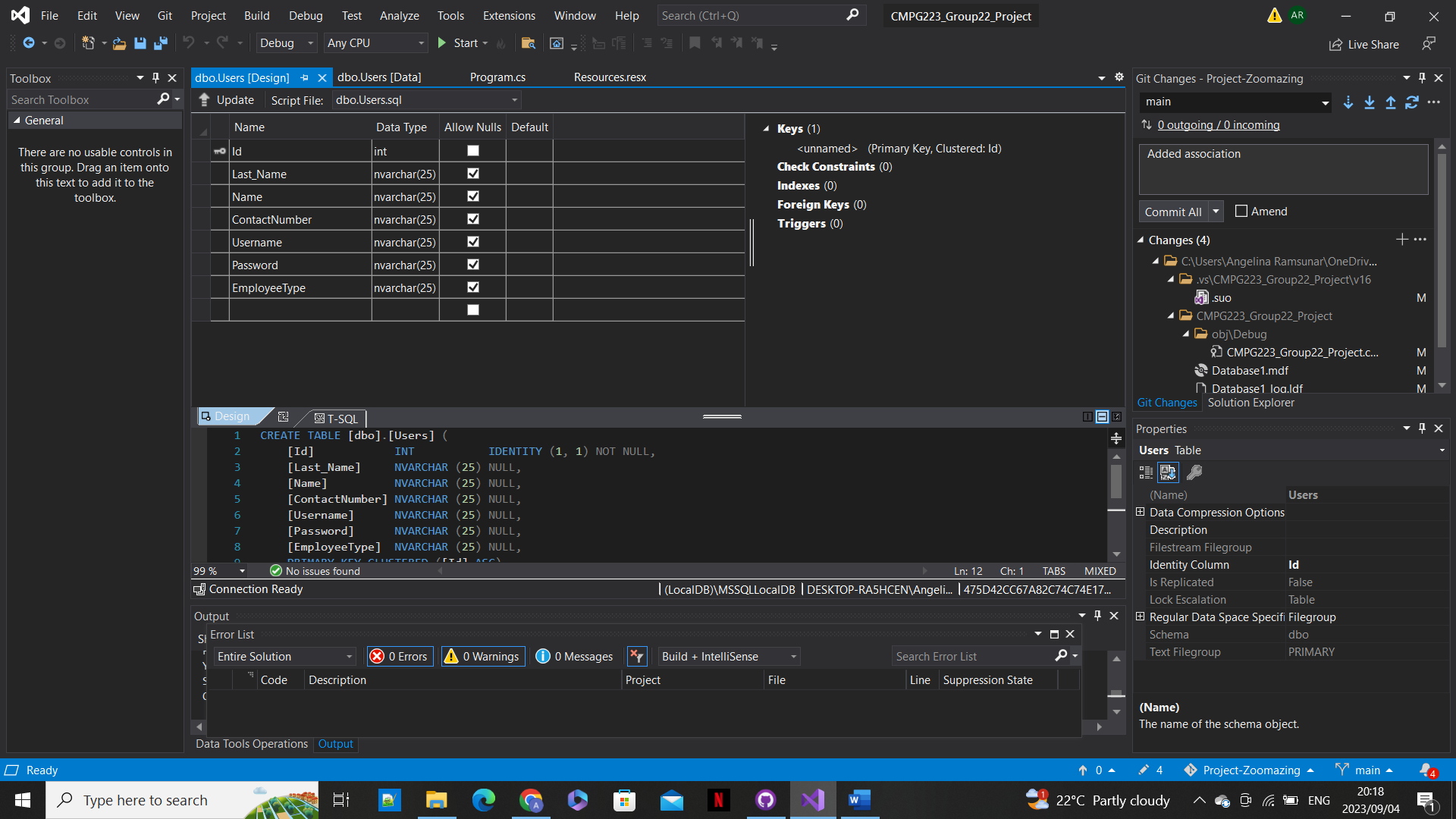


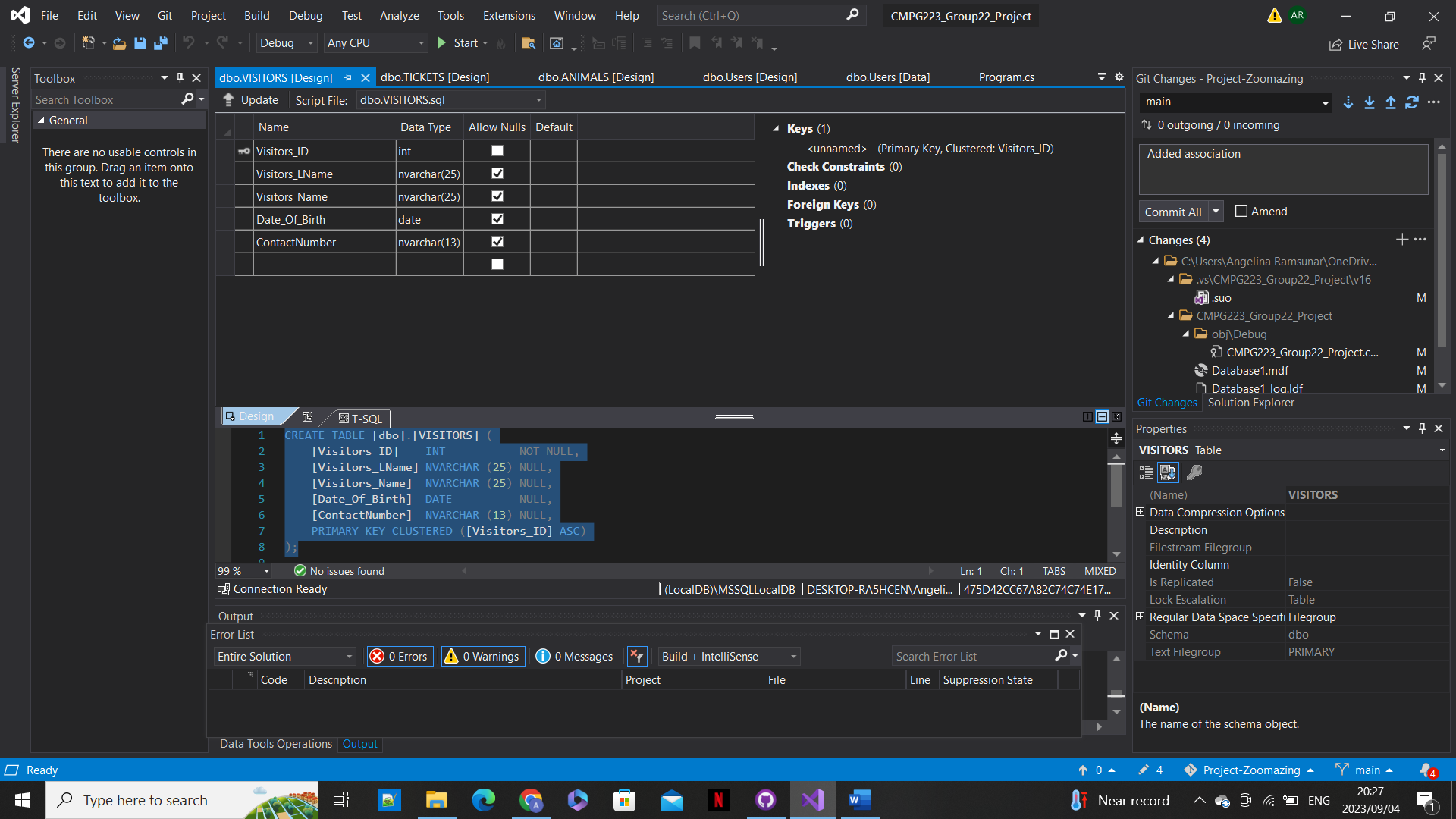
A screenshot of a computer

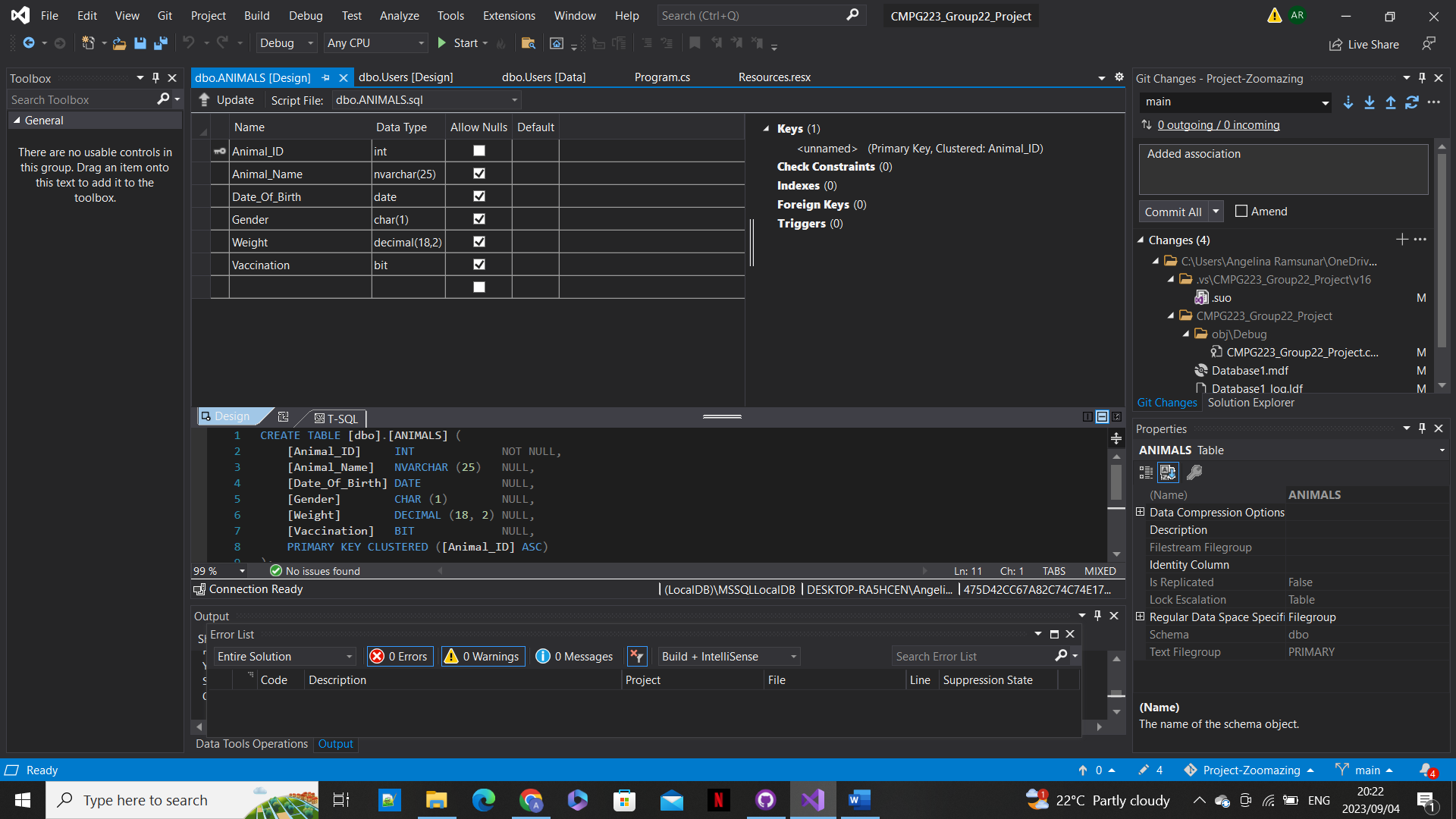
Description automatically generated

# Database schema and entity relation diagram in SQL server









A computer screen shot of a program

Description automatically generated

A computer screen shot of a program

Description automatically generated

# Code re-use

# 

# A computer screen shot of a computer code Description automatically generated

# SQL CODE

## Booking a ticket

-Show all visitors and tickets

string sql = "SELECT \* FROM VISITORS";

string sql = "SELECT \* FROM TICKETS";

## -Insert into tickets

String insert\_query = $"INSERT INTO VISITORS VALUES('{Surname}','{Name}','{DOB}','{ContactNo}') ";

String insert\_query = $"INSERT INTO TICKETS(Description,Price,isTicketValid) VALUES('{txtDesc.Text}',{lblShowPrice.Text},'{1}') ";

## Maintaining animals

## - Select:

string sql\_id = "SELECT Animal\_Id FROM ANIMALS";

string sql\_pop = "SELECT Animal\_Id FROM ANIMALS";

string sql = "SELECT \* FROM ANIMALS";

## - Add a new animal

string sql = $"INSERT INTO ANIMALS VALUES({animalId + 1}, '{getName()}', '{getDateOfBirth()}', '{getGender()}', {getWeight()}, '{getVacc()}')";

## - Update the animal :

string sql = $"UPDATE ANIMALS SET Animal\_Name = '{getName()}', Date\_Of\_Birth = '{getDateOfBirth()}', Gender = '{getGender()}', Weight = {getWeight()}, Vaccination = '{getVacc()}' WHERE Animal\_ID = '{animalId}'";

## - Delete an animal

\* string sql = $"DELETE FROM ANIMALS WHERE Animal\_ID = '{animalId}'";

## Maintaining visitors

## - Select:

string sql\_id = "SELECT Visitors\_Id FROM VISITORS";

string sql\_pop = "SELECT Visitors\_Id FROM VISITORS";

string sql = "SELECT \* FROM VISITORS";

## - Add new visitor:

string sql = $"INSERT INTO VISITORS VALUES({visitorId+1},'{vPerson.getSurname()}', '{vPerson.getName()}', '{getDateOfBirth()}', '{vPerson.getContactNum()}')";

## - Update the visitor details:

string sql = $"UPDATE VISITORS SET Visitors\_LName = '{vPerson.getSurname()}', Visitors\_Name = '{vPerson.getName()}', Date\_Of\_Birth = '{getDateOfBirth()}', ContactNumber = '{vPerson.getContactNum()}' WHERE Visitors\_ID = '{visitorId}'";

- Delete the visitor :

string sql = $"DELETE FROM VISITORS WHERE Visitors\_ID = '{visitorId}'";

## REQUEST REPORTS (TICKETS):

## - Select receive:

string sql\_receive = "SELECT isTicketValid FROM TICKETS";

string sql\_receive = "SELECT Price, isTicketValid FROM TICKETS";

Login

## -Increase count and select employee(Count, select)

string query = "SELECT COUNT (\*) FROM Users WHERE Username= @LoginName AND Password = @LoginPassword";

string userTypeQuery = "SELECT EmployeeType FROM Users WHERE Name = @LoginName";

## Creating database tables

CREATE TABLE [dbo]. [ANIMALS] (

[Animal\_ID] INT IDENTITY (1, 1) NOT NULL,

[Animal\_Name] NVARCHAR (25) NULL,

[Date\_Of\_Birth] DATE NULL,

[Gender] CHAR (1) NULL,

[Weight] DECIMAL (18, 2) NULL,

[Vaccination] BIT NULL,

PRIMARY KEY CLUSTERED ([Animal\_ID] ASC)

);

CREATE TABLE [dbo].[TICKETS] (

[Ticket\_ID] INT IDENTITY (1, 1) NOT NULL,

[Description] NVARCHAR (25) NULL,

[Price] INT NULL,

[isTicketValid] BIT NULL,

PRIMARY KEY CLUSTERED ([Ticket\_ID] ASC)

);

CREATE TABLE [dbo].[Users] (

[Id] INT IDENTITY (1, 1) NOT NULL,

[Last\_Name] NVARCHAR (25) NULL,

[Name] NVARCHAR (25) NULL,

[ContactNumber] NVARCHAR (25) NULL,

[Username] NVARCHAR (25) NULL,

[Password] NVARCHAR (25) NULL,

[EmployeeType] NVARCHAR (25) NULL,

PRIMARY KEY CLUSTERED ([Id] ASC)

);

CREATE TABLE [dbo].[VISITORS] (

[Visitors\_ID] INT IDENTITY (1, 1) NOT NULL,

[Visitors\_LName] NVARCHAR (25) NULL,

[Visitors\_Name] NVARCHAR (25) NULL,

[Date\_Of\_Birth] DATE NULL,

[ContactNumber] NVARCHAR (13) NULL,

PRIMARY KEY CLUSTERED ([Visitors\_ID] ASC)

);

CREATE TABLE [dbo].[VISITORS\_ON\_TICKET] (

[Visitors\_on\_Ticket\_ID] INT IDENTITY (1, 1) NOT NULL,

[Visitors\_ID] INT NULL,

[Ticket\_ID] INT NULL,

[Ticket\_Date] DATE NULL,

[Ticket\_Amount] INT NULL,

PRIMARY KEY CLUSTERED ([Visitors\_on\_Ticket\_ID] ASC),

FOREIGN KEY ([Ticket\_ID]) REFERENCES [dbo].[TICKETS] ([Ticket\_ID]),

FOREIGN KEY ([Visitors\_ID]) REFERENCES [dbo].[VISITORS] ([Visitors\_ID])

);

# Screen print of 2 reports:

A screenshot of a computer

Description automatically generated

# Project Scope

The system must satisfy the following functional requirements:

* **Maintenance of animals**
* **Maintenance of visitors**
* **Buying a ticket**
* **Entering the Zoo**
* **Exiting the Zoo**
* **Comprehensive reporting**, which includes tickets sold per time and number of tickets(?)

In addition, the system must satisfy the following **non-functional requirements**:

* The buying-a-ticket process should be fast and efficient for the customer. The ticket(s) should be sent to customer after being processed.
* The processed ticket(s) should not take longer than a minute to be sent to the customer.
* Only employees of Zoo-mazing should have access to the system. Thus, unique identifiers should be established and implemented (e.g., Usernames and Passwords)
* The system will have to types of users: the managers (database admin/ IT Manager) and the assistants. The managers would have full functionality/ control/ access of system whereas the assistants would only have limited functionality/ control/ access.
* Database hold data of various reports important to the business. Such as monthly costs, transactions, ticket sales per time frame.
* The system will feature a help function for any inconveniences.

# Zoo System User Manual

## Table of Contents:

1. Introduction

- Purpose of the Zoo System

- System Requirements

2. Installation

- Installing Visual Studio

3. Getting Started

- Logging In

- User Roles and Home Page

4. Staff Page

- Visitors and Tickets

- Booking Tickets

- Receipt

5. Admin Page

- Animals

- Visitors

- Reports

6. User Interface

- Cute Koala Theme

7. Contact Information

## 1. Introduction

## Purpose of the Zoo System

The Zoo System is an application designed for zoo administrators and staff to efficiently manage zoo operations. This manual guides you through using the system.

## System Requirements

To run the Zoo System, ensure that your computer or laptop meets the following minimum requirements:

- Operating System : Windows

- Memory (RAM) : At least 8GB DDR4 RAM

- Storage : HDD with 500GB or SSD with 256GB

## 2. Installation

## Installing Visual Studio

[How to install VISUAL STUDIO 2022 and C# on Windows | Mazen Labs - YouTube](https://www.youtube.com/watch?v=8sVKmwQ-ykg)

Visual Studio is the development environment used for the Zoo System. Follow the provided installation guide to set it up on your computer.

## 3. Getting Started

## Logging In

1. Open the Zoo System application.

2. Enter your credentials to log in as either admin or staff.

## User Roles and Home Page

- If you log in as admin , you will access the admin features.

- If you log in as staff , you will access the staff features.

## 4. Staff Page

## Visitors and Tickets

On the staff page, you can switch between two sections using the tab control: "Visitors" and "Tickets."

- Visitors : Add visitor information.

- Tickets : Book tickets for visitors, specifying the number of adults, children, and pensioners. The system calculates the price based on the chosen options.

## Booking Tickets

1. Select the number of adults, children, and pensioners.

2. Press the "Book Ticket" button.

3. You will be directed to a receipt form displaying visitor information and the total price.

## Receipt

The receipt form provides details of the booking, including visitor information and the total cost.

## 5. Admin Page

## Animals

- View Animals : Click on the "Animals" tab to access the animals page. Here, you can view information about zoo animals, such as their names, species, and habitats.

## Visitors

- Manage Visitors : Click on the "Visitors" tab to access the visitors page. Choose from actions like insert, delete, and update for visitor information. Perform the action by pressing the corresponding button.

## Reports

- Generate Reports : Click on the "Reports" tab to access the reports page. Here, you can view the number of tickets sold and the total ticket cost.

## 6. User Interface

The user interface of the Zoo System features a friendly design with a cute koala theme. The main page and forms are adorned with green elements to create an inviting atmosphere.

## 7. Contact Information

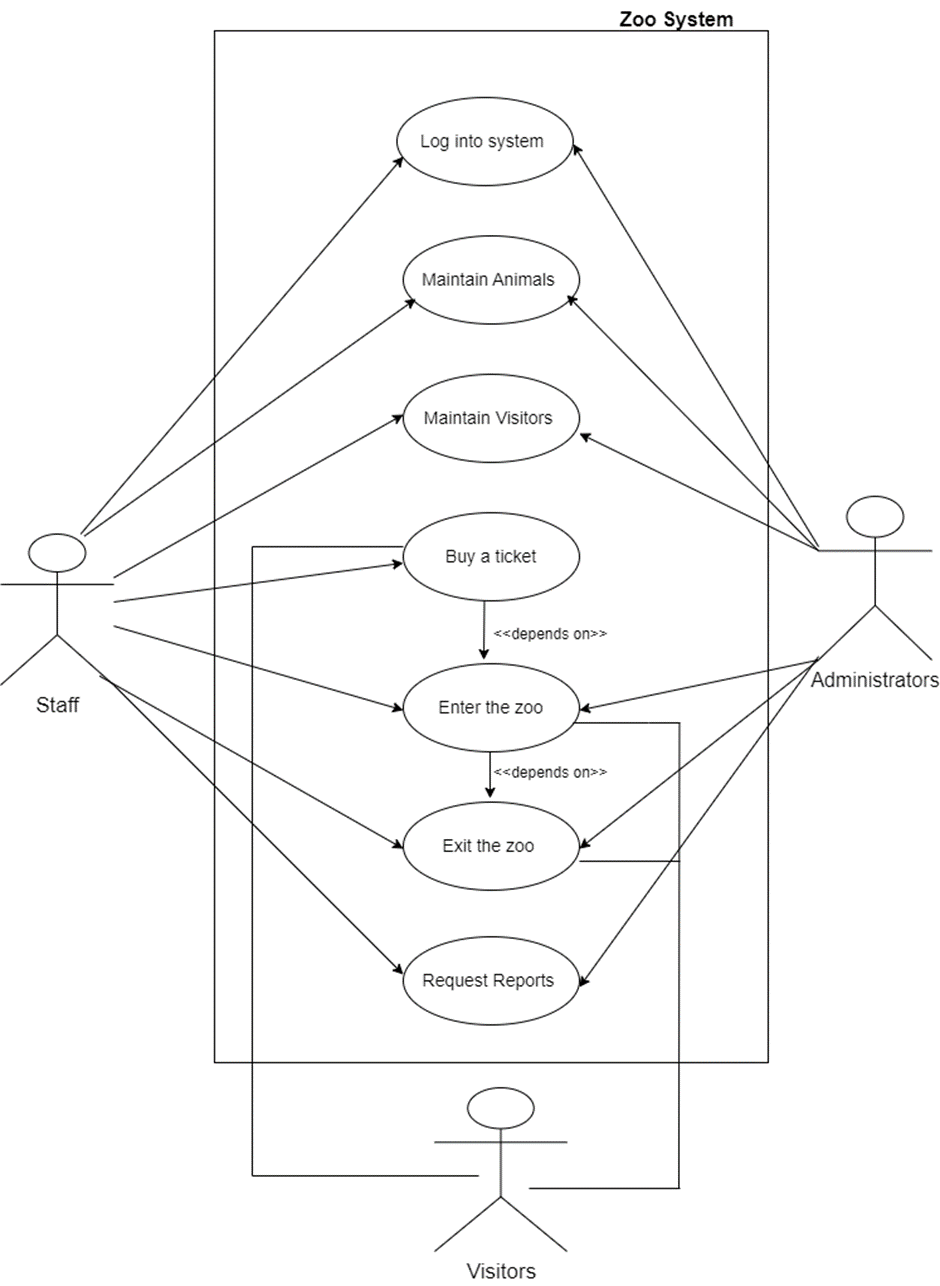
For any inquiries or assistance, please contact us:

- Email : zoo-mazing@zoomaze.co.za

- Main Office Contact Number : 018-254-6819

This user manual provides an overview of the Zoo System, from installation to using its core features. Please refer to the relevant sections for detailed instructions and guidance. If you encounter any issues or have questions, don't hesitate to reach out to our provided contact information. Enjoy using the Zoo System!

# Zoo-mazing Use Case



# Class Diagram

A diagram of a computer

Description automatically generated with medium confidence

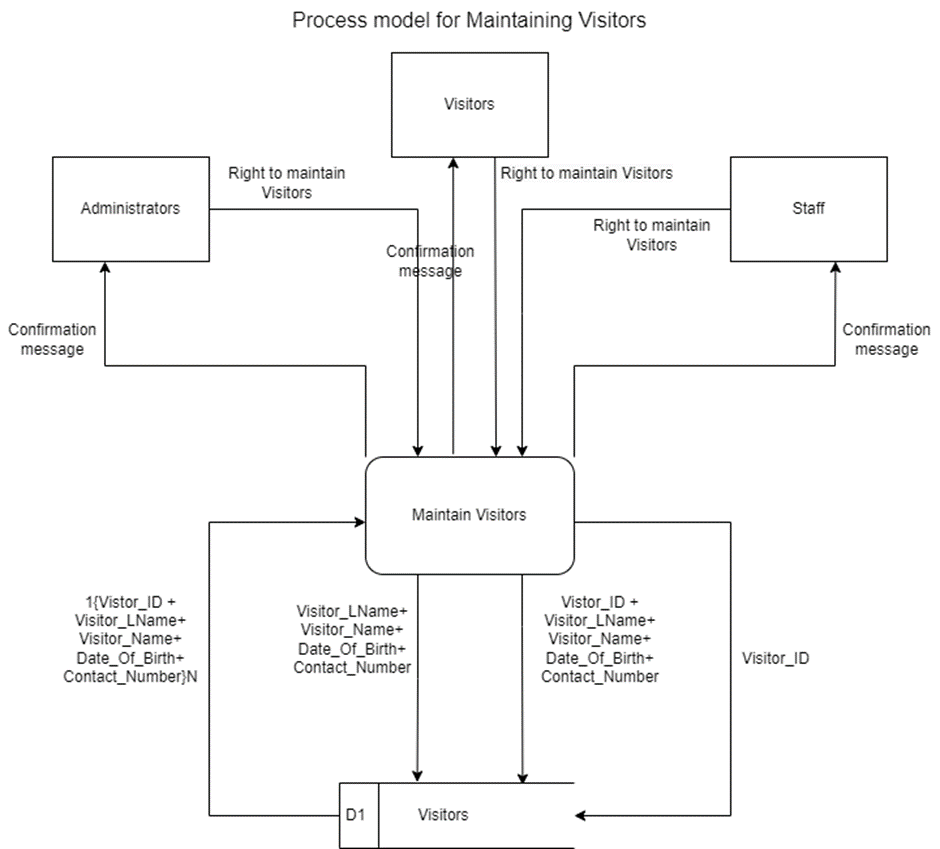
# Logical process models

A diagram of a system

Description automatically generated

A diagram of a process model

Description automatically generated



A screenshot of a computer screen

Description automatically generated

A diagram of a zoo

Description automatically generated

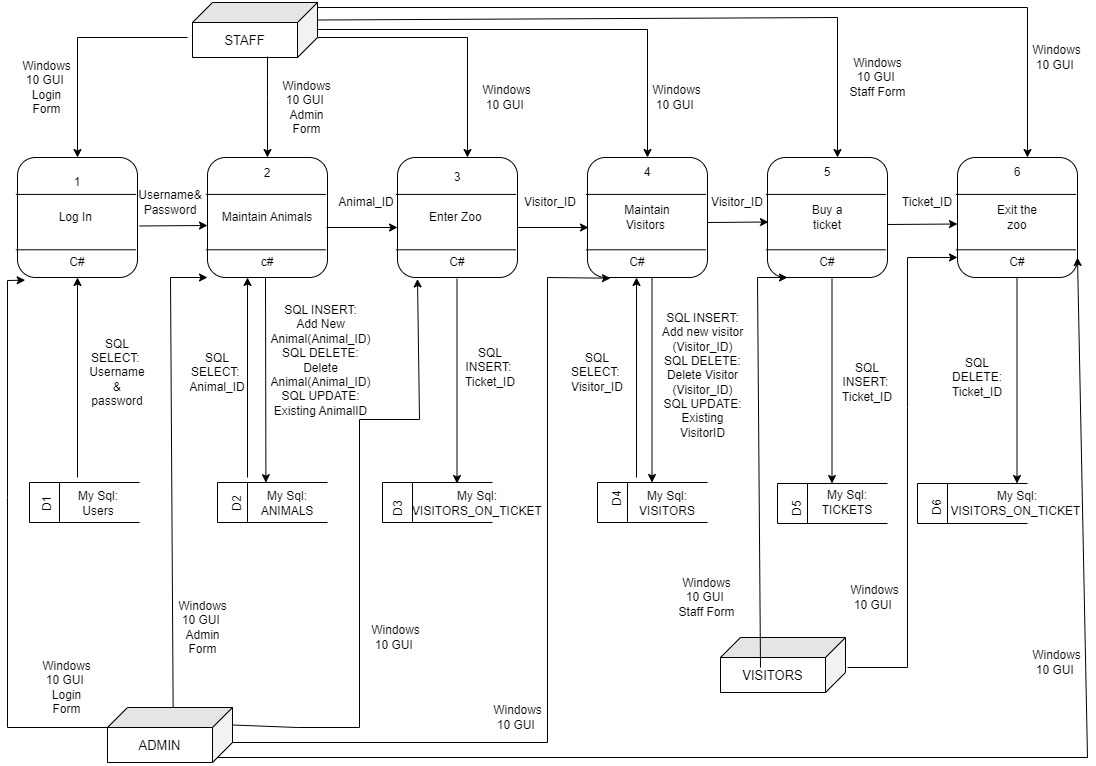
A screenshot of a computer screen

Description automatically generated

A diagram of a process

Description automatically generated

# Physical process model



# Logical Data Model

A screenshot of a computer screen

Description automatically generated

# Physical Data Model

A screenshot of a computer screen

Description automatically generated