NAALAIYA THIRAN PROJECT

PROJECT TITLE

DIGITAL NATURALIST-AI ENABLED TOOL FOR BIO DIVERSITY RESEARCHERS

TEAM MEMBERS

S.LOGESH (212019104020)

V.SANKARANARAYANAN(212019104032)

P.NANDHA KUMAR (212019104023)

N.NITHISH IYAPPAN (212019104027)

Digital Naturalist Project Report

Date	19 November 2022
Team ID	PNT2022TMID36992
Project Name	Digital Naturalist – AI Enabled tool for Biodiversity Researchers

1. INTRODUCTION

Project Overview

This project is used by Field Naturalists to identify the birds, flowers, mammals and other species they see on their hikes, canoe trips and other excursions. This is done by creating a web application which uses a deep learning model which is trained on different species of birds, flowers and mammals and get the prediction of the bird when an image is given.

Purpose

The purpose of this project is to help the Field Naturalists identify the species of birds, flowers, mammals etc. by capturing an image of them. Instead of relying on time consuming approaches like having to refer to a guidebook or having to seek help from experienced ornithologists, the naturalists can use this application to immediately identify the species.

2. LITERATURE SURVEY

Existing problem

Currently, relevant technologies, such as digital cameras, mobile devices, and remote access to databases, are ubiquitously available, accompanied by significant advances in image processing and pattern recognition but since the images are not integrated into the system to provide automated species identification, Field Naturalists and Biodiversity Researchers manually have to check and identify the species by referring guidebooks and contacting experienced specialists which is time-consuming and hampers their researching. This project is made to solve this problem by using Artificial Intelligence.

References

[1] How can Big Data and machine learning benefit environment and water managementhttps://iopscience.iop.org/article/10.1088/1748-9326/ab1b7d

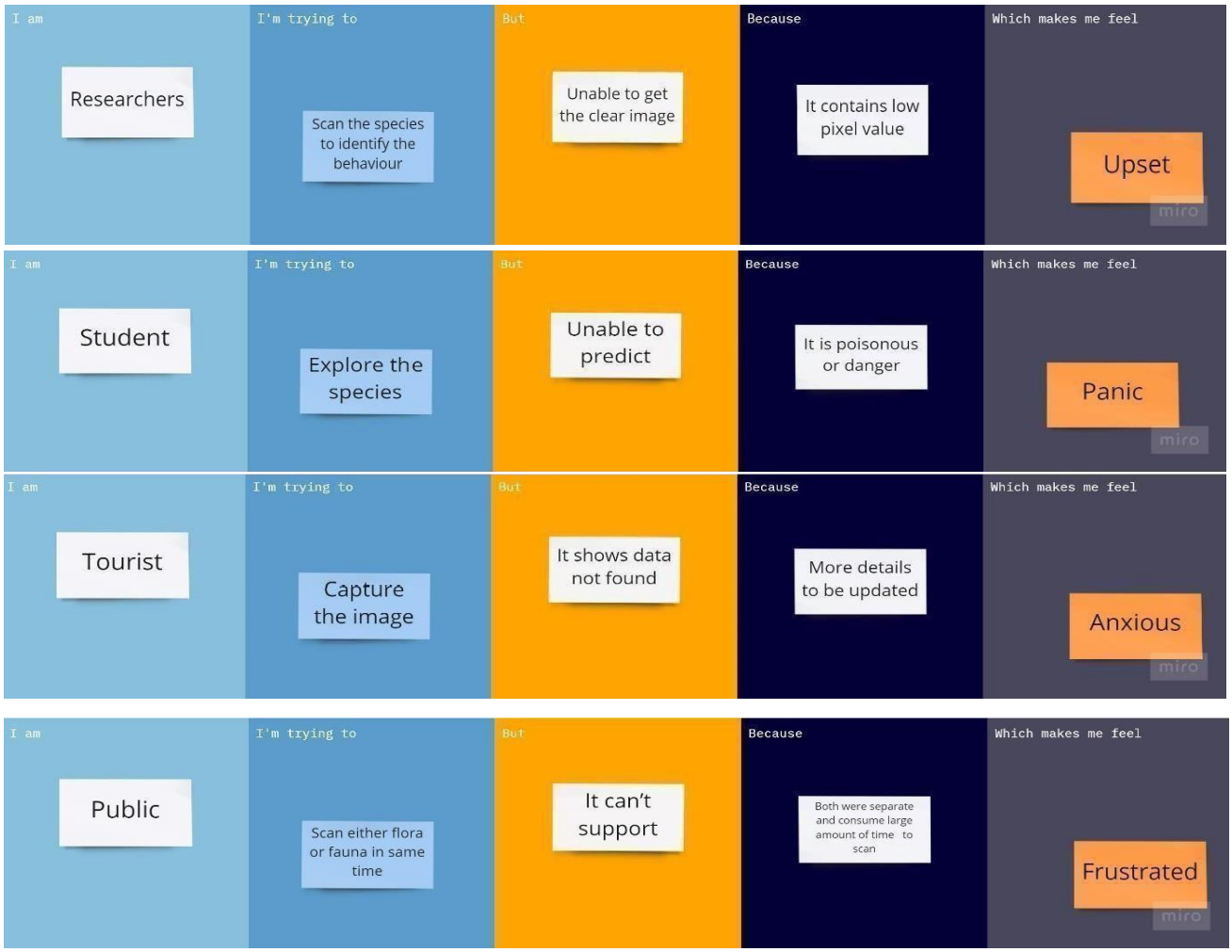
[2] A new subset based deep feature learning method for intelligent fault diagnosis of bearing*https://www.sciencedirect.com/science/article/abs/pii/S0957417418303324*

[3] Machine learning for image based species identification*https://besjournals.onlinelibrary.wiley.com/doi/full/10.1111/2041-210X.13075* [4] Automated plant species identification - Trends and future directions *https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1005993*

[5] A look inside the Pl@ntNet experience *https://doi.org/10.1007/s00530-015-0462-9*

[6] A research tool for long-term and continuous analysis of fish assemblage in coral-reefs usingunderwater camera footage. *https://www.sciencedirect.com/science/article/abs/pii/S1574954113001003?via%3Dihub*

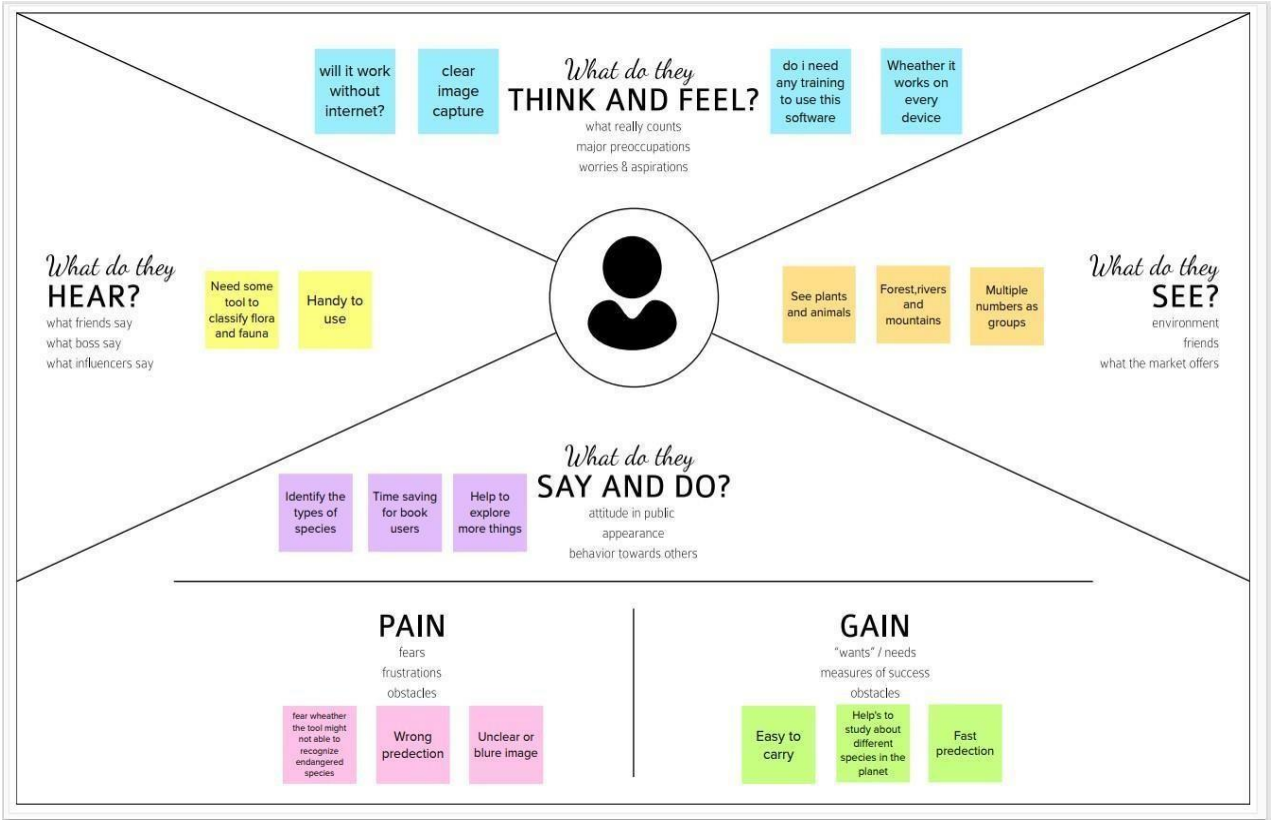
Problem Statement Definition



3. IDEATION & PROPOSED SOLUTION

Empathy Map Canvas:

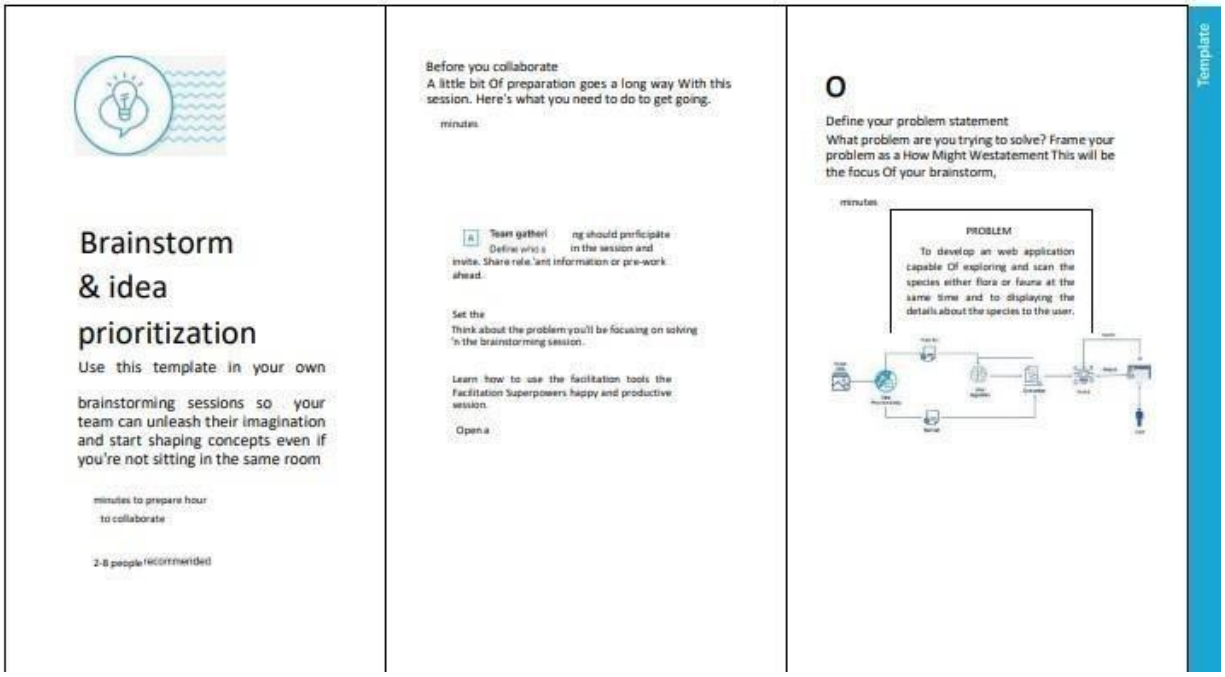
- An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviors and attitudes.
- It is a useful tool to help teams better understand their users.
- Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.



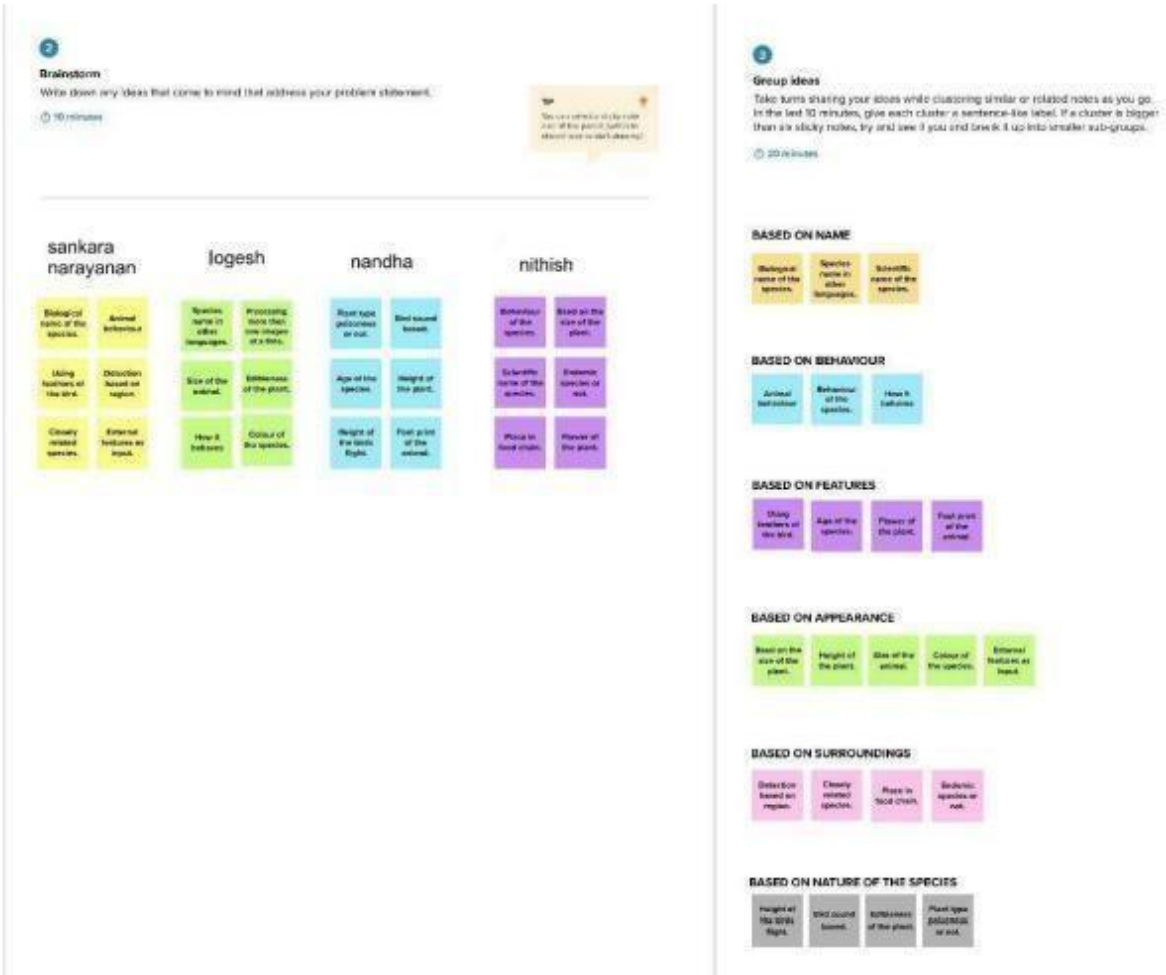
Ideation & Brainstorming

- Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem-solving.
- Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich number of creative solutions.

Step-I: Team Gathering, Collaboration and Select the Problem Statement



Step-2: Brainstorm, Idea Listing and Grouping



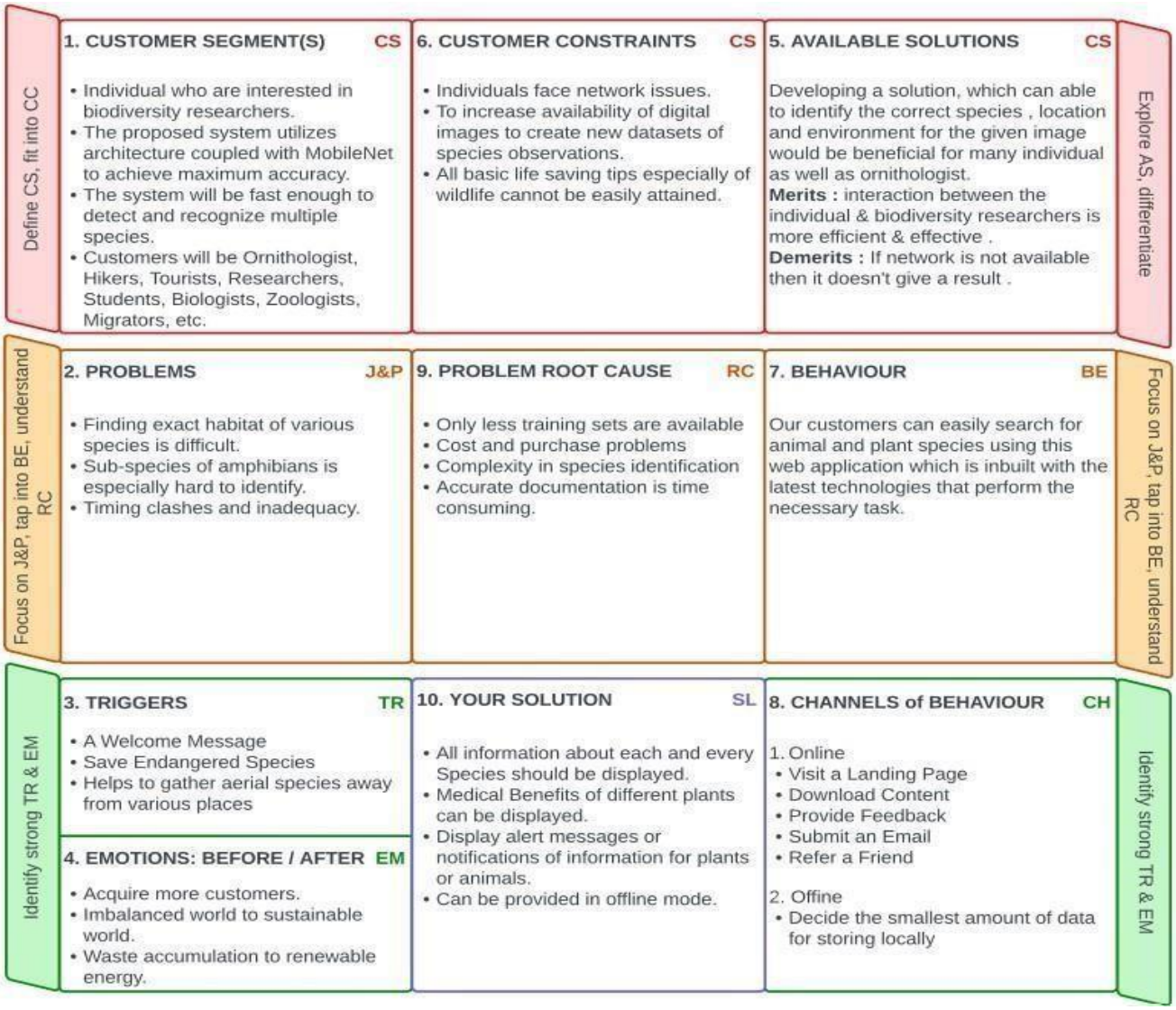
Proposed Solution Template:

Project team shall fill the following information in proposed solution template.

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	To build an efficient AI based image recognition tool which effectively to curb out the following constraints: * To capture the flora and fauna using the AI tool * To provide the information about the flora and fauna resp..
2.	Idea / Solution description	This system is built by using the Image/object recognition and classification using (CNN) Convolutional neural network. By using this system, we can capture the image of any animals and plants and can obtain the information about the flora and fauna at any time
3.	Novelty / Uniqueness	This AI powered chatbot gives a 24*7 efficient automated so that the service can be used anywhere and anytime. This system carries out the visualisations of the interpreted results. It also provides various information regarding the respective flora and fauna.
4.	Social Impact / Customer Satisfaction	The feasibility of implementing this idea is moderate neither easy nor tough because the system needs to satisfy the basic requirements of the customer as well as it should act as a bridge towards achieving high accuracy on predicting and analysing the image taken as

		input and to deliver the output with respective to the input image.
5.	Business Model (Revenue Model)	By using this system, the users can predict and analyse the picture of the animals or plants. In which it results to the visualizing the description of the flora or fauna which taken as input.
6.	Scalability of the Solution	By implementing this system, the people can efficiently and effectively to gain knowledge about the nature they want and they wish to use at anytime. This system can also be integrated with the future technologies

3.4 Problem Solution fit



4. REQUIREMENT ANALYSIS

Functional requirement

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	● Registration through Form ● Registration through Gmail ● Registration through LinkedIN
FR-2	User Confirmation	● Confirmation via Email ● Confirmation via OTP
FR-3	Transactions	● Through UPI, Credit/Debit cards and Net Banking.
FR-4	Authentication	● Through OTP sent to mobile. ● Users created secured passwords.
FR-5	Authorization	● Basic Authorization
FR-6	Administrative functions	● Adding, Updating and Maintaining description ● data about various species.
FR-7	External interfaces	● Easy to access UI ● Community for discussions

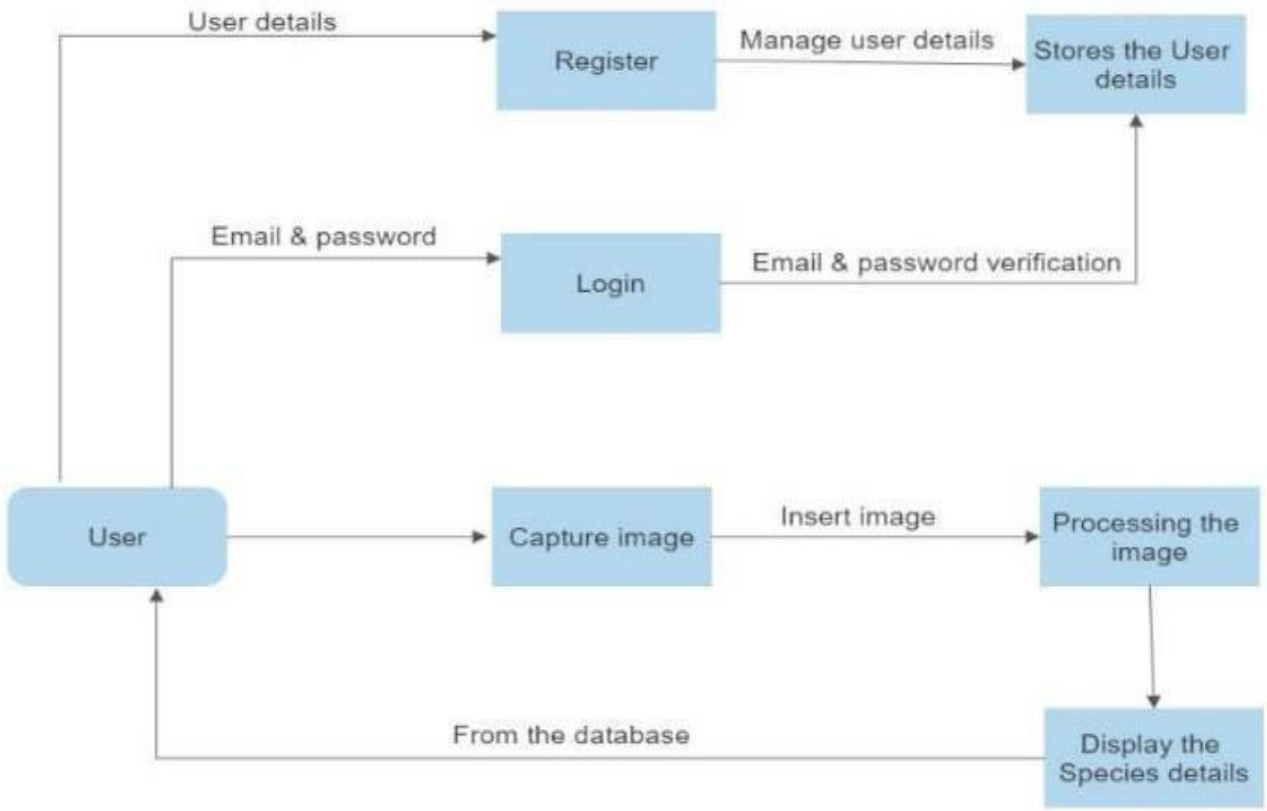
Non-Functional requirements

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Our solution is demanded for scientific researchers Such as Ornithologists , Zoologists in order to predict and analyze flora and fauna.
NFR-2	**Security**	Authentication process involves multilayer securityto make user data and collected data more secure, also to avoid unknown authorization and data integrity issues. Most security methods include Encryption and Authorization.
NFR-3	**Reliability**	Our framework should be reliable to cover wide range of species spanning across various habitats.
NFR-4	**Performance**	Data Augmentation to increase dataset size along with transfer learning to increase accuracy and performance for better working of applications.
NFR-5	**Availability**	Our application possess full-time service (either offline or online) and the dataset is constantly updated.
NFR-6	**Scalability**	Our application supports large number of concurrent users without any hurdles or errors through scaled cloud resources.

5. PROJECT DESIGN

Data Flow Diagrams



Solution & Technical Architecture

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders.
- Define features, development phases, and solution requirements.
- Provide specifications according to which the solution is defined, managed, and delivered.

Solution Architecture Diagram:

Technical Architecture:

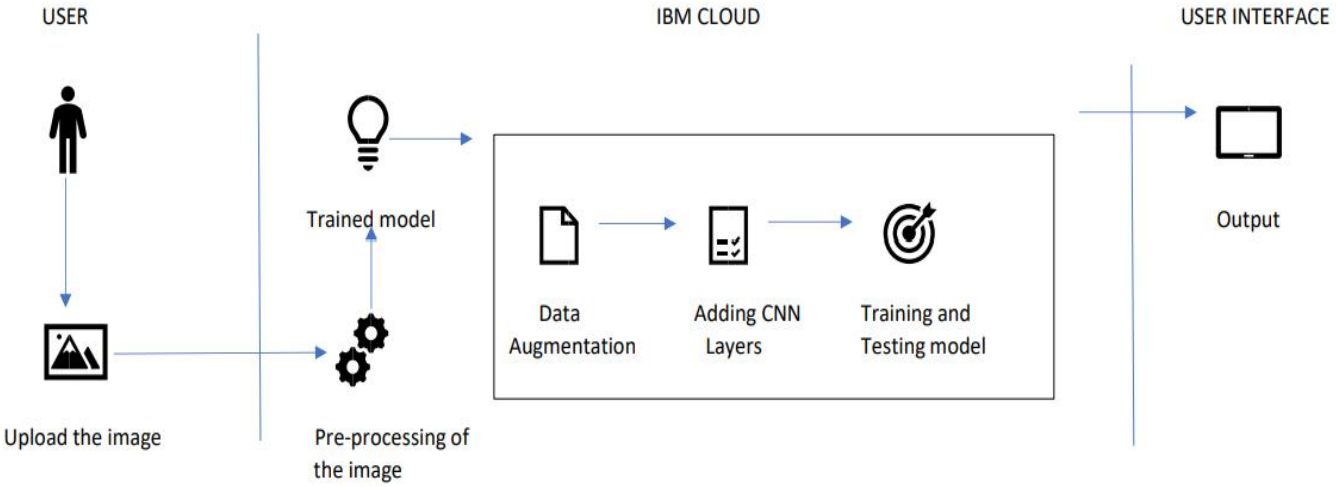


Table-1: Components & Technologies:

S. No	Component	Description	Technology
1.	User Interface	The end user interacts with web application through Web UI	HTML, CSS, JavaScript.
2.	Application Logic	Interpret the input image	Python
3.	Cloud Database	Database Service on Cloud.	IBM DB2, IBM Cloudant.
4.	File Storage	File storage requirements	IBM Block Storage, Local Filesystem.
5.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration:	Local, Cloud Foundry, Kubernetes.

Table-2: Application Characteristics:

S. No	Characteristics	Description	Technology
1.	Open-Source Frameworks	The open-source framework used is python flask	Python flask
2.	Security Implementations	MAC access control is used.	e.g. SHA-256, Encryptions, IAM Controls, OWASP etc
3.	Scalable Architecture	3 – tier architecture	Web Server – HTML, CSS, JavaScript Application Server – Python Database Server – IBM DB2
4.	Availability	Use of Load Balancing to distribute network traffic across servers	IBM Load Balancer
5.	Performance	Design consideration for the performance of the application	IBM Content Delivery Network

6. PROJECT PLANNING & SCHEDULING

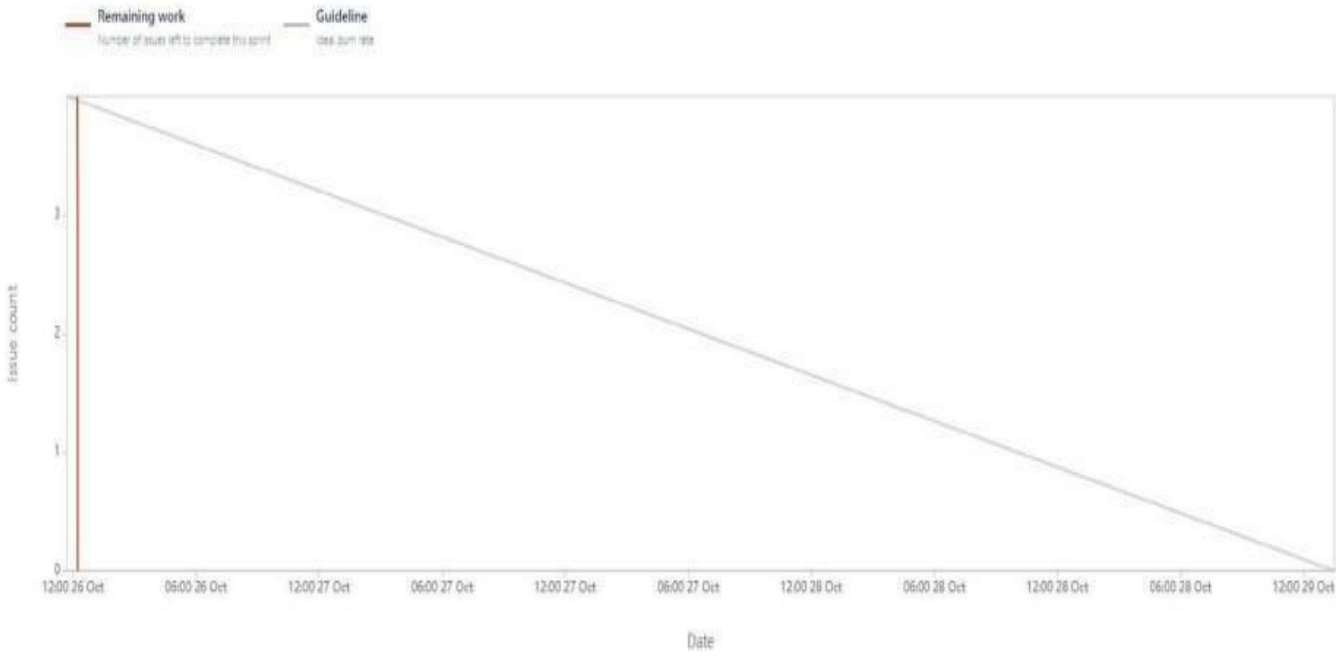
Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Initial Phase	USN-1	Data Collecting and grouping it for processing	4	Medium	SANKARA NARAYANAN
Sprint-1		USN-2	Addition of more data sets to make the model more reliable	3	Medium	LOGESH
Sprint-1		USN-3	Processing the model using CNN algorithm	6	High	NANDHA KUMAR
Sprint-1		USN-4	Evaluating the data sets to verify its correctness and quality	4	High	NITHISH IYAPPAN
Sprint-2	Development Phase	USN-5	Creating an interactive introduction page for the user side	2	Low	SANKARA NARAYANAN
Sprint-2		USN-6	Adding feature in that page to upload the image	3	Medium	LOGESH
Sprint-2		USN-7	Analysis is done on the image and appropriate output is displayed according to the input	4	Medium	NANDHA KUMAR
Sprint-2		USN-8	Importing and using the required API's for model	5	High	NITHISH IYAPPAN
Sprint-3	Implementation Phase	USN-9	Integration of UI & backend – Connecting the UI and backend using API calls	5	Medium	SANKARA NARAYANAN
Sprint-3		USN-10	The model is finally stored in IBM cloud for future use	5	High	LOGESH
Sprint-4	Testing phase	USN-11	Functional testing –The scalability and robustness of the application is measured	6	High	NANDHA KUMAR
Sprint-4		USN-12	Non-Functional testing – Integration and Acceptance tests are carried out	6	High	NITHISH IYAPPAN

Sprint Delivery Schedul

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	17	6 Days	24 Oct 2022	29 Oct 2022	17	29 Oct 2022
Sprint-2	14	6 Days	31 Oct 2022	05 Nov 2022	14	05 Nov 2022
Sprint-3	10	6 Days	07 Nov 2022	12 Nov 2022	10	12 Nov 2022
Sprint-4	12	6 Days	14 Nov 2022	19 Nov 2022	12	19 Nov 2022

Reports from JIRA



1. CODING & SOLUTIONING (Explain the features added in the project along with code)

Feature 1

A CNN-based model which is trained up with the help of a pre-stored dataset of different species and performs with a high accuracy in predicting any new given restricted data to the model and the response/output from the model is delivered through a webpage for the user. Genuinely the model runs on a cloud platform called "IBM cloud" where the input files (i.e) dataset that are necessary for the model to predict properly are stored in the cloud as like the model itself. We used inception net pretrained network to train the model which helps in avoiding the overfitting issues and for efficient computation as well. It is then integrated with the flask application to allow the user to give input image-files to the model via a web page in order to get knowledge about the species that they are looking for.

Feature 2

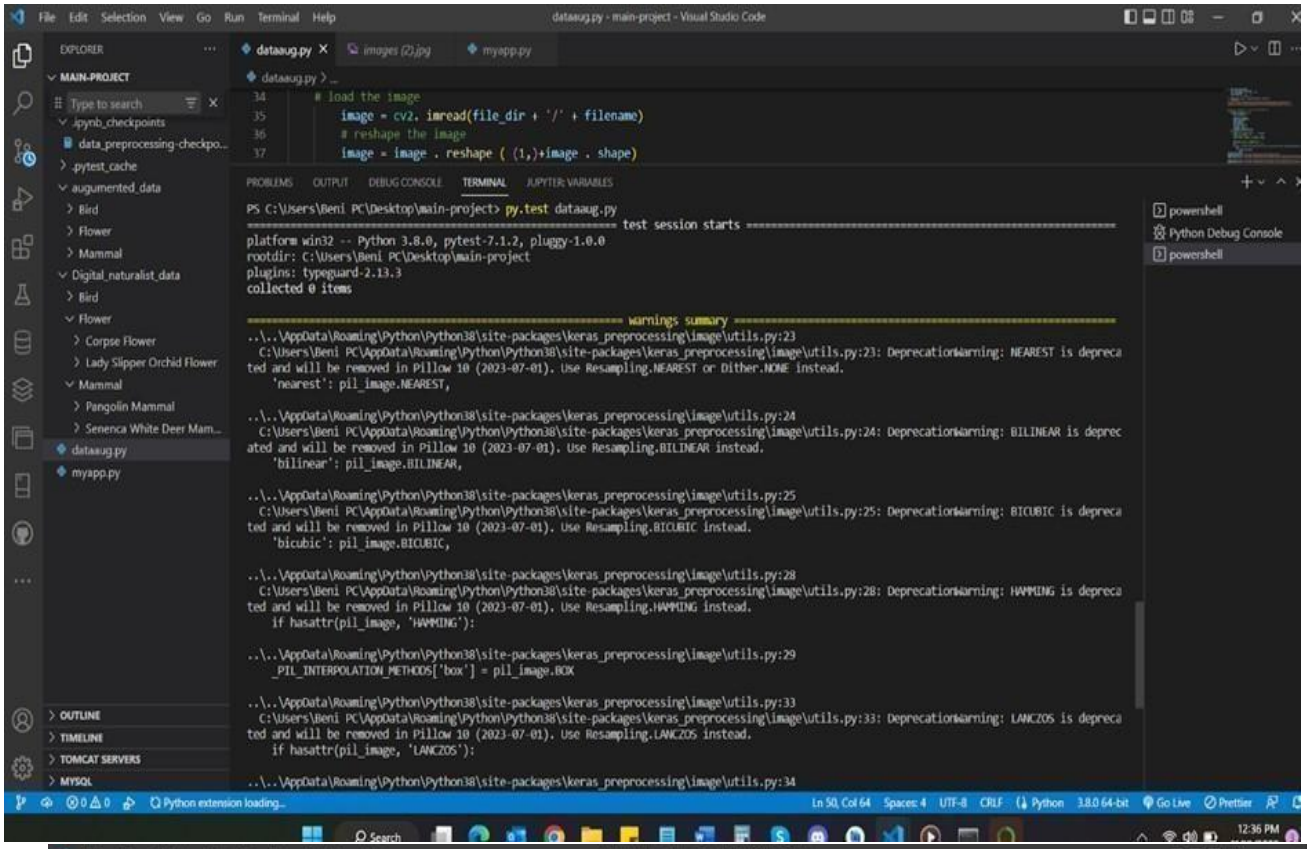
A feature called upload option which is present in the webpage for the purpose of delivering the input image-file from the user to the model for the computation purpose of finding out what exactly the species is. This feature is linked up with a function from flask application whereby when a user clicks on this very upload button then the uploaded image-file is taken to the model where the image-file is stored locally and turned into an image array before the actual computation process begins and later sending back the response/output to the webpage for user's view.

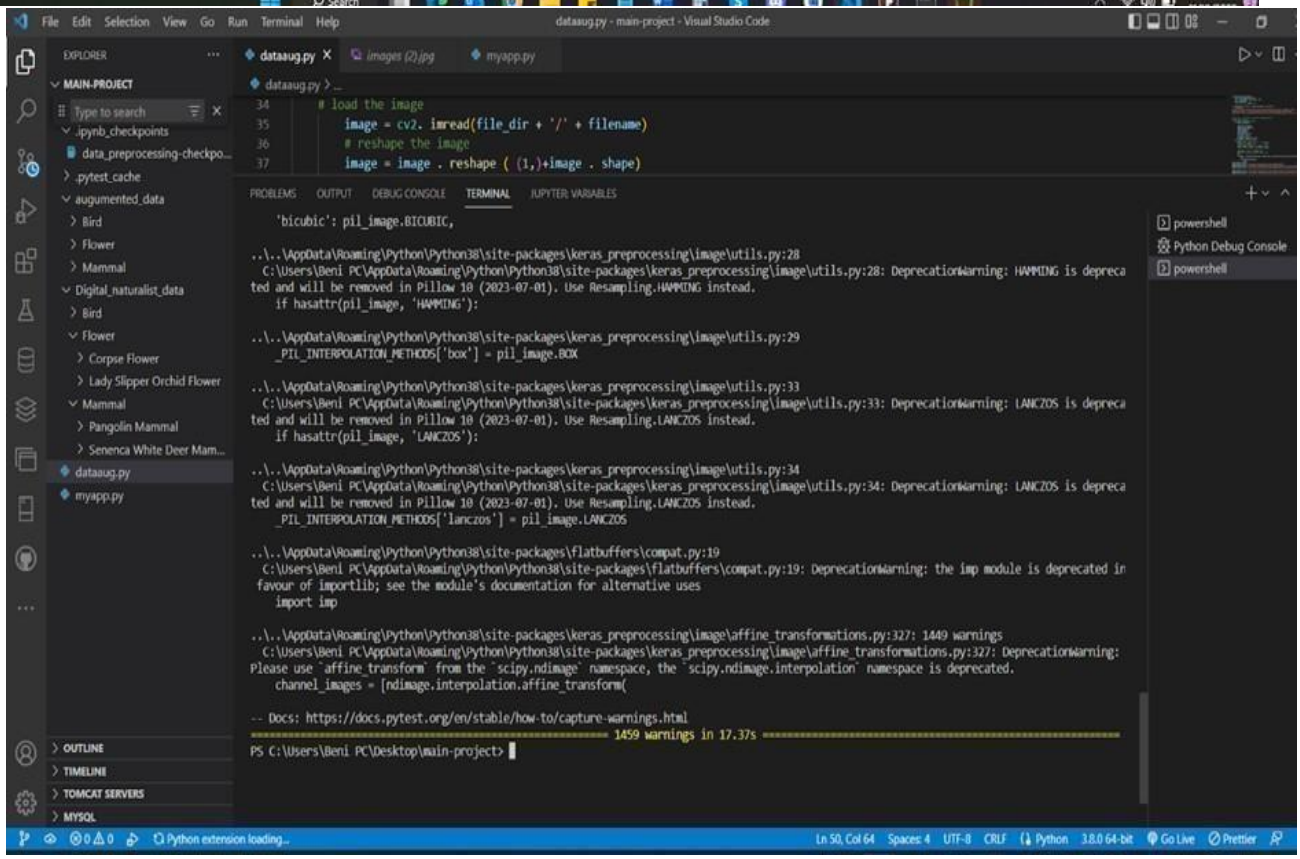
2. TESTING

Test Cases

Resolution	**Severity 1**	**Severity 2**	**Severity 3**	**Severity 4**	**Subtotal**
By Design	3	12	7	4	26
Duplicate	1	0	0	1	2
External	3	2	0	2	7
Fixed	7	8	3	12	30
Not Reproduced	0	1	0	0	1
Skipped	1	0	1	1	3
Won’t Fix	1	3	2	1	7
Totals	16	26	13	21	76

Testing screenshots



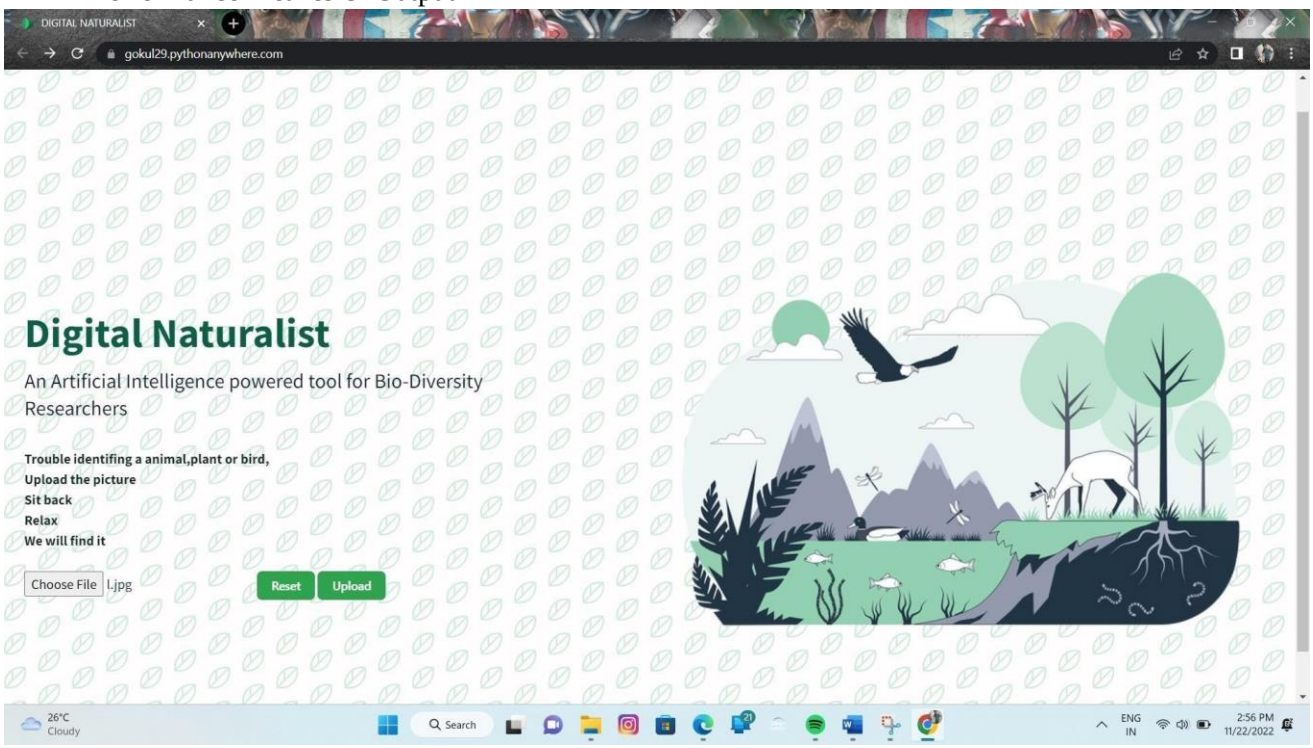


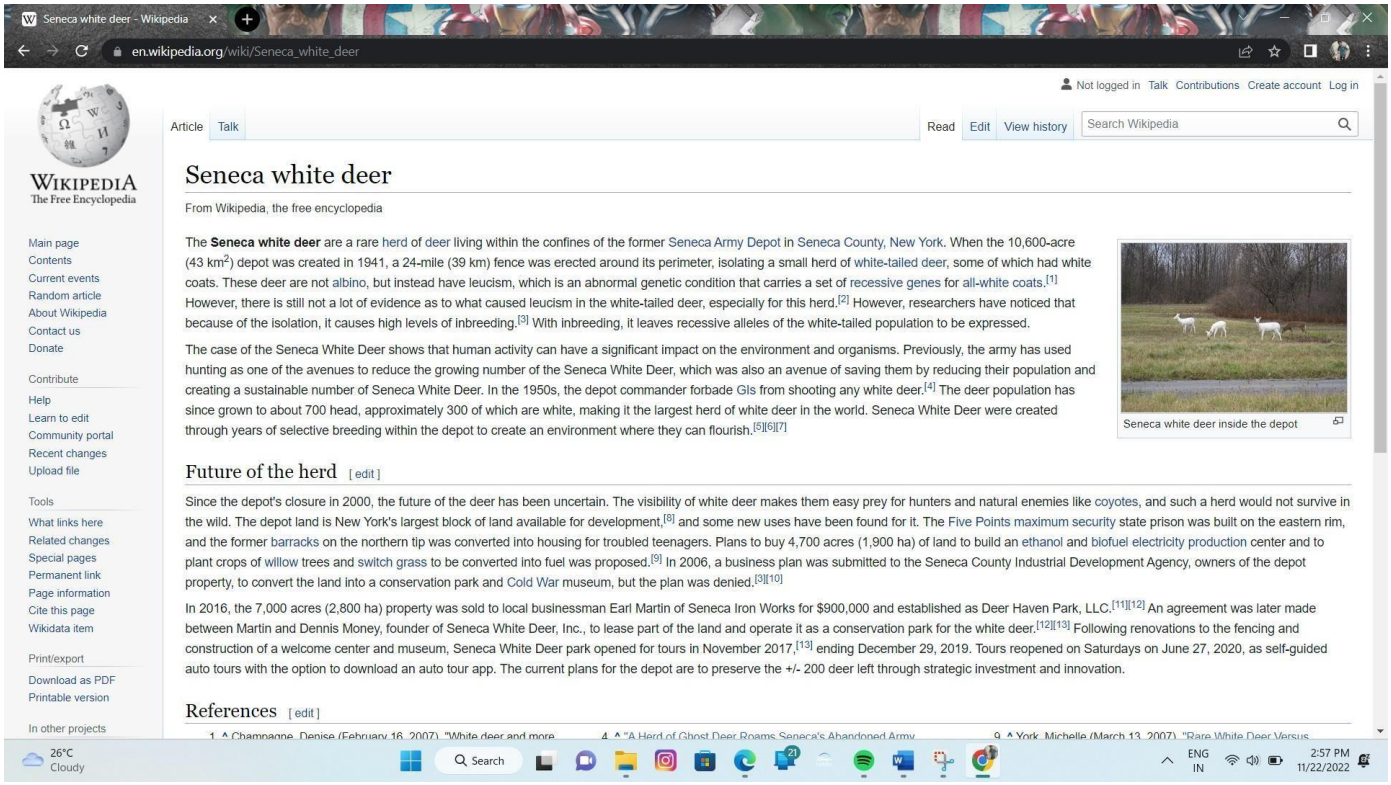
User Acceptance

Section	Test Case	Not Tested	Fail	Pass
Print Engine	5	0	1	4
Client Application	26	0	0	26
Security	1	0	0	1
Outscore Shipping	4	0	0	4
Exception Reporting	26	0	0	26
Final Report Output	3	0	0	3
Version Control	2	0	0	2

3. RESULTS

Performance Metrics & Output





4. ADVANTAGES & DISADVANTAGES

- To understand more about flora and fauna for researchers and students our project will be helpful.
- It recognise what type of animals from its foot tracks and plants from its leaves structure
- Researchers can take notes on new behavior on the spot in software and can analyze it afterwards for research purpose
- It will also display the scientific names of the flora and fauna.
- The app requires an internet connectivity to run which can curb the access in remote areas
- The input device requirement is expensive since cameras with decent quality are required for image classification.
- The app cannot detect species that can camouflage with the environment.

5. CONCLUSION

The app, made with flask uses convolutional neural network (CNN) is trained with predefined dataset and the output is displayed and the app is deployed in IBM Cloud. When the field scientist takes a picture of a flora or fauna using their input device, that image is processed against the trained model and the species are identified. The solution is platform independent, meaning that it can be accessed with any device that satisfies the minimum hardware requirement and can be accessed anywhere with just an input device and internet connection. The modularity of the app enables the developer to include additional features in the future such as fossil detection, route analysis and many more.

6. FUTURE SCOPE

- We can update the model to recognise fossils also through this app ● It can show us which species are endangered white their known count ● It will be able to recognise animals by their sounds.

- In the future we can develop this app where we can show the numbers of endangered species and their count.
- We can bring awareness to the users to protect these animals we can also develop this software as such it can recognise the animals from their sounds and displays the users which animal it is
- They can also keep track on the research animals and their behavior through this app which is helpful to the researchers to analyze their habits

7. APPENDIX Source Code: *dataaug.py*

```
from keras . preprocessing. image import ImageDataGenerator
import cv2
from os import listdir
import time

# Nicely formatted time string to make a note of how much time it takes for augmentation
def hms_string (sec_elapsed) :
    h=int(sec_elapsed / (60 * 60) )
    m=int ((sec_elapsed % (60 * 60) ) / 60)
    s=sec_elapsed%60
    return f"{h}: {m}:{round(s, 1)}"

def augment_data (file_dir, n_generated_samples, save_to_dir) :
    """Arguments:
file_dir: A string representing the directory where images that we want to augment are found.
i-generated samples.  A string representing the number of generated samples using the giv
save_to_dir: A string representing the directory in which the generated images will be saved."""



#from keras . preprocessing. image import ImageDataGenerator
#from os import listdir
    data_gen = ImageDataGenerator(
        rotation_range=30,

        width_shift_range=0.1,
        height_shift_range=0.15,
        shear_range=0-25,
        zoom_range=0.2,
```

```
        horizontal_flip=True,
        vertical_flip=False,

        fill_mode= 'nearest' ,

        brightness_range= (0.5, 1.2))

    for filename in listdir(file_dir) :

    # load the image
```

```
        image = cv2. imread(file_dir + '/' + filename)

        # reshape the image

        image = image . reshape ( (1,)+image . shape)

        # prefix of the names for the generated sampels.


        save_prefix = 'aug_' + filename [:-4]

        #generate 'n_generated_samples' sample images

        i=0

                                        for batch in data_gen.flow(x=image, batch_size=1,
    save_to_dir=save_to_dir,save_prefix=save_prefix, save_format='jpg' ) :

            i+=1 if
            i>n_generated_samples :

                break

start_time=time.time()

augumented_data_path='C:/Users/Beni PC/Desktop/main-project/augumented_data/'

augment_data(file_dir='C:/Users/Beni PC/Desktop/main-project/Digital_naturalist_data/Bird/Great
    Indian                                                                    Bustard
    Bird',n_generated_samples=8,save_to_dir=augumented_data_path+'Bird/GIB_AUG')

augment_data(file_dir='C:/Users/Beni PC/Desktop/main-project/Digital_naturalist_data/Bird/Spoon
    Billed                                                                    Sandpiper
    Bird',n_generated_samples=8,save_to_dir=augumented_data_path+'Bird/SPS_AUG')


augment_data(file_dir='C:/Users/Beni
    PC/Desktop/main-project/Digital_naturalist_data/Flower/Corpse
    Flower',n_generated_samples=0,save_to_dir=augumented_data_path+'Flower/Corpse_AUG')

augment_data(file_dir='C:/Users/Beni PC/Desktop/main-project/Digital_naturalist_data/Flower/Lady
    Slipper                                                                   Orchid
```

```
    Flower',n_generated_samples=0,save_to_dir=augumented_data_path+'Flower/LS_Orchid_AUG')


augment_data(file_dir='C:/Users/Beni
    PC/Desktop/main-project/Digital_naturalist_data/Mammal/Pangolin
    Mammal',n_generated_samples=0,save_to_dir=augumented_data_path+'Mammal/Pangolin_AU
    G')
augment_data(file_dir='C:/Users/Beni
    PC/Desktop/main-project/Digital_naturalist_data/Mammal/Senenca White Deer
    Mammal',n_generated_samples=0,save_to_dir=augumented_data_path+'Mammal/SW_Deer_AU
    G')
```

app.py

```
import numpy as np

#Importing libraries required for the model

import tensorflow as tf

import keras

import keras.backend as K


from tensorflow.keras.optimizers import SGD, Adam, Adagrad, RMSprop

from keras.applications import *

from keras.preprocessing import *

from keras.preprocessing . image import ImageDataGenerator, array_to_img, img_to_array, load_img

from keras.callbacks import EarlyStopping, ModelCheckpoint
from keras . models import Sequential

from keras . layers import Dense, Conv2D, MaxPool2D, Flatten, Activation, BatchNormalization ,
    Dropout

from keras . utils . np_utils import to_categorical

from sklearn . model_selection import train_test_split


#For plotting charts used for data visualizations

import matplotlib . pyplot as pit


#Libraries for Locating and loading data

import glob
```

```
from PIL import Image
import os

from os import listdir

#Setting path to our dataset folder

dirName = 'C:/Users/Beni PC/Desktop/main-project/Digital_naturalist_data'

folders = listdir (dirName)

#Getting the names for all the folders containing data
def getListOfFiles (dirName) :



# create a list of sub directories and files (if any)

# names in the given directory

    listOfFile = os . listdir(dirName)

    allFiles = list( )
```

```
    for fol_name in listOfFile:
        fullPath = os . path . join (dirName, fol_name)
        allFiles . append ( fullPath)
    return allFiles


Folders = getListOfFiles (dirName)
len (Folders)
subfolders = []
for num in range (len (Folders ) ) :
    sub_fols = getListOfFiles (Folders[num] )
    subfolders+=sub_fols
#Now, the subfolders contains the address to all our data folders for each class
subfolders
#Loading the data and pre processing it to make it in trainable format
#1III


#X data will includes the data generated for each image
#Y data will include a id no, unique for every different species, so are having 6 classes
#there for we will get 6 ids = [0 , 1, 2,3,4,5]
#That will be tha label we're classifying.
X_data=[]
Y_data=[]
```

```
id_no=0

#to make a list of tuples, where we'll store the info about the image, category and species
found = []
#itering in all folders under Augmented data folder
for paths in subfolders:
#setting folder path for each unique class and category
    files = glob . glob (paths + "/*. jpg")
    #adding tuples to the list that contain folder name and subfolder name
    found . append((paths . split('\\' ) [-2] , paths . split( '\\' ) [-1]))


#itering all files under the folder one by one
    for myFile in files:
        img = Image. open (myFile)
        img=img.resize( (224, 224) , Image. ANTIALIAS) # resizes image without ratio
        #convert the images to numpy arrays


        img = np . array (img)
        if img . shape == ( 224, 224, 3) :
    # Add the numpy image to matrix with all data
            X_data . append (img)
            Y_data . append (id_no)
    id_no+=1
#to see our
    print (X_data)
    print (Y_data)

    #converting lists to np arrays again
    X = np . array (X_data)
    Y = np. array (Y_data)
    # Print shapes to see if they are correct
    print ( "x-shape" , X. shape, "y shape", Y. shape)

    X = X.astype ( 'float32' ) /255.0
    #The keras Library offers a function called to_categorical() that you can use to one hot encode
    #so we can use the to categorical () function directly
    y_cat = to_categorical (Y_data, len(subfolders) )
    print ("X shape" , X, "y_cat shape", y_cat)
    print ("X shape" , X. shape, "y_cat shape" , y_cat . shape)
```

```
X_train,X_test,y_train,y_test=train_test_split(X,y_cat,test_size=0.2)
print("The model has "+str(len(X_train))+" inputs")
```

GitHub Link: **https://github.com/IBM-EPBL/IBM-Project-30893-1660191957**

Project Demo Link: https://youtu.be/2xlL0SIw384