

1. INTRODUCTION

1.1 OVERVIEW OF THE SYSTEM

Bird behaviour and populace patterns have become a significant issue now a days. Birds help us to recognize different life forms on the earth effectively as they react rapidly to ecological changes. Be that as it may, assembling and gathering data about bird species requires immense human exertion just as it turns into an extremely costly technique. In such a case, a solid framework that will give enormous scale preparation of data about birds and will fill in as a significant apparatus for scientists, legislative offices, and so forth is required. In this way, bird species distinguishing proof assumes a significant job in recognizing that a specific picture of birds has a place with which categories. Bird species identification means predicting the bird species belongs to which category by using an image.

The recognition of bird species can be possible through a picture, audio or video. Normally, people discover images more effectively than sounds or recordings. So, an approach to classify birds using an image over audio or video is preferred. Bird species identification is a challenging task to humans as well as to computational procedures that carry out such a task in an automated fashion.

In our world, there are above 9000 bird species. Some bird species are being found rarely and if found also prediction becomes very difficult. In order to overcome this problem, we have an effective and simple way to recognize these bird species based on their features. Also, the human ability to recognize the birds through the images is more understandable than audio recognition. So, we have used Convolutional Neural Networks (CNN). CNN's are the strong assemblage of machine learning which has proven efficient in image processing.

1.2 PROBLEM DEFINITION AND OBJECTIVES OF THE PROJECT

The everyday pace of life tends to be fast and frantic and involves extramural activities. Birdwatching is a recreational activity that can provide relaxation in daily life and promote resilience to face daily challenges. It can also offer health benefits and happiness derived from enjoying nature. Birdwatching is a common hobby but to identify their species requires the assistance of bird books. To provide birdwatchers with a handy tool to admire the beauty of birds, we developed a deep learning platform to assist users in recognizing species of birds using a mobile app. Bird images were learned by a convolutional neural network (CNN) to localize prominent features in the images. First, we established and generated a bounded region of interest to refine the shapes and colors of the object granularities and subsequently balanced the distribution of bird species. Then, a skip connection method was used to linearly combine outputs of the previous and current layers to improve feature extraction. Finally, we applied the softmax function to obtain a probability distribution of bird features. The learned parameters of bird features were used to identify pictures uploaded by mobile users. The proposed CNN model with skip connections achieved higher accuracy of 99.00% compared with the 93.98% from a CNN and 89.00% from the SVM for the training images. As for the test dataset, the average sensitivity, specificity, and accuracy were 93.79%, 96.11%, and 95.37%, respectively.

2. SYSTEM ANALYSIS

2.1 INTRODUCTION

System analysis or study is an important phase of any system development process. The system is studied to the minutest details and analyzed. The system analyst plays the role of an interrogator and dwells deep into the working of the present system. The system is viewed as a whole and the inputs to the system are identified. The outputs from the organization are traced to through the various processing that the input phases through in the organization. A detailed study of this process must be made by various techniques like interviews; questionnaires etc. the data collected by these sources must be scrutinized to arrive to a conclusion. The conclusion is an understanding of the system functions. This system is called existing system. Now the existing system is close study and problem solver tries to sort out difficulties that the enterprise faces.

2.2 IDENTIFICATION OF NEED

The main objective of the project is to identify the species of the birds by analysing an image. Some of the experts like ornithologists couldn't identify species of the bird correctly by looking at an image. Although bird classification can be done manually by domain experts, with growing amounts of data, this rapidly becomes a tedious and time-consuming process. So, by this model we can identify the species of the birds accurately and in less time.

2.3 EXISTING SYSTEM

To identify the bird species there are many websites that produce the results using different technologies. But the results are not accurate. For suppose if we will give an input in those websites and android applications it gives us multiple results instead of a single bird name. It shows us the all bird names which are having similar characteristics. So, we aimed to develop a project to produce better and accurate results. In order to achieve this, we have used Convolutional Neural Networks to classify the bird species.

2.3.1 LIMITATIONS OF THE EXISTING SYSTEM

- Less accuracy on image identification
- Gives us multiple results instead of a single bird name
- Some applications consume a large amount of storage space for the local dataset

Based on could rectify all the existing systems. For that discussions were carried out to choose the best package for developing a new system. The drawbacks and inadequacies of the existing system, the new system is designed which

2.4 PROPOSED SYSTEM

In order to overcome the problems of the existing system, we proposed a system that uses an effective and simple way to recognize these bird species based on their features using the Convolutional Neural Networks (CNN), which is more accurate and precise. The CNN's are the strong assemblage of machine learning which has proven efficient in image processing. Convolution neural network algorithm is a multilayer perceptron that is the special design for the identification of two-dimensional image information. After identifying the bird we provide brief information about the bird and more images. Also, we include sophisticated users who have advanced knowledge about birds like scientists, legislative offices, birders, etc. The common users can chat with sophisticated users, which can be useful for understanding.

2.4.1 ADVANTAGES OF THE PROPOSED SYSTEM

- Accurate and precise image identification using CNN
- Compact Application
- Chat system for information exchange and better understanding

2.5 FEASIBILITY ANALYSIS

All projects are feasible when given unlimited resources and infinite time. It is both necessary and prudent to evaluate the feasibility of a project at the earliest possible time. A feasible study is not warranted for the system in which economic justification is observed, technical risk is low, few legal problems are expected and no reasonable alternative exists. An estimate is made of whether the identified user needs may be satisfied using our recent software and hardware technologies. The study will decide if the proposed system will be cost-effective, from the business point of view and it can be developed in the existing budgetary. The feasibility study should be relatively sharp and quick. The gesture should inform the decision of whether to go ahead with a more detailed analysis. A feasibility study may be documented as a separate report to higher officials of the top-level management and can be included as appendices to the system specification. Feasibility and risk analysis are detailed in many worries. If there is project risk then the feasibility of producing the quality software is reduced. The study is done in three phases:

- Operational Feasibility
- Technical Feasibility
- Economical Feasibility
- Schedule Feasibility

2.5.1 OPERATIONAL FEASIBILITY

The purpose of the operational feasibility study is to determine whether the new system will be used if it is developed and installed. And whether there will be resistance from users that will undermine the possible application benefit. The first challenge was whether the system meets the organizational requirements. This is checked by the system requirement collected from the users and the management and the operational feasibility proved that the system is capable to meet its functional requirements. The developed system is completely driven and user-friendly. Operational feasibility is a measure of how the proposed system solves the problem and how it satisfies requirements identified in the requirement analysis phase of system development. The maintenance and working of the new system need less human effort.

2.5.2 TECHNICAL FEASIBILITY

The technical feasibility study is a study of function, performances, and constraints, and improves the ability to create an acceptable system. Technical feasibility is frequently the most difficult area to achieve at the stage of the product engineering process. Considering that are normally associated with the technical feasibility include:

- Development Phase
- Resource availability
- Technology

In the proposed system the technical feasibility study is conducted by considering the risk related to developing the system. The system must be evaluated from a technical viewpoint first. The assessment of this feasibility must be based on the outline design of the system requirements in the terms of inputs, outputs program procedure, and staff. This new system is said to be technically feasible. Technical feasibility centers on the existing computer systems and extends to which it can support the proposed system. This involves financial consideration to technical enhancements.

2.5.3 ECONOMIC FEASIBILITY

Economic Feasibility is the most frequently used method for evaluating the effectiveness of the proposed system. It evaluates whether the system benefits greater than cost. The proposed system is an effective one since the benefits of the software outweigh the cost incurred in installing it. It can be developed under optimal expenses with the available hardware and software. This analysis is the most frequently used method for comparing the cost with benefit or income that is expected from the developed system. The development of this application is highly economically feasible. The only thing to be done is to make an environment with effective supervision. It is cost-effective in the sense that has eliminated the paperwork. The system is also time effective because the calculations are automated which are made at the end of the month or as per the user requirement

2.5.4 SCHEDULE FEASIBILITY

Schedule feasibility is a measure of how reasonable is the project time table. Each phase of the system development should be done within a specific time frame to avoid delays in the implementation of testing of the final product.

2.6 SYSTEM SPECIFICATION

After the analyst has collected all required information regarding the software to be developed, and has removed all completeness, inconsistency, and anomalies from specification, he starts to systematically organize the requirements the form of an SRS document. The software developers refer to the SRS document to make sure that they developed exactly what the customer requires. The SRS document helps the maintenance engineers to understand the functionality of the system.

2.6.1 SOFTWARE REQUIREMENTS

A software requirement specification (SRS), a requirements specification for a software system, is a complete description of the behavior of a system to be developed and may include a set of use cases that describe interactions the users will have with the software. In addition, it also contains non-functional requirements. Non-functional requirements impose constraints on the design or implementation (such as performance engineering requirements, quality standards, or design constraints) the software requirements specification document enlists all requirements that are required for the project development. To derive the requirements we need to have a clear and thorough understanding of the products to be developed. This is prepared after detailed communications with the project team and customer.

- Operating System: WINDOWS 8 or above for better performance
- Front end: Python (For web application), Android (Mobile Application)
- Back end: MYSQL
- Software: Sublime Text, WAMP, Android Studio
- Web Browser: Internet Explorer/Google Chrome/Firefox
- Web Server: Apache

2.6.2 HARDWARE REQUIREMENTS

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware. A hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in

the case of operating systems. An HCL lists tested compatible, and sometimes incompatible hardware devices for a particular operating system or application.

- Processor: Intel Pentium or above.
- Hard Disc: 320GB.
- Display Type: PC Display.
- Keyboard: PC/AT Enhanced PS/2Keyboard (110/10Key).
- Mouse: First/Pilot Mouse Serial (c48).
- Input Device: Mouse, keyboard
- Output Device: Monitor, Mobile Display

3.SYSTEM DESIGN

3.1 INTRODUCTION

System design is an interactive process through which requirements are transmitted to a “blue print” for constructing the software initial; the blue print depicts a holistic view of software that is design is represented at a high-level abstraction a level that can be directly traced to specific data, functional and behavioural requirement. As design interaction occurs subsequent refinement leads to design representation at much lower level of abstraction. System design is a creative art of inventing and developing input, data bases, off line files, method and procedures, for processing data to get meaning full output that satisfy the organization objectives. Through the design phase consideration to the human factor, that is inputs to the users will have on the system.

Some of the main factors that have to be noted using the design of the system are:

➤ **Practicability**

The system must be capable of being operated over a long period and must have ease of use.

➤ **Efficiency**

Make better use of available resources. Efficiency involves accuracy, timeliness, and comprehensiveness of system output.

➤ **Cost**

Aim of minimum cost and better results.

➤ **Security**

Ensure physical security of data

3.2 INPUT DESIGN

Input design is the process of converting user-oriented input to a based format. Inaccurate input data are the most common cause of errors in data processing. Errors entered by data entry operators can be controlled by input design.

The goal of designing input data is to make data entry as easy, logical, and free from errors. When we approach input data design; we design the data source documents that capture the data and then select the media used to enter them into the computer.

User-friendly screen format can reduce the burden on end-users, who are not highly proficient in computers. An important step in the input design stage is the design of the source document. The source document is the form in which the data can initially capture. The next step is the

design of the document layout. The layout organizes the document by placing information, where it will be noticed and establishes the appropriate sequence of items.

3.3 OUTPUT DESIGN

Computer output is the most important and direct source of information to the user. Efficient and intelligent output design improves the system's relationship and helps user decision-making.

In the output design, it is determined how the implementation is to be played for immediate need and also the hardcopy output. A major form of input is a hard copy from the printer. Printouts should be designed around the output requirement of the user. Printers, CRT screen displays are examples for providing computer-based output. The output design associated with the system includes the various reports of the table generations and query executions.

Output design is one of the, most important features of the information system. The logical design of an information system is analogous to an engineering blueprint of an automobile. It shows the major features and how they are related to one another. The outputs, inputs, and databases are designed are in this phase

3.4 DATABASE DESIGN

Database design, The most important part of the system design phase. In a database environment, data available are used by several users instead of each program managing its data, authorized users share data across the application with the database software managing the data as an entity.

Primary key is one of the candidate keys that are chosen to be the identifying key for the entire table.

Normalization is a process of organizing the data in the database to avoid data redundancy, insertion anomaly, update anomaly & deletion anomaly.

Here are the most commonly used normal forms:

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

FIRST NORMAL FORM (1NF)

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

It should hold only atomic values.

SECOND NORMAL FORM (2NF)

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

- The table is in 1NF (First normal form)
- No non-prime attribute is dependent on the proper subset of any candidate key of the table.

An attribute that is not part of any candidate key is known as a non-prime attribute.

THIRD NORMALFORM (3NF)

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

- The table must be in 2NF
- Transitive functional dependency of non-prime attribute on any super key should be removed.

An attribute that is not part of any candidate key is known as a non-prime attribute.

In other words, 3NF can be explained like this: A table is in 3NF if it is in 2NF, and for each functional dependency $X \rightarrow Y$ at least one of the following conditions hold:

- X is a super key of the table
- Y is a prime attribute of the table

An attribute that is a part of one of the candidate keys is known as a prime attribute.

3.5 DATA FLOW DIAGRAM

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system without going into great detail, which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design). A DFD shows what kind of information will be input to and output from the system, how the data will advance through the system, and where the data will be stored. It does not show information about process timing or whether processes will operate in sequence or parallel, unlike a traditionally structured flowchart that focuses on control flow, or a UML activity workflow diagram, which presents both control and data, flows as a unified model.

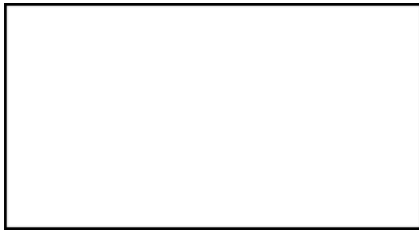
- RULES FOR DRAWING DATA FLOW DIAGRAM

- Each data store should have at least one data flow in and one data flow out.
- Data stored in a system must go through a process.
- All processes in a DFD go to another process or a data store.

BASIC DFD SYMBOLS

The primitive symbols used in the DFD are:

- External entity

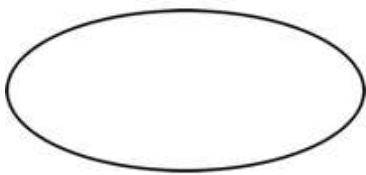


The external entities are essentially those physical entities external to the software system which

interact with the system by inputting data to the system or by consuming the data produced by the system. For example, the user of a system, Entities supplying data are known as sources and

those that consume data are sinks.

- Process



The functions are using circles. Bubbles are annotated with the names of the corresponding functions. They convert data into information.

- Data Flow



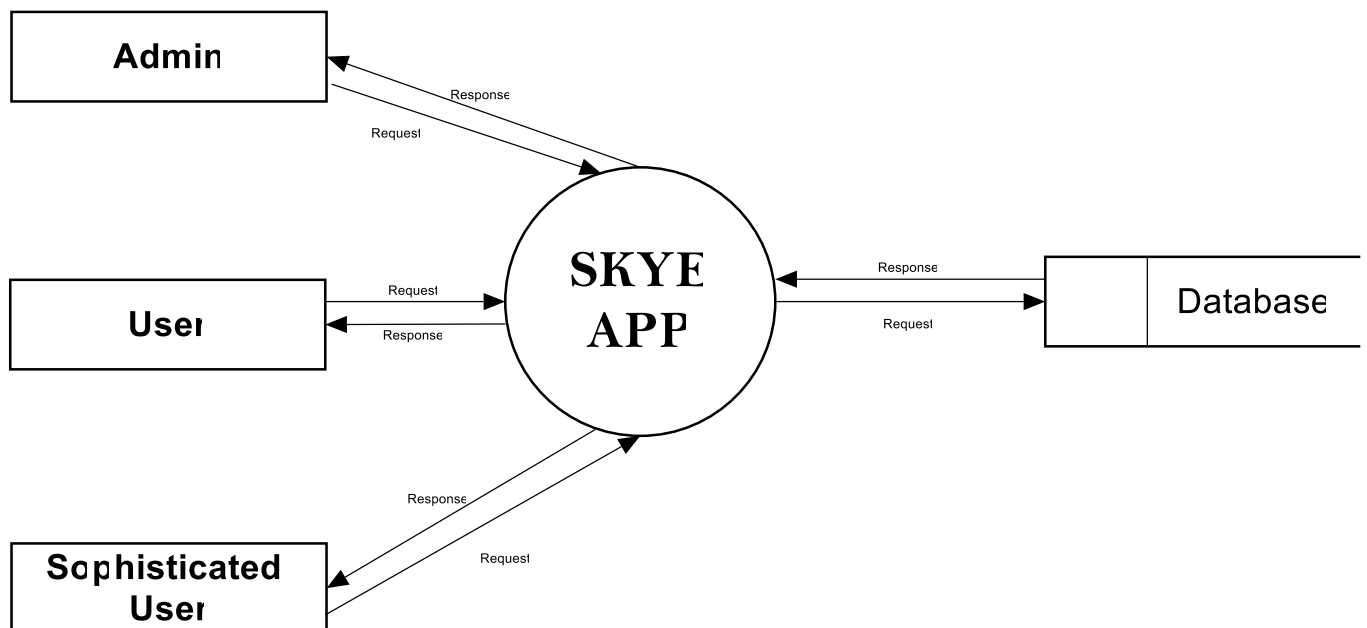
A directed arrow or arc is used as a data flow symbol that represents the data flow occurring between two processes, or between an external entity and a process.

➤ Data Store

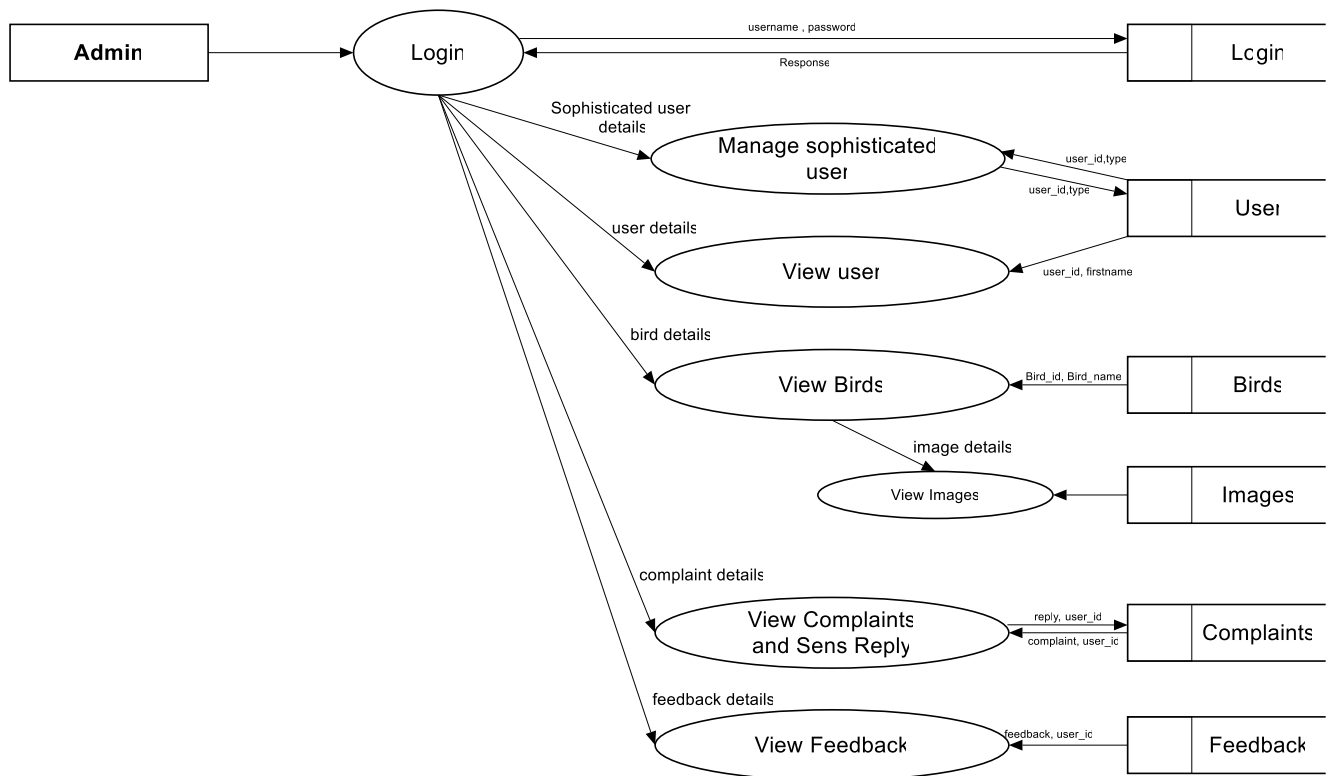


A data store represents logical files. Each datastore is connected to a process using a data flow symbol.

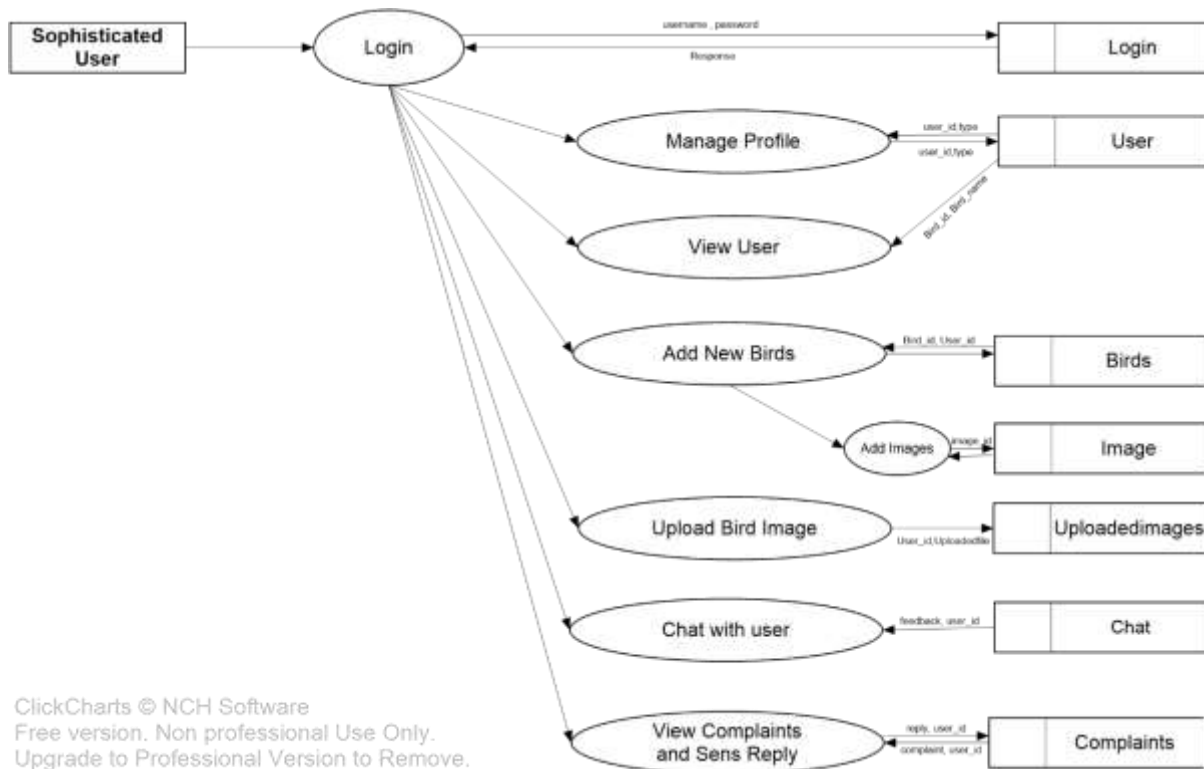
LEVEL 0 DFD



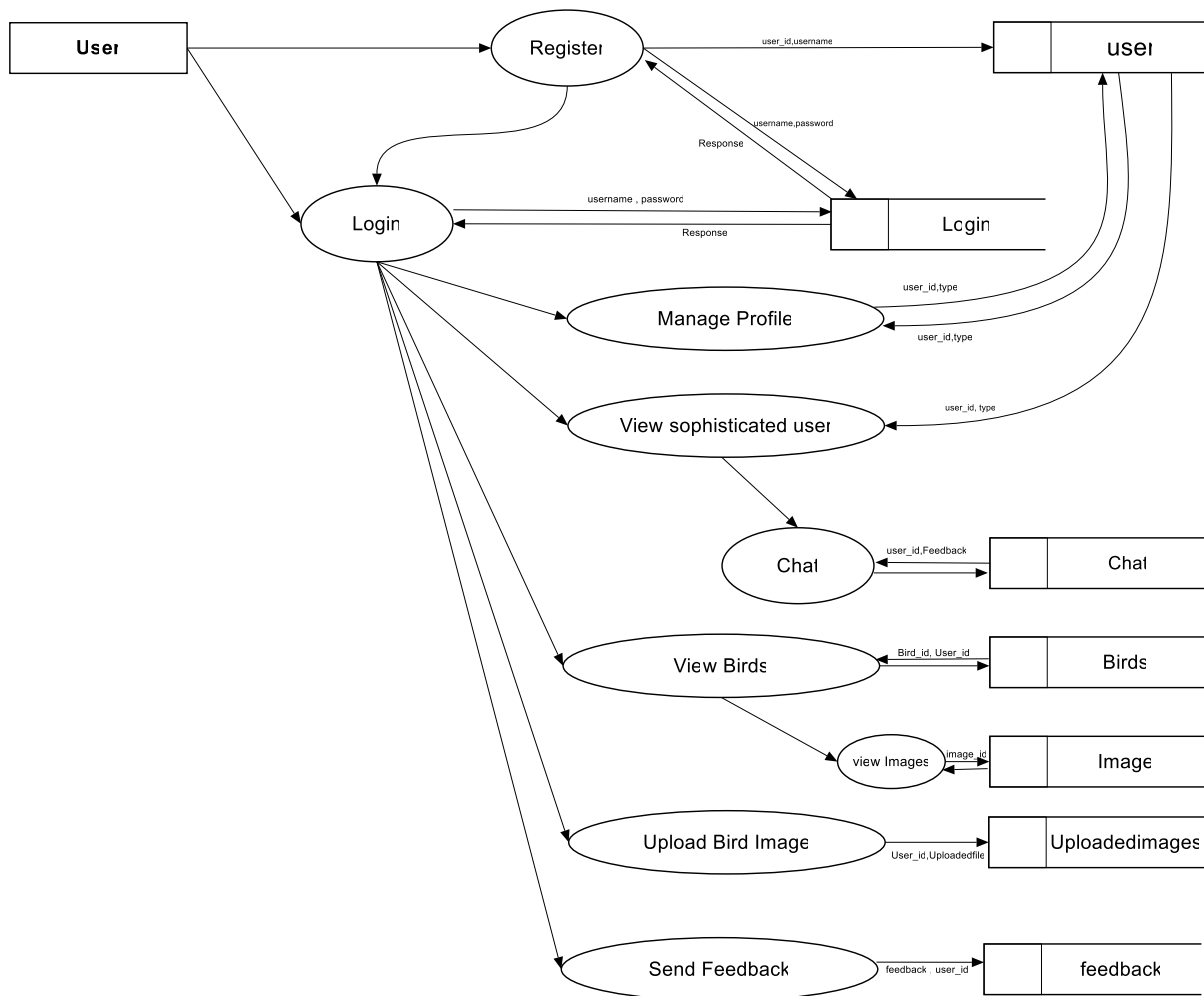
LEVEL 1 DFD FOR ADMIN



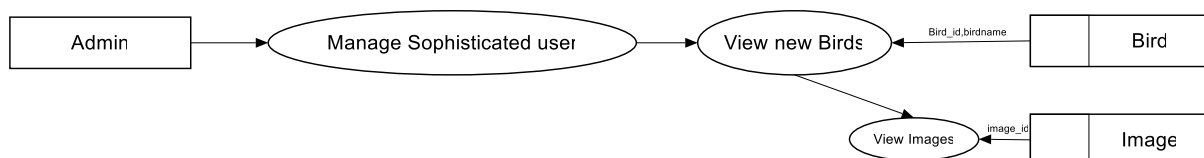
LEVEL 1 DFD FOR SOFESTICATED USERS



LEVEL 1 DFD FOR USERS



LEVEL 2 DFD FOR ADMIN



ENTITY-RELATIONSHIP DIAGRAM

The ER diagram is a graphical representation of entities and their relationships to each other, typically used in computing regarding the organization of data within database or information systems. An entity is a piece of data an object or concept about which data is stored. A relationship is how the data is shared between entities.

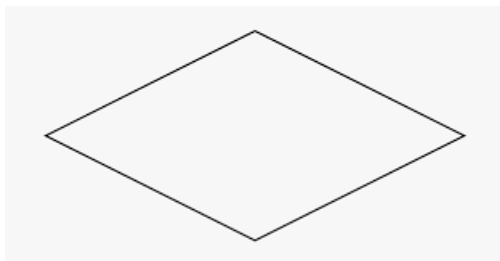
ER-SYMBOLS

➤ ENTITY



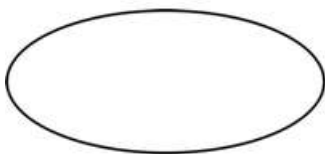
An entity is defined as a thing that is recognized as being capable of independent existence and which can be uniquely identified. An entity is abstract from the complexities of a domain. When we speak of an entity, we normally speak of some aspect of the real world that can be distinguished from other aspects of the world.

➤ RELATIONSHIP



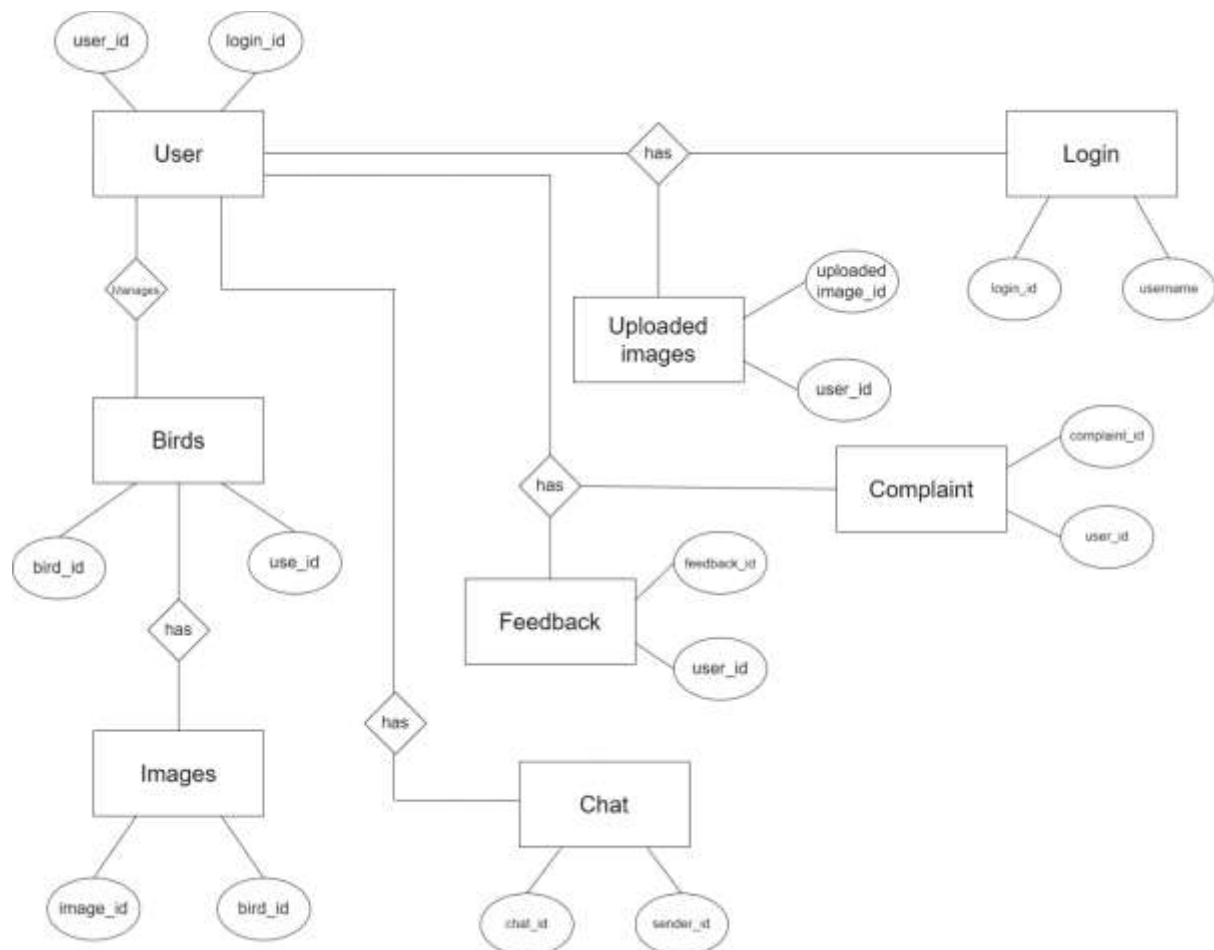
The relationship is a meaningful association between or among entities. A relationship provides useful information that could not be discerned with just the entity types.

➤ ATTRIBUTES



Attributes are characteristics of an entity, a many-to-one relationship, or a one-to-one relationship.

ER DIAGRAM



USE CASE DIAGRAM

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses. While a use case itself might drill into a lot of detail about every possibility, a use-case diagram can help provide a higher-level view of the system. It has been said before that "Use case diagrams are the blueprints for your system". They provide a simplified and graphical representation of what the system must do.

GUIDELINES FOR DRAWING USE CASE DIAGRAM

A use case describes how a user uses a system to accomplish a particular goal. A use case diagram consists of the system, the related use cases, and actors and relates these to each other to visualize: what is being described? (system), who is using the system? (actors) and what do the actors want to achieve? (use cases), thus, use cases help ensure that the correct system is developed by capturing the requirements from the user's point of view.

- Give meaningful business-relevant names for actors – For example, if your use case interacts with an outside organization it's much better to name it with the function rather than the organization name. (Eg: Airline Company is better than PanAir)
- Primary actors should be to the left side of the diagram – This enables you to quickly highlight the important roles in the system.
- Actors model roles (not positions) – In a hotel both the front office executive and shift manager can make reservations. So, something like "Reservation Agent" should be used for the actor's name to highlight the role.
- External systems are actors – If your use case is send-email and if interacts with the email management software then the software is an actor to that particular use case.
- Actors don't interact with other actors – In case actors interact within a system you need to create a new use case diagram with the system in the previous diagram represented as an actor.
- Place inheriting actors below the parent actor – This is to make it more readable and to quickly highlight the use cases specific for that actor.

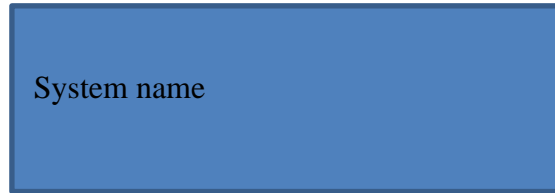
USE CASES

- Names begin with a verb – A use case models an action so the name should begin with a verb.
- Make the name descriptive – This is to give more information for others who are looking at the diagram. For example, "Print Invoice" is better than "Print".
- Highlight the logical order – For example, if you're analyzing a bank customer typical use cases include open account, deposit, and withdrawal. Showing them in the logical order makes more sense.
- Place included use cases to the right of the invoking use case – This is done to improve readability and add clarity.
- Place inheriting use case below parent use case – Again this is done to improve the readability of the diagram.

BASIC USE CASE DIAGRAM SYMBOLS

The primitive symbols used in the USE CASE are:

➤ SYSTEM



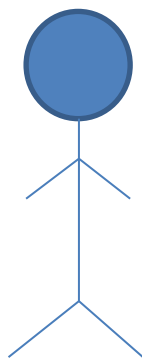
Draw your system's boundaries using a rectangle that contains use cases. Place actors outside the system's boundaries.

➤ USE CASE



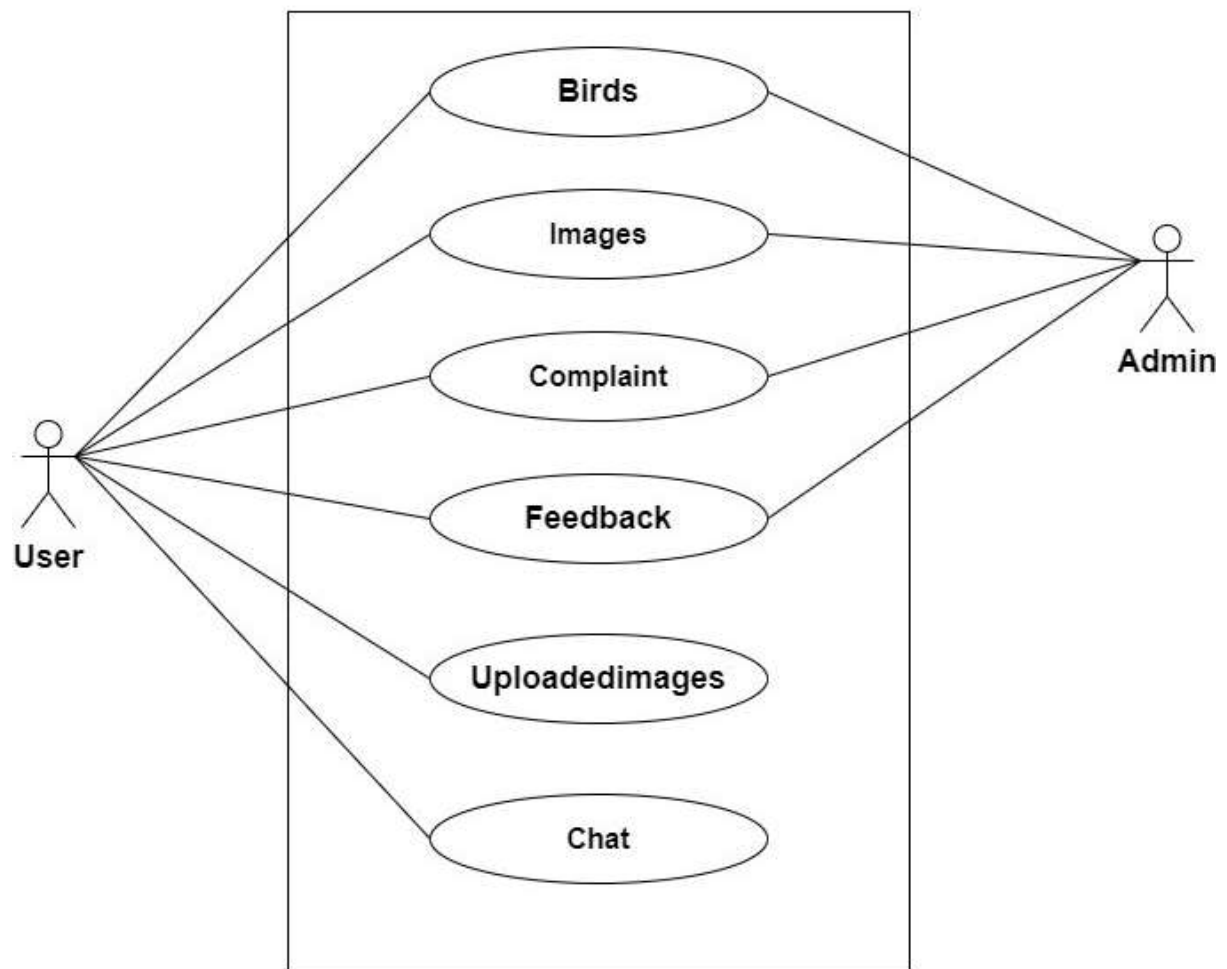
Draw use cases using ovals. Label the ovals with verbs that represent the system's functions.

➤ ACTOR



Actors are the users of a system. When one system is the actor of another system, label the actor system with the actor stereotype. An actor is a person, organization, or external system that plays a role in one or more interactions with your system (actors are typically drawn as stick figures on UML Use Case diagrams).

USECASE DIAGRAM



4. SYSTEM DEVELOPMENT

4.1 MODULE DESCRIPTION

There are basically 3 modules in this proposed system

1. Admin

- Login
- Manage Sophisticated Users
 - View New birds
 - View Images
- View Users
- View Birds
 - View Images
- view Complaint
 - sent Reply
- view Feedback

2. Sophisticated User

- Login
- Manage profile
- Add New Birds
 - Add Images
- Upload Birds Images
 - View Predicted Output
- View Users
- Chat with users
- Send Complaint
 - View Reply

3. User

- Register
- Login
- Edit Profile
- View Birds
 - View Images

- Upload Bird Images
 - View Predicted Output
- View Sophisticated Users
 - Chat
- Send Feedback

➤ **ADMIN**

The admin module consists of activities that are managed by the administrator of the system. He is the user with the most privileges and responsibilities in the system. The activities of the admin are:

- Login
- Manage profile
- Add New Birds
 - Add Images
- Upload Birds Images
 - View Predicted Output
- View Users
- Chat with users
- Send Complaint
 - View Reply

The admin must login into the system for performing all of the above activities. So that he will be provided with a unique login ID and password for logging into the system. He also has a separate login page for enhancing security. Once the admin login process is completed he can view all the operations listed above. He can add other users, while adding new users, the admin will provide details like username, password, etc

➤ **SOPHISTICATED USER(ANDROID)**

The second important user in this system is sophisticated user. They are added by the admin. Sophisticated users are the users with high knowledge about the system. Following are the activities that are done by the health workers.

- Login
- Manage profile
- Add New Birds
 - Add Images
- Upload Birds Images
 - View Predicted Output
- View Users
- Chat with users
- Send Complaint
 - View Reply

sophisticated user login in to the application, can view their profile. sophisticated user add new bird images and bird details to the system.

➤ **USERS (ANDROID)**

The common people are the registered users. The activities of the user are :

- Register
- Login
- Edit Profile
- View Birds
 - View Images
- Upload Bird Images
 - View Predicted Output
- View Sophisticated Users
 - Chat
- Send Feedback

The users must login in to the system to perform all the above activities. These users can select their username and password. Users can view the bird, They can upload bird images and view

the predicted output. The users can chat with the sophisticated users for clearing their doubts and to gain more knowledge about birds.

5. SYSTEM IMPLEMENTATION

5.1 TESTING

System testing is a critical aspect of Software Quality Assurance and represents ultimate view of specification, design and coding. Testing is a process of executing a program with the intent of finding an error a good test is one that has a probability of finding an as yet undiscovered error. The process of testing is to identify and correct bugs in the developed system. Nothing is complete without testing. Testing is the vital to the success of the system.

In the code testing the logic of the developed system is tested. For every module of program is executed to find an error. To perform specification test, the examination of the specifications starting what the program should do and how it should perform under various conditions.

Unit testing focuses first on the modules in the proposed system to locate errors. This enables from the interaction between modules are initially avoided. In unit testing step each module has to be checked separately.

5.2 VALIDATION CHECK

Validation is the process, whether we are building the right product i.e., to validate the product which we have developed is right or not. Activities involved in this is Testing the software application in simple words, Validation is to validate the actual and expected output of the software The process of evaluating software during or at the end of the development process to determine whether it satisfies specified requirements It's a High-Level Activity. Validation is a dynamic process of testing the real product.

Validation is intended to ensure a product, service, or system (or portion thereof, or set thereof) results in a product, service, or system (or portion thereof, or set thereof) that meets the operational needs of the user. For a new development flow or verification flow, validation procedures may involve modeling either flow and using simulations to predict faults or gaps that might lead to invalid or incomplete verification or development of a product, service, or system (or portion thereof, or set thereof). A set of validation requirements (as defined by the user), specifications, and regulations may then be used as a basis for qualifying a development flow or verification flow for a product, service, or system (or portion thereof, or set thereof).

	Test case	Form	Test Result
1	Validate the fields of username and password, only for the ones that have been registered, else print error	Login Form	Username and Password text fields validated and print error of "invalid username" or "invalid password" if there is a wrong entry
2	Validating the fields in the form entering, Company Name or any other field if left blank will display a message beside the empty field to enter the values for the particular field	Registration Form	"Please fill out this field" label is displayed and the values are re-entered and proceed with further registration
3	For entering the company name input field is validated to enter only alphabets and not include any numbers or special characters	Registration Form	Error message displayed if any special characters or numbers found in the Company Name field
4	For entering the contact number the total number of characters (numbers) allowed is 10. If found any value less than 10 or any characters other than numbers validate	Registration Form	Invalid contact number is displayed on entry of less than 10 numbers or alphabets
5	For entering the email. it should be a valid e mail id	Registration Form	If an email is entered without "@" in the text it displays an error to enter a valid email id
6	For entering the place, input field is validated to enter only alphabets and not include any numbers or special characters	Registration Form	Error message displayed if any special characters or numbers found in the Place field
7	For uploading bird images for prediction	Bird detection	Displayed predicted output

5.3 SYSTEM IMPLEMENTATION

The implementation is the final state and it is an important phase. It involves the individual programming; system testing, user training and the operational running of developed proposed system that constitutes the application subsystems. A major task of preparing for implementation is education of users, which should really have been taken place much earlier in the project when they were being involved in the investigation and design work. During the implementation phase system actually takes physical shape. In order to develop a system implemented planning is very essential.

The implementation phase of the software development is concerned with translating design specification into source code. The user tests the developed system and changes are made according to their needs. Our system has been successfully implemented. Before implementation several tests have been conducted to ensure that no errors are encountered during the operation. The implementation phase ends with an evaluation of the system after placing into the operation for a period of time.

The process of putting the developed system in actual use is called system implementation. This includes all those activities that take place to convert from old system to new system. The system can be implemented only after testing is done and is found to be working to specifications. The implementation stage involves following tasks:

- Careful planning.
- Investigation of system and constraints.
- Design of method to achieve change over.
- Evaluation of the changeover method.

5.4 SECURITY

As technology advances, application environment become more complex and application development security becomes more challenging. Applications, systems, and networks are constantly under various security attacks such as malicious code or denial of service. Some of the challenges from the application development security point of view include Viruses, Trojan horses, Logic bombs, Worms and Agent.

6.SYSTEM MAINTANENCE AND FUTURE ENHANCEMENTS

6.1 SYSTEM MAINTENANCE

Software maintenance is the modification of a software product after delivery to correct faults, to improve performance or other attributes. Maintenance is the ease with which a program can be corrected if any error is encountered, adapted if its environment changes or enhanced if the customer desires a change in requirement.

Maintenance follows conversation to extend that changes are necessary to maintain satisfactory operations relative to changes in the user's environment. Maintenance often includes minor enhancements or corrections to problems that surface in the system's operation. Maintenance is also done based on fixing the problems reported, changing the interface with other software or hardware enhancing the software.

6.2 FUTURE ENHANCEMENTS

The Future Enhancement which shall include in this software are:

1. We look forward to working for implementing the recommendations and seeing an improvement in the effectiveness of the organization.
2. Create an iOS app for iOS user.
3. Improve the dataset with more number of birds and higher accuracy.

7. CONCLUSION

7.1 CONCLUSION

This system is very flexible and changes can be made without much difficulty. The further extension in the system can be made to submit more outputs to the management. The development of the system is based on the ideas that we got during the detailed study of the system. We have tried our best to incorporate as many helps features as possible in order to make it easy for the user, such that even a person with just basic computer knowledge can adjust and work with it to produce the result required. On a concluding note, we would like to thank all people who, with their suggestion and encouragement helped to see me through the development of this project.

8. APPENDIX

8.1 TABLE DESIGN

1.Bird

FIELD	DATA TYPE	SIZE	DESCRIPTION
bird_id	Int	30	Used for bird id(primary key)
user_id	Int	30	Used for user id(foreign key)
birdname	Varchar	30	Used for bird name
scientific_name	Varchar	30	Used for scientific name
description	Varchar	1000	Used for description
date	varchar	30	Used for date

2.Chat

FIELD	DATA TYPE	SIZE	DESCRIPTION
Chat_id	int	30	Used for chat id(primary key)
Sender_id	int	30	Used for sender id(foreign key)
Receiver_id	int	30	Used for receiver id
message	varchar	1000	Used for message
date	varchar	30	Used for date

3.Complaint

FIELD	DATA TYPE	SIZE	DESCRIPTION
complaint_id	Int	30	Used for complaint id(primary key)
user_id	Int	30	Used for user id (foreign key)
complaint	Varchar	150	Used for complaint
reply	Varchar	150	Used for reply
Date	varchar	30	

4.Feedback

FIELD	DATA TYPE	SIZE	DESCRIPTION
feedback_id	int	30	Used for feedback(primary key)
user_id	int	30	Used for user id(foreign key)
feedback	varchar	30	Used for feedback
date	varchar	30	Used for date

5.Image

FIELD	DATA TYPE	SIZE	DESCRIPTION
image_id	Int	30	Used for image id(primary key)
bird_id	Int	30	Used for bird id(foreign key)
imagepath	varchar	100	Used for image path

6.Login

FIELD	DATA TYPE	SIZE	DESCRIPTION
Login_id	int	30	Used for login id(primary key)
Username	varchar	30	Used for username
Password	Varchar	30	Used for password
usertype	varchar	30	Used for user type

7.Uploadimage

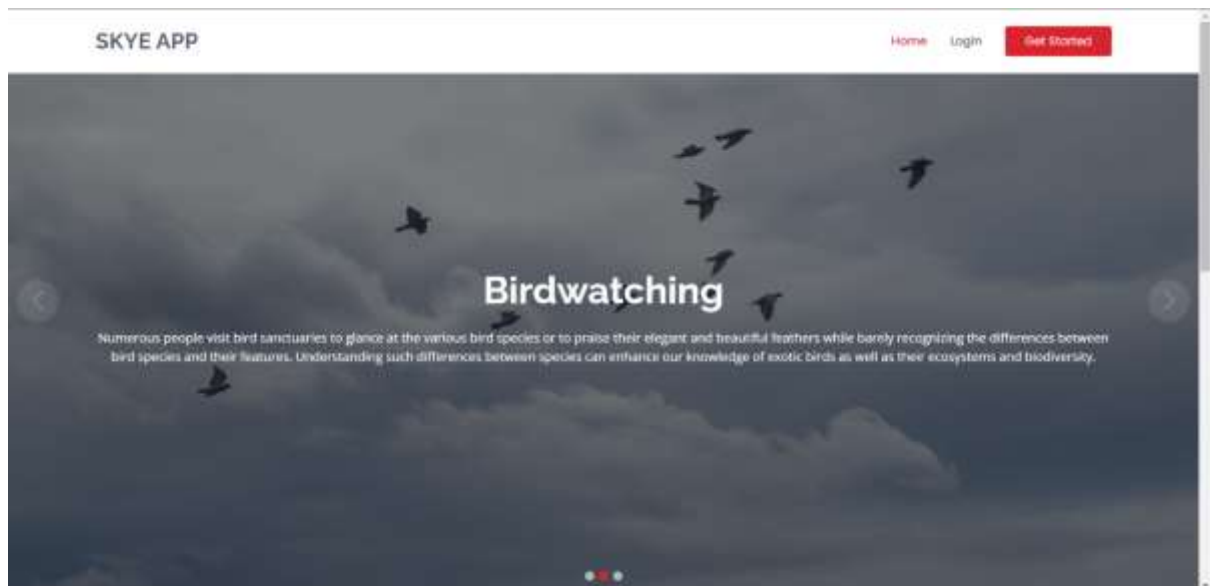
FIELD	DATA TYPE	SIZE	DESCRIPTION
Uploadimage_id	Int	30	Used for uploadimage id(primary key)
User_id	Int	30	Used for user id
Uploadedfile	Varchar	1000	Used for uploadfile
Output	Varchar	1000	Used for output
date	varchar	30	Used for date

8.Users

FIELD	DATA TYPE	SIZE	DESCRIPTION
User_id	Int	30	Used for user id(primary key)
Login_id	int	30	Used for login id(foreign key)
Fname	Varchar	30	Used for first name
Lname	Varchar	30	Used for last name
Place	Varchar	30	Used for place
Phone	Intvarchar	10	Used for phone
Email	Varchar	30	Used for email
Dob	Varchar	30	Used for date of birth
gender	Varchar	30	Used for gender
dp	Varchar	100	Used for dp
type	varchar	30	Used for type

8.2 SAMPLE INPUT DESIGN OUTPUT DESIGN

8.2.1 Home Page -Web



8.2.2 Login -Web

SKYE APP

Home Login Get Started

Login

Username

Password

Login

8.2.3 Admin Manage Sophisticated users

Home View Manage Complaints Feedback Logout

Sophisticated Users Registration

Firstname

Lastname

Place

Phone

Email

Add profile picture No file chosen

Date of Birth

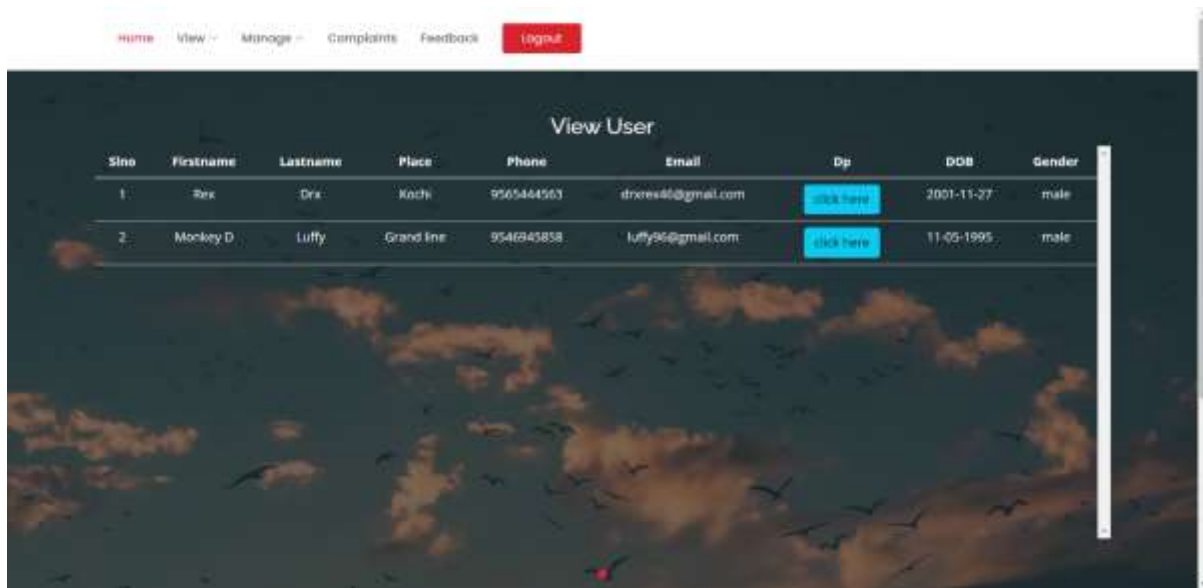
Gender ☒ Male ☐ Female ☐ Others

Username

Password

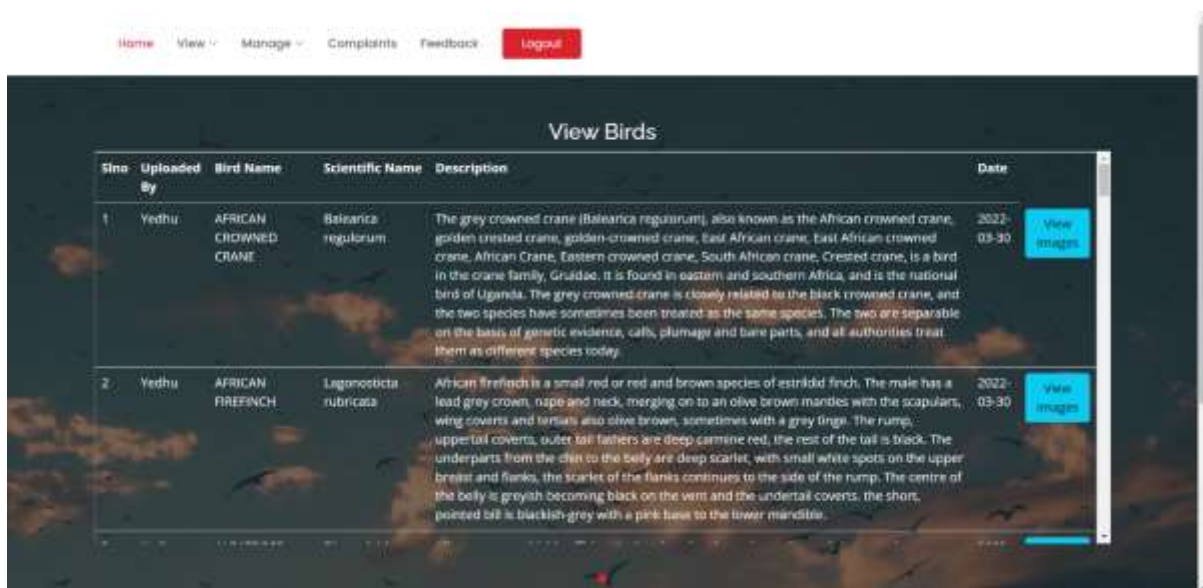
Register

8.2.4 Admin View Users



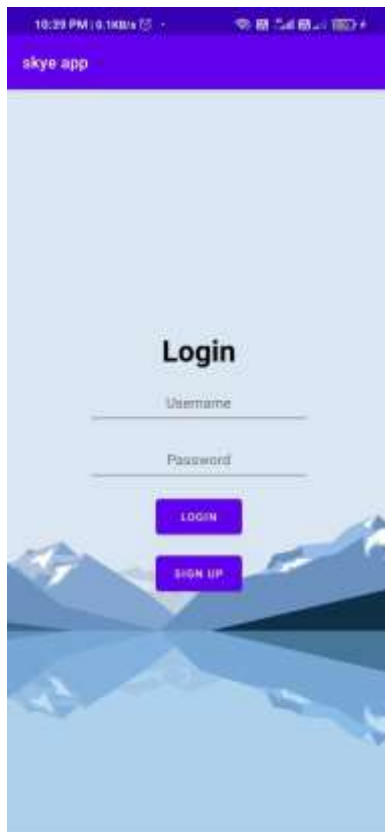
Sino	Firstname	Lastname	Place	Phone	Email	Dp	DOB	Gender
1	Rex	Dix	Kochi	9565444563	dixres4@gmail.com	click here	2001-11-27	male
2	Monkey D	Luffy	Grand line	9546943858	luffy96@gmail.com	click here	11-05-1995	male

8.2.5 Admin View Birds

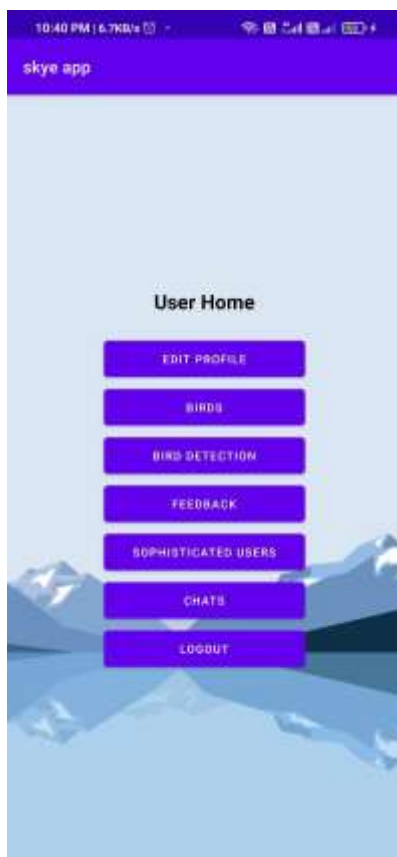


Sino	Uploaded By	Bird Name	Scientific Name	Description	Date
1	Yedhu	AFRICAN CROWNED CRANE	Balearcica regulorum	The grey crowned crane (Balearcica regulorum), also known as the African crowned crane, golden crested crane, golden-crowned crane, East African crane, East African crowned crane, African Crane, Eastern crowned crane, South African crane, Crested crane, is a bird in the crane family, Gruidae. It is found in eastern and southern Africa, and is the national bird of Uganda. The grey crowned crane is closely related to the black crowned crane, and the two species have sometimes been treated as the same species. The two are separable on the basis of genetic evidence, calls, plumage and bare parts, and all authorities treat them as different species today.	2022-03-30 View images
2	Yedhu	AFRICAN FIREFINCH	Lagonoticta rubricata	African firefinch is a small red or red and brown species of estrildid finch. The male has a lead grey crown, nape and neck, merging on to an olive brown mantle with the scapulars, wing coverts and tertials also olive brown, sometimes with a grey tinge. The rump, uppertail coverts, outer tail feathers are deep carmine red, the rest of the tail is black. The underparts from the chin to the belly are deep scarlet, with small white spots on the upper breast and flanks, the scarlet of the flanks continues to the side of the rump. The centre of the belly is greyish becoming black on the vent and the undertail coverts. The short, pointed bill is blackish-grey with a pink base to the lower mandible.	2022-03-30 View images

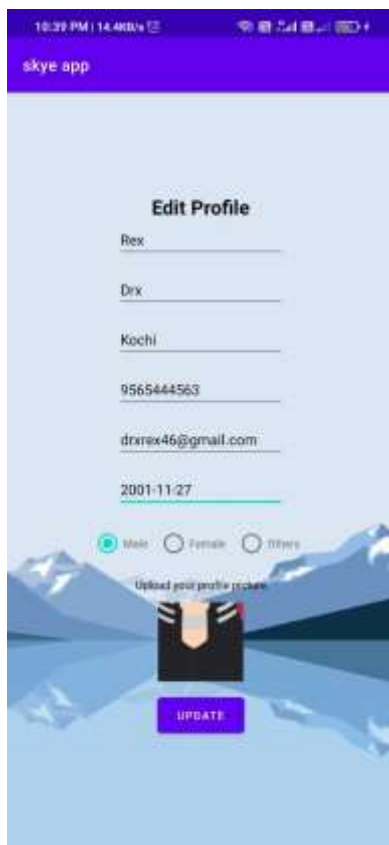
8.2.6 Login - Android



8.2.7 User Home



8.2.8 User Edit Profile



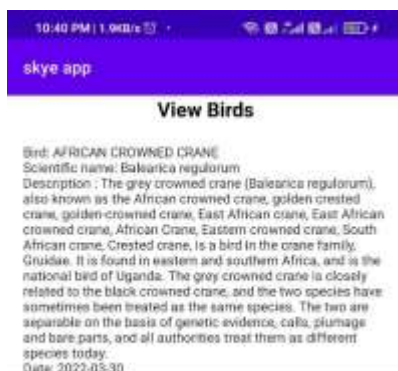
8.2.9 User View Birds Images



8.2.10 Bird Detection

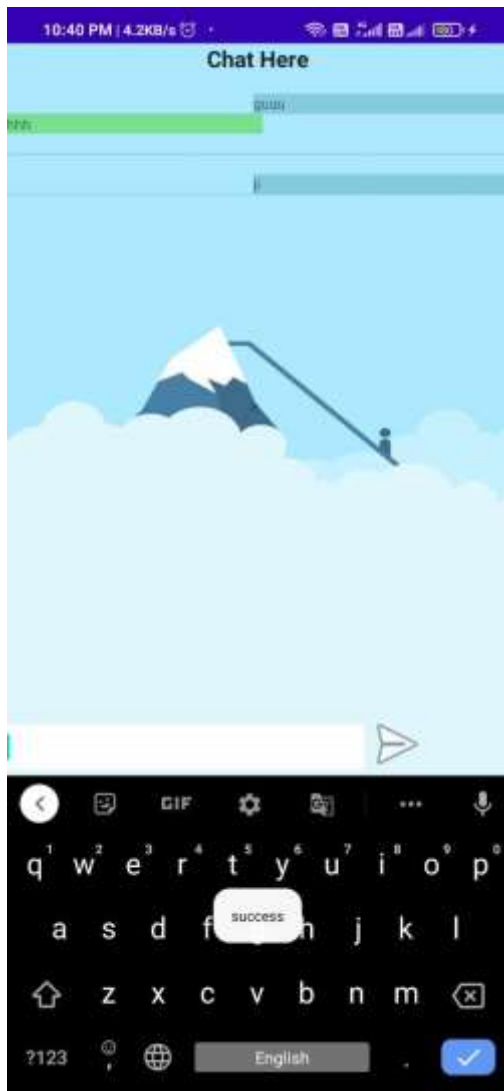


8.2.11 Bird Details



Bird: AFRICAN CROWNED CRANE
Scientific name: Balearica regulorum
Description : The grey crowned crane (Balearica regulorum), also known as the African crowned crane, golden crested crane, golden-crowned crane, East African crane, East African crowned crane, African Crane, Eastern crowned crane, South African crane, Crested crane, is a bird in the crane family, Gruidae. It is found in eastern and southern Africa, and is the national bird of Uganda. The grey crowned crane is closely related to the black crowned crane, and the two species have sometimes been treated as the same species. The two are separable on the basis of genetic evidence, calls, plumage and bare parts, and all authorities treat them as different species today.
Date: 2023-03-30

8.2.12 Chat



8.3 SOURCE CODE

8.3.1 main.py

```
from flask import Flask
from public import public
from admin import admin
from api import api
app=Flask(__name__)
app.secret_key="asdfg"
app.register_blueprint(public)
app.register_blueprint(admin,url_prefix='/admin')
app.register_blueprint(api,url_prefix='/api')
app.run(debug=True,port=5007,host="192.168.43.152")
```

8.3.2 api.py

```
from flask import *
from database import *
import demjson
import uuid
from test import *
api=Blueprint('api',__name__)
@api.route('/login')
def login():
    data={ }
    uname=request.args['username']
    password=request.args['password']
    q="select * from login where username='%s' and password='%s'"%(uname,password)
    res=select(q)
    if res:
        data['status']="success"
        data['data']=res
    else:
        data['status']="failed"
    return demjson.encode(data)
@api.route('/user_register',methods=['get','post'])
def user_register():
```

```
data={ }
fname=request.form['fname']
lname=request.form['lname']
place=request.form['place']
phone=request.form['phone']
email=request.form['email']
img=request.files['image']
dob=request.form['dob']
gender=request.form['gender']
uname=request.form['uname']
password=request.form['password']
path='static/'+str(uuid.uuid4())+img.filename
img.save(path)
q="select * from login where username='%s'"%(uname)
res=select(q)
if res:
    data['status']="duplicate"
else:
    q="insert into login values(null,'%s','%s','user')"%(uname,password)
    ids=insert(q)
    q2="insert into users
values(null,'%s','%s','%s','%s','%s','%s','%s','%s','user')"%(ids,fname,lname,place,phone,email,dob,gender,p
ath)
    insert(q2)
    data['status']="success"
return demjson.encode(data)
@api.route('/addbirds',methods=['get','post'])
def addbirds():
    data={ }
    logid=request.args['lid']
    birdname=request.args['bird']
    scientific_name=request.args['sname']
    description=request.args['desc']
    # img=request.files["image"]
    # path='static/'+str(uuid.uuid4())+img.filename
    # img.save(path)
    q="select * from bird where birdname='%s' and scientific_name='%s'"%(birdname,scientific_name)
```

```
res=select(q)
if res:
    data['status']="duplicate"
else:
    q2="insert into bird values(null,(select user_id from users where
login_id='%s'),'%s','%s','%s',curdate()))"%(logid,birdname,scientific_name,description)
    ids=insert(q2)
    # q3="insert into image values(null,'%s','%s')"%(ids,path)
    # insert(q3)
    data['status']="success"
    data['method']="addbirds"
    return demjson.encode(data)

@api.route('/viewbirds')
def viewbirds():
    data={ }
    logid=request.args['lid']
    q="select * from bird where user_id=(select user_id from users where login_id='%s')"%(logid)
    res=select(q)
    if res:
        data['status']="success"
        data['data']=res
    else:
        data['status']="failed"
    data['method']="viewbirds"
    return demjson.encode(data)

@api.route('/complaints', methods=['get','post'])
def complaints():
    data={ }
    logid=request.args['lid']
    complaint=request.args['complaint']
    q="insert into complaint values(null,(select user_id from users where
login_id='%s'),'%s','pending',curdate()))"%(logid,complaint)
    insert(q)
    data['status']="success"
    data['method']="complaints"
    return demjson.encode(data)
```



```
@api.route('/viewcomplaints', methods=['get','post'])
def viewcomplaints():
    data={ }
    logid=request.args['lid']
    q="select * from complaint where user_id=(select user_id from users where login_id='%s')"% (logid)
    res=select(q)
    if res:
        data['status']="success"
        data['data']=res
    else:
        data['status']="failed"
    data['method']="viewcomplaints"
    return demjson.encode(data)

@api.route('/feedbacks')
def feedbacks():
    data={ }
    logid=request.args['lid']
    feedback=request.args['feedback']
    q="insert into feedback values(null,(select user_id from users where login_id='%s'), '%s',curdate())"% (logid,feedback)
    insert(q)
    data['status']="success"
    data['methods']="feedbacks"
    return demjson.encode(data)

@api.route('/UserView')
def UserView():
    data={ }
    q="select * from users where type='user'"
    res=select(q)
    if res:
        data['status']="success"
        data['data']=res
    else:
        data['status']="failed"
    return demjson.encode(data)

@api.route('/SuView')
```

```
def SuView():
    data={ }
    q="select * from users where type='suser'"
    res=select(q)
    if res:
        data['status']="success"
        data['data']=res
    else:
        data['status']="failed"
    return demjson.encode(data)

@api.route('/userviewprofile')
def userviewprofile():
    data={ }
    lid=request.args['lid']
    q="select * from users where login_id='%s'" %(lid)
    res=select(q)
    if res:
        data['status']="success"
        data['data']=res
    else:
        data['status']="failed"
    data['methods']="userviewprofile"
    return demjson.encode(data)

@api.route('/usereditprofile',methods=['get','post'])
def usereditprofile():
    data={ }
    lid=request.form['lid']
    fname=request.form['fname']
    lname=request.form['lname']
    place=request.form['place']
    phone=request.form['phone']
    email=request.form['email']
    dob=request.form['dob']
    gender=request.form['gender']

    if "image" in request.files:
```

```
        print("Haii")

        img=request.files['image']

        path='static/uploads/'+str(uuid.uuid4())+img.filename

        img.save(path)

        q2="update users set
fname='%s',lname='%s',place='%s',phone='%s',email='%s',dob='%s',gender='%s',dp='%s' where
login_id='%s'"%(fname,lname,place,phone,email,dob,gender,path,lid)

        update(q2)

    else:

        q2="update users set
fname='%s',lname='%s',place='%s',phone='%s',email='%s',dob='%s',gender='%s' where
login_id='%s'"%(fname,lname,place,phone,email,dob,gender,lid)

        update(q2)

        data['status']="success"

        data['methods']="usereditprofile"

        return demjson.encode(data)

@api.route('/Userviewbirds')
def Userviewbirds():

    data={ }

    bids=request.args['bids']

    q="select * from bird inner join users using(user_id) where bird_id='%s'"%(bids)

    res=select(q)

    if res:

        data['status']="success"

        data['data']=res

    else:

        data['status']="failed"

        data['methods']="Userviewbirds"

        return demjson.encode(data)

@api.route('/Checkbird', methods=['get','post'])
def Checkbird():

    data={ }

    logid=request.form['logid']

    image=request.files['image']

    path="static/uploads/"+str(uuid.uuid4())+image.filename

    image.save(path)
```

```
res=valspredict(path)

q="insert into uploadedimage values(null,(select user_id from users where
login_id='%s'), '%s', '%s', curdate())" %(logid, path, res)

insert(q)

data['status']="success"

data['method']="Checkbird"

return demjson.encode(data)

@api.route('/viewpredictedoutput')
def viewpredictedoutput():
    data={ }
    logid=request.args['logid']
    q="select * from uploadedimage where user_id=(select user_id from users where login_id='%s')"
    %(logid)
    res=select(q)
    if res:
        data['status']="success"
        data['data']=res
    else:
        data['status']="failed"
        data['method']="viewpredictedoutput"
    return demjson.encode(data)

@api.route('/chatdetail')
def chatdetail():
    data={ }
    sender_id=request.args['sender_id']
    receiver_id=request.args['receiver_id']
    q="select * from chat where (sender_id='%s' and receiver_id='%s') or (sender_id='%s' and
receiver_id='%s') " %(sender_id, receiver_id, receiver_id, sender_id)
    res=select(q)
    if res:
        data['status']="success"
        data['data']=res
    else:
        data['status']="failed"
        data['method']="chatdetail"
    return demjson.encode(data)

@api.route('/chat')
```

```
def chat():
    data={ }
    sender_id=request.args['sender_id']
    receiver_id=request.args['receiver_id']
    details=request.args['details']
    q="insert into chat values(null,'%s','%s','%s',curdate())"%(sender_id,receiver_id,details)
    r=insert(q)
    data['status']="success"
    data['method']="chat"
    return demjson.encode(data)
@api.route('/User_view_bird_images')
def User_view_bird_images():
    data={ }
    q="SELECT * FROM `image` "
    r=select(q)
    if r:
        data['status']="success"
        data['data']=r
        data['method']="User_view_bird_images"
        return demjson.encode(data)
@api.route('/su_view_bird_images')
def su_view_bird_images():
    data={ }
    q="SELECT * FROM `image` "
    r=select(q)
    if r:
        data['status']="success"
        data['data']=r
        data['method']="su_view_bird_images"
        return demjson.encode(data)
@api.route('/user_view_company_chats', methods=['get', 'post'])
def user_view_company_chats():
    data = { }
    login_id = request.args['login_id']
    q = "SELECT *,concat(`fname`,`lname`) as username FROM `users` WHERE `login_id` IN(SELECT IF(sender_id='%s',receiver_id,sender_id) FROM `chat` WHERE `sender_id`='%s' OR `receiver_id`='%s' )"%(login_id,login_id,login_id)
```

```
res = select(q)
print(q)
if res:
    data['status'] = 'success'
    data['data'] = res
else:
    data['status'] = 'failed'
data['method'] = 'user_view_company_chats'
return demjson.encode(data)
@api.route('/Addimages', methods=['get', 'post'])
def Addimages():
    data = { }
    image = request.files['image']
    path='static/'+str(uuid.uuid4())+image.filename
    image.save(path)
    bid=request.form['bid']
    q = "INSERT INTO `image` VALUES(NULL,'%s','%s')"%(bid,path)
    insert(q)
    if id:
        data['status'] = 'success'
    else:
        data['status'] = 'failed'
    data['method'] = 'Addimages'
    return demjson.encode(data)
@api.route('/Userviewbirdsss')
def Userviewbirdsss():
    data={ }
    bname=request.args["bname"]+"%"
    q="select * from bird  where birdname like '%s'"%(bname)
    res=select(q)
    print(res)
    if res:
        data['status']="success"
        data['data']=res
    else:
        data['status']="failed"
```

```
data['methods']="Userviewbirdsss"  
  
return demjson.encode(data)
```

8.3.3 test.py

```
from keras.models import load_model  
  
from keras.preprocessing.image import load_img, img_to_array  
  
from keras.preprocessing.image import ImageDataGenerator  
  
import matplotlib.pyplot as plt  
  
import matplotlib.image as mpimg  
  
import numpy as np  
  
model1 = load_model('model/BC.h5', compile=False)  
  
lab = {0: 'AFRICAN CROWNED CRANE', 1: 'AFRICAN FIREFINCH', 2: 'ALBATROSS', 3:  
'ALEXANDRINE PARAKEET', 4: 'AMERICAN AVOCET', 5: 'AMERICAN BITTERN', 6: 'AMERICAN  
COOT', 7: 'AMERICAN GOLDFINCH', 8: 'AMERICAN KESTREL', 9: 'AMERICAN PIPIT', 10:  
'AMERICAN REDSTART', 11: 'ANHINGA', 12: 'ANNAS HUMMINGBIRD', 13: 'ANTBIRD', 14: 'ARARIPE  
MANAKIN', 15: 'ASIAN CRESTED IBIS', 16: 'BALD EAGLE', 17: 'BALI STARLING', 18: 'BALTIMORE  
ORIOLE', 19: 'BANANAQUIT', 20: 'BANDED BROADBILL', 21: 'BAR-TAILED GODWIT', 22: 'BARN  
OWL', 23: 'BARN SWALLOW', 24: 'BARRLED PUFFBIRD', 25: 'BAY-BREASTED WARBLER', 26:  
'BEARDED BARBET', 27: 'BEARDED REEDLING', 28: 'BELTED KINGFISHER', 29: 'BIRD OF  
PARADISE', 30: 'BLACK & YELLOW bROADBILL', 31: 'BLACK FRANCOLIN', 32: 'BLACK SKIMMER',  
33: 'BLACK SWAN', 34: 'BLACK TAIL CRAKE', 35: 'BLACK THROATED BUSHTIT', 36: 'BLACK  
THROATED WARBLER', 37: 'BLACK VULTURE', 38: 'BLACK-CAPPED CHICKADEE', 39: 'BLACK-  
NECKED GREBE', 40: 'BLACK-THROATED SPARROW', 41: 'BLACKBURNIAM WARBLER', 42: 'BLUE  
GROUSE', 43: 'BLUE HERON', 44: 'BOBOLINK', 45: 'BORNEAN BRISTLEHEAD', 46: 'BORNEAN  
LEAFBIRD', 47: 'BROWN NOODY', 48: 'BROWN THRASHER', 49: 'BULWERS PHEASANT', 50:  
'CACTUS WREN', 51: 'CALIFORNIA CONDOR', 52: 'CALIFORNIA GULL', 53: 'CALIFORNIA QUAIL',  
54: 'CANARY', 55: 'CAPE MAY WARBLER', 56: 'CAPUCHINBIRD', 57: 'CARMINE BEE-EATER', 58:  
'CASPIAN TERN', 59: 'CASSOWARY', 60: 'CEDAR WAXWING', 61: 'CHARA DE COLLAR', 62:  
'CHIPPING SPARROW', 63: 'CHUKAR PARTRIDGE', 64: 'CINNAMON TEAL', 65: 'COCK OF THE  
ROCK', 66: 'COCKATOO', 67: 'COMMON FIRECREST', 68: 'COMMON GRACKLE', 69: 'COMMON  
HOUSE MARTIN', 70: 'COMMON LOON', 71: 'COMMON POORWILL', 72: 'COMMON STARLING', 73:  
'COUCHS KINGBIRD', 74: 'CRESTED AUKLET', 75: 'CRESTED CARACARA', 76: 'CRESTED  
NUTHATCH', 77: 'CROW', 78: 'CROWNED PIGEON', 79: 'CUBAN TODY', 80: 'CURL CRESTED  
ARACURI', 81: 'D-ARNAUDS BARBET', 82: 'DARK EYED JUNCO', 83: 'DOUBLE BARRED FINCH', 84:  
'DOWNY WOODPECKER', 85: 'EASTERN BLUEBIRD', 86: 'EASTERN MEADOWLARK', 87: 'EASTERN  
ROSELLA', 88: 'EASTERN TOWEE', 89: 'ELEGANT TROGON', 90: 'ELLIOTS PHEASANT', 91:  
'EMPEROR PENGUIN', 92: 'EMU', 93: 'ENGGANO MYNA', 94: 'EURASIAN GOLDEN ORIOLE', 95:  
'EURASIAN MAGPIE', 96: 'EVENING GROSBEAK', 97: 'FIRE TAILED MYZORNIS', 98: 'FLAME  
TANAGER', 99: 'FLAMINGO', 100: 'FRIGATE', 101: 'GAMBELS QUAIL', 102: 'GANG GANG  
COCKATOO', 103: 'GILA WOODPECKER', 104: 'GILDED FLICKER', 105: 'GLOSSY IBIS', 106: 'GO  
AWAY BIRD', 107: 'GOLD WING WARBLER', 108: 'GOLDEN CHEEKED WARBLER', 109: 'GOLDEN  
CHLOROPHONIA', 110: 'GOLDEN EAGLE', 111: 'GOLDEN PHEASANT', 112: 'GOLDEN PIPIT', 113:  
'GOULDIAN FINCH', 114: 'GRAY CATBIRD', 115: 'GRAY PARTRIDGE', 116: 'GREAT POTOO', 117:  
'GREATOR SAGE GROUSE', 118: 'GREEN JAY', 119: 'GREEN MAGPIE', 120: 'GREY PLOVER', 121:  
'GUINEA TURACO', 122: 'GUINEAFOWL', 123: 'GYRFALCON', 124: 'HARPY EAGLE', 125: 'HAWAIIAN  
GOOSE', 126: 'HELMET VANGA', 127: 'HIMALAYAN MONAL', 128: 'HOATZIN', 129: 'HOODED  
MERGANSER', 130: 'HOOPOES', 131: 'HORNBILL', 132: 'HORNED GUAN', 133: 'HORNED SUNGEM',  
134: 'HOUSE FINCH', 135: 'HOUSE SPARROW', 136: 'IMPERIAL SHAQ', 137: 'INCA TERN', 138:  
'INDIAN BUSTARD', 139: 'INDIAN PITTA', 140: 'INDIGO BUNTING', 141: 'JABIRU', 142: 'JAVA  
SPARROW', 143: 'KAKAPO', 144: 'KILLDEAR', 145: 'KING VULTURE', 146: 'KIWI', 147:  
'KOOKABURRA', 148: 'LARK BUNTING', 149: 'LEARS MACAW', 150: 'LILAC ROLLER', 151: 'LONG-  
EARED OWL', 152: 'MAGPIE GOOSE', 153: 'MALABAR HORNBILL', 154: 'MALACHITE KINGFISHER',  
155: 'MALEO', 156: 'MALLARD DUCK', 157: 'MANDRIN DUCK', 158: 'MARABOU STORK', 159:  
'MASKED BOOBY', 160: 'MASKED LAPWING', 161: 'MIKADO PHEASANT', 162: 'MOURNING DOVE',
```

163: 'MYNA', 164: 'NICOBAR PIGEON', 165: 'NOISY FRIARBIRD', 166: 'NORTHERN BALD IBIS', 167: 'NORTHERN CARDINAL', 168: 'NORTHERN FLICKER', 169: 'NORTHERN GANNET', 170: 'NORTHERN GOSHAWK', 171: 'NORTHERN JACANA', 172: 'NORTHERN MOCKINGBIRD', 173: 'NORTHERN PARULA', 174: 'NORTHERN RED BISHOP', 175: 'NORTHERN SHOVELER', 176: 'OCELLATED TURKEY', 177: 'OKINAWA RAIL', 178: 'OSPREY', 179: 'OSTRICH', 180: 'OYSTER CATCHER', 181: 'PAINTED BUNTIG', 182: 'PALILA', 183: 'PARADISE TANAGER', 184: 'PARUS MAJOR', 185: 'PEACOCK', 186: 'PELICAN', 187: 'PEREGRINE FALCON', 188: 'PHILIPPINE EAGLE', 189: 'PINK ROBIN', 190: 'PUFFIN', 191: 'PURPLE FINCH', 192: 'PURPLE GALLINULE', 193: 'PURPLE MARTIN', 194: 'PURPLE SWAMPHEN', 195: 'QUETZAL', 196: 'RAINBOW LORIKEET', 197: 'RAZORBILL', 198: 'RED BEARDED BEE EATER', 199: 'RED BELLIED PITTA', 200: 'RED BROWED FINCH', 201: 'RED FACED CORMORANT', 202: 'RED FACED WARBLER', 203: 'RED HEADED DUCK', 204: 'RED HEADED WOODPECKER', 205: 'RED HONEY CREEPER', 206: 'RED TAILED THRUSH', 207: 'RED WINGED BLACKBIRD', 208: 'RED WISKERED BULBUL', 209: 'REGENT BOWERBIRD', 210: 'RING-NECKED PHEASANT', 211: 'ROADRUNNER', 212: 'ROBIN', 213: 'ROCK DOVE', 214: 'ROSY FACED LOVEBIRD', 215: 'ROUGH LEG BUZZARD', 216: 'ROYAL FLYCATCHER', 217: 'RUBY THROATED HUMMINGBIRD', 218: 'RUFIOUS KINGFISHER', 219: 'RUFUOS MOTMOT', 220: 'SAMATRAN THRUSH', 221: 'SAND MARTIN', 222: 'SCARLET IBIS', 223: 'SCARLET MACAW', 224: 'SHOEBILL', 225: 'SHORT BILLED DOWITCHER', 226: 'SMITHS LONGSPUR', 227: 'SNOWY EGRET', 228: 'SNOWY OWL', 229: 'SORA', 230: 'SPANGLED COTINGA', 231: 'SPLENDID WREN', 232: 'SPOON BILED SANDPIPER', 233: 'SPOONBILL', 234: 'SRI LANKA BLUE MAGPIE', 235: 'STEAMER DUCK', 236: 'STORK BILLED KINGFISHER', 237: 'STRAWBERRY FINCH', 238: 'STRIPPED SWALLOW', 239: 'SUPERB STARLING', 240: 'SWINHOES PHEASANT', 241: 'TAIWAN MAGPIE', 242: 'TAKAHE', 243: 'TASMANIAN HEN', 244: 'TEAL DUCK', 245: 'TIT MOUSE', 246: 'TOUCHAN', 247: 'TOWNSENDS WARBLER', 248: 'TREE SWALLOW', 249: 'TRUMPTER SWAN', 250: 'TURKEY VULTURE', 251: 'TURQUOISE MOTMOT', 252: 'UMBRELLA BIRD', 253: 'VARIED THRUSH', 254: 'VENEZUELIAN TROUPIAL', 255: 'VERMILION FLYCATHER', 256: 'VICTORIA CROWNED PIGEON', 257: 'VIOLET GREEN SWALLOW', 258: 'VULTURINE GUINEAFOWL', 259: 'WATTLED CURASSOW', 260: 'WHIMBREL', 261: 'WHITE CHEEKED TURACO', 262: 'WHITE NECKED RAVEN', 263: 'WHITE TAILED TROPIC', 264: 'WILD TURKEY', 265: 'WILSONS BIRD OF PARADISE', 266: 'WOOD DUCK', 267: 'YELLOW BELLIED FLOWERPECKER', 268: 'YELLOW CACIQUE', 269: 'YELLOW HEADED BLACKBIRD'}

```
def output(location):
```

```
    img=load_img(location,target_size=(224,224,3))
```

```
    img=img_to_array(img)
```

```
    img=img/255
```

```
    img=np.expand_dims(img,[0])
```

```
    answer=model1.predict(img)
```

```
    y_class = answer.argmax(axis=-1)
```

```
    y = " ".join(str(x) for x in y_class)
```

```
    y = int(y)
```

```
    res = lab[y]
```

```
    return res
```

```
def valspredict(path):
```

```
    result = output(path)
```

```
    print(result)
```

```
    return result
```


8.3.4 Login -Web

```
{% include 'public_header.html' %}

<section id="hero">

  <div id="heroCarousel" data-bs-interval="5000" class="carousel slide carousel-fade" data-bs-
ride="carousel">

    <ol class="carousel-indicators" id="hero-carousel-indicators"></ol>

    <div class="carousel-inner" role="listbox">

      <div class="carousel-item active" style="background-image: url(/static/assets/img/slide/slide-1.jpg)">

        <div class="carousel-container">

          <div class="container">

            <h2 class="animate__animated animate__fadeInDown"><span>Login</span></h2>

            <center>

              <form method="POST">

                <table align="center" style="font-size:20px ;color: white" class="animate__animated
animate__fadeInDown ">

                  <tr>

                    <th>Username</th>

                    <td><input type="text" name="uname" class="form-control"
placeholder="Enter your username"></td>

                  </tr>

                  <tr>

                    <th style="margin-right: 10px" >Password</th>

                    <td><input type="password" name="password" class="form-control"
placeholder="Enter your password"></td>

                  </tr>

                  <tr>

                    <td align="center" colspan="2"><input type="submit" name="login"
value="Login" class="btn btn-primary "></td>

                  </tr>

                </table>

              </form>

            </center>

          </div>

        </div>

      </div>

    </div>

  </section>

{% include 'footer.html' %}
```

8.3.5 Login – Android

```
package com.example.skyeapp;

import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.preference.PreferenceManager;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import org.json.JSONArray;
import org.json.JSONObject;

public class login extends AppCompatActivity implements JsonResponse {

    EditText e1, e2;
    Button b1,b2;
    String uname,passw;
    SharedPreferences sh;
    public static String logid,usertype;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

        sh = PreferenceManager.getDefaultSharedPreferences(getApplicationContext());
        e1 = (EditText) findViewById(R.id.e1);
        e2 = (EditText) findViewById(R.id.e2);
        b1 = (Button) findViewById(R.id.b1);
        b2 = (Button) findViewById(R.id.b2);
        b2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                startActivity(new Intent(getApplicationContext(), user_registration.class));
            }
        });
    }
}
```

```
b1.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View v) {  
        uname = e1.getText().toString();  
        passw = e2.getText().toString();  
        if (uname.equalsIgnoreCase("")) {  
            e1.setError("Enter Your Username");  
            e1.setFocusable(true);  
        } else if (passw.equalsIgnoreCase("")) {  
            e2.setError("Enter Your Password");  
            e2.setFocusable(true);  
        } else {  
            JsonRequest JR = new JsonRequest();  
            JR.json_response = (JsonResponse) login.this;  
            String q = "/login?username=" + uname + "&password=" + passw;  
            q = q.replace(" ", "%20");  
            JR.execute(q);  
        }  
    }  
});  
}  
@Override  
public void response(JSONObject jo) {  
    try {  
        String status=jo.getString("status");  
        Log.d("pearl",status);  
        if(status.equalsIgnoreCase("success")){  
            JSONArray ja1=(JSONArray)jo.getJSONArray("data");  
            loginid=ja1.getJSONObject(0).getString("login_id");  
            usertype=ja1.getJSONObject(0).getString("usertype");  
            SharedPreferences.Editor e=sh.edit();  
            e.putString("log_id", loginid);  
            e.putString("type", usertype);  
            e.commit();  
            if(usertype.equals("user"))  
            {
```

```
        Toast.makeText(getApplicationContext()," You are Login  
Successfully!...",Toast.LENGTH_LONG).show();  
        startActivity(new Intent(getApplicationContext(),User_view_bird_images.class));  
    }  
    else if(usertype.equals("suser"))  
    {  
        Toast.makeText(getApplicationContext()," You are Login  
Successfully!...",Toast.LENGTH_LONG).show();  
        startActivity(new Intent(getApplicationContext(),sophisticated_users_home.class));  
    }  
    }  
    else {  
        Toast.makeText(getApplicationContext(),"Login failed..!Please enter correct username or password  
",Toast.LENGTH_LONG).show();  
        startActivity(new Intent(getApplicationContext(),login.class));  
    }  
    }catch (Exception e) {  
        Toast.makeText(getApplicationContext(),e.toString(), Toast.LENGTH_LONG).show();  
    }  
    }  
    public void onBackPressed()  
    {  
        super.onBackPressed();  
        Intent b=new Intent(getApplicationContext(),IpSettings.class);  
        startActivity(b);  
    }  
}
```

9. BIBLIOGRAPHY

BIBLIOGRAPHY

<https://www.kaggle.com>

<https://www.stackoverflow.com>

<https://www.w3schools.com>

<https://www.google.com>

<https://developer.android.com>

https://en.wikipedia.org/wiki/Main_Page

<https://www.github.com>