1. INTRODUCTION

1.1 Project Overview

A naturalist is someone who studies the patterns of nature, identifies a different kind of flora and fauna in nature. Being able to identify the flora and fauna around us often leads to an interest in protecting wild spaces and collecting and sharing information about the species we see on our travels is very useful for conservation groups like NCC. When venturing into the woods, field naturalists usually rely on common approaches like always carrying a guidebook around everywhere or seeking help from experienced ornithologists.

There should be a handy tool for them to capture, identify and share the beauty to the outside world. Field naturalists can only use this web app from anywhere to identify the birds, flowers, mammals, and other species they see on their hikes, canoe trips and other excursions. In this project, we are creating a web application which uses a deep learning model, trained on different species of birds, flowers and mammals (2 subclasses in each for a quick understanding) and get the prediction of the bird when an image is been given.

1.2 Purpose

This Project aims to identify a species in a forest or in any other place, we need to carry a heavy book or seek a professional like botanist or zoologist or an ornithologist, but there should be a handy tool for them to capture, identify and share the beauty to the outside world.

PROJECT OBJECTIVES:

EMPATHIZE:

These techniques have profoundly transformed our ability to extract information from visual data. AI techniques have been applied for a long time in security and industrial domains, for example, in iris recognition or the detection of faulty objects in manufacturing.

They were nevertheless only recently made more widely accessible after their use in smartphone apps for face recognition and song identification.

Combined with increasing access to cloud-based computation, AI techniques can now automatically analyze hundreds of thousands of visual data every day.

APPLICATION OF AI:

To biological recording have to date typically focused on active sampling, that is, images collected specifically for the purpose of recording wildlife (e.g., wildlife recording apps or camera traps). However, this has neglected large amounts of image data that are not collected for the purposes of biological recording, but which nonetheless may contain useful information about biodiversity.

This includes social media imagery (e.g., Flickr and Instagram), CCTV, and imagery collected along linear infrastructure (e.g., Google Streetview). These unexploited image data could be rapidly analyzed using "AI naturalists" designed to locate potential images of biodiversity and classify what they see.

LITERATURE REVEIW

Survey 1:

Simon Haykin (1994)

'Bird classification using CNN'

This work presents a scenario with classification of birds using CNN technique based on color features. They used color images of birds with almost similar types. Image segmentation is carried in various stages. At first, the pixels are arranged and segmented on the basis of edges and

spatial segmentation, where clustering is done. Next, the blocks are segmented using edge detection. The computational efficiency increases for image and training becomes easier. This approach provides with better and robust results for different images. Here they took sparrow for the case study and evaluated the features of it using the steps up listed. Their experimental results classify the effectiveness of proposed approach to improve the segmentation quality in aspects of precision and computational time.

Survey 2:

Paul Viola, Michael Jones (2001)

'Classification and Grading of Image Using Texture Based Block-Wise Local Binary Patterns'

Paul Viola, Michael Jones et al., used global textural feature viz., Local Binary Pattern for feature extraction. Initially, an image is divided into k number of blocks. Subsequently, the texture feature is extracted from each k blocks of the image. The k value is varied and has been fixed empirically. In their approach experimentation purpose, the bird dataset is created using 4 different classes and experimentation is done for whole image and also with different blocks like 2, 4 and 8. Grading of Bird is done using Support Vector Machine classifier. Finally, the performance of the grading system is evaluated through metrics like accuracy, precision, recall and F-measure computed from the confusion matrix. Their experimental results show that most promising result is obtained for 8 blocks of the image.

Survey 3:

Gary Bradski and Adrian Kaehler (2008)

'Texture Classification from Random Features'

In this research they presented an approach for texture classification based on random projection, suitable for large texture database applications. A small set of random features are extracted from local image patches and those features are embedded into a bag-of-words model to perform texture classification.

Survey 4:

Schmid Huber, J. (2015)

‘Adapted approach for Species Classification’

In this work, an adaptive approach for the identification of species is proposed and experimentally validated. Image processing technique is followed. In the first step K-Means clustering is used for image segmentation, in the second step some state of art features is extracted from segmented image, and finally images are classified under one of the classes by using multi-class support vector machine. The classification accuracy is achieved up to 89%.

Survey 5:

Haibing Wu and Xiaodong Gu (2015)

‘Detection And Classification of images using Detection Line’

In this study, they present an application of neural networks and image processing techniques for detecting and classifying images. Images were segmented by a detection line (DL) method. Six geometric features (i.e., the principal axis length, the secondary axis length, axis number, area, perimeter and compactness of the image), 3 color features (i.e., the mean gray level of image on the R, G, and B bands. The methodology presented herein effectively works for classifying image to an accuracy of 90.9%.

REFERENCES

1. Simon Haykin, “Bird classification using CNN: a comprehensive foundation,” Prentice Hall PTR, 1994.
2. Paul Viola, Michael Jones, “Classification and Grading of Image Using Texture Based Block-Wise Local Binary Patterns” CVPR (1) 1 (2001), 511–518, 2001.
3. Gary Bradski and Adrian Kaehler. “Texture Classification from Random Features”, 2008.
4. Schmid Huber J, “Adapted approach for Species Classification: An Overview Neural Networks” 61: 85-117, 2015.
5. Haibing Wu and Xiaodong Gu, “Detection and Classification of images using Detection Line” 71,1–10, 2015.

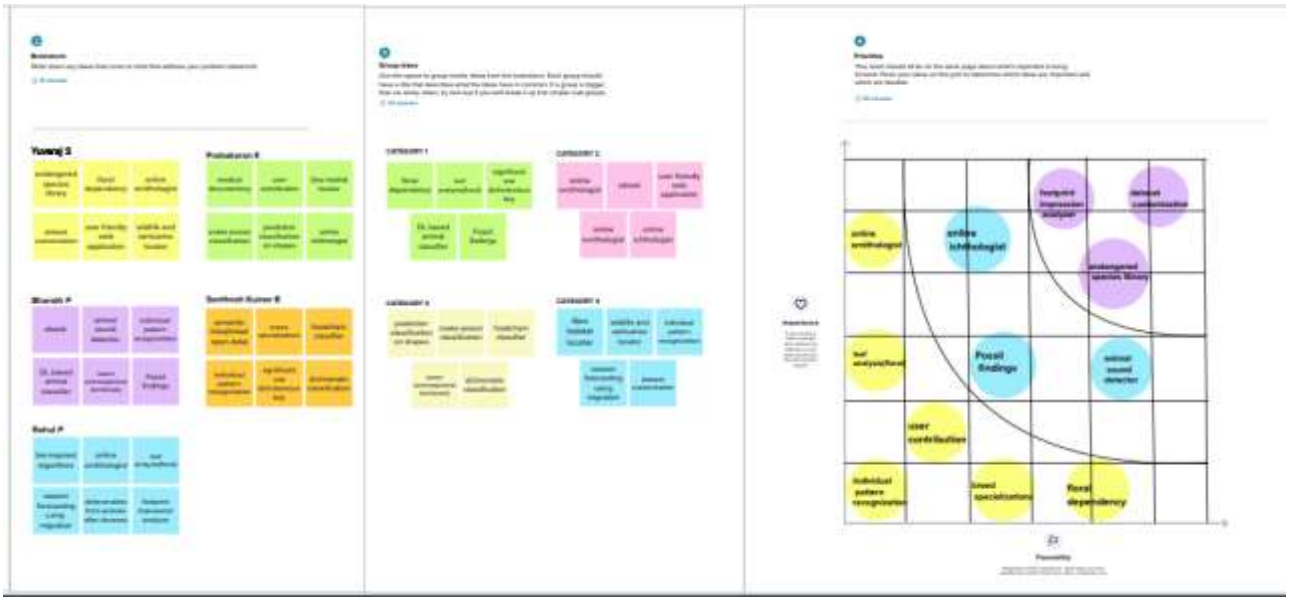
2.3 Problem Statement Definition

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas


Carrying guidebooks and using other conventional methods is tiring
Inability to identify flora or fauna
What do they THINK AND FEEL?
what really counts
major preoccupations
worries & aspirations
Getting accurate information about the identified species
What do they HEAR?
what friends say
what boss say
what influencers say
You should bring along a naturalist while venturing into the forest
You should learn more about the flora and fauna and be able to identify them yourself
You should bring along the appropriate resources to identify the species
Various types of flora and fauna
Accurate and Timely identification of the species
Sharing information and pictures of the identified species
What do they SEE?
environment
friends
what the market offers
Easy and quickly accessible web application
Automatically detect and identify the species of the flora or fauna
What do they SAY AND DO?
attitude in public
appearance
behavior towards others
Accurate prediction
Proper information about the species identified
PAIN
fears
frustrations
obstacles
Reduction in accuracy
Short dataset
GAIN
"wants" / needs
measures of success
obstacles
Ease of Access
High accuracy of prediction
Fast Results
Proper display of information

3.2 Ideation & Brainstorming


Yuvraj S
Prathakaran R
Bharath P
Santhosh Kumar R
Rahul P
Fossil findings
online ornithologist
online ichthiologist
underground electric library
Fossil dependency

3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Identification of species in forest is very difficult , unnamed or extinct species are very typical to found .Researchers and trekkers cannot carry a big catalogue so this enabled AI tool will be helpful to find the accurate species
2.	Idea / Solution description	A system is built by using image object recognition and classification using Convolution neural network.so we can capture the image of any animals and plants and obtain information of the flora,fauna,unnamed and extinct species.
3.	Novelty / Uniqueness	By using the transfer learning in pew trained models to increase accuracy and training time along the data augmentation .
4.	Social Impact / Customer Satisfaction	The model as enhanced innovations like indications of nearest creatures and also provide bookmarks and reference images
5.	Business Model (Revenue Model)	The model contains some review ads with connection of network this tool can be accessed using Global positioning system
6.	Scalability of the Solution	It is used upgrade knowledge for students to be aware of biodiversity it can also be called compact Wikipedia.

3.4 Problem Solution fit

4. REQUIREMENT ANALYSIS

4.1 Functional requirements

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 Google API
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. • User 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

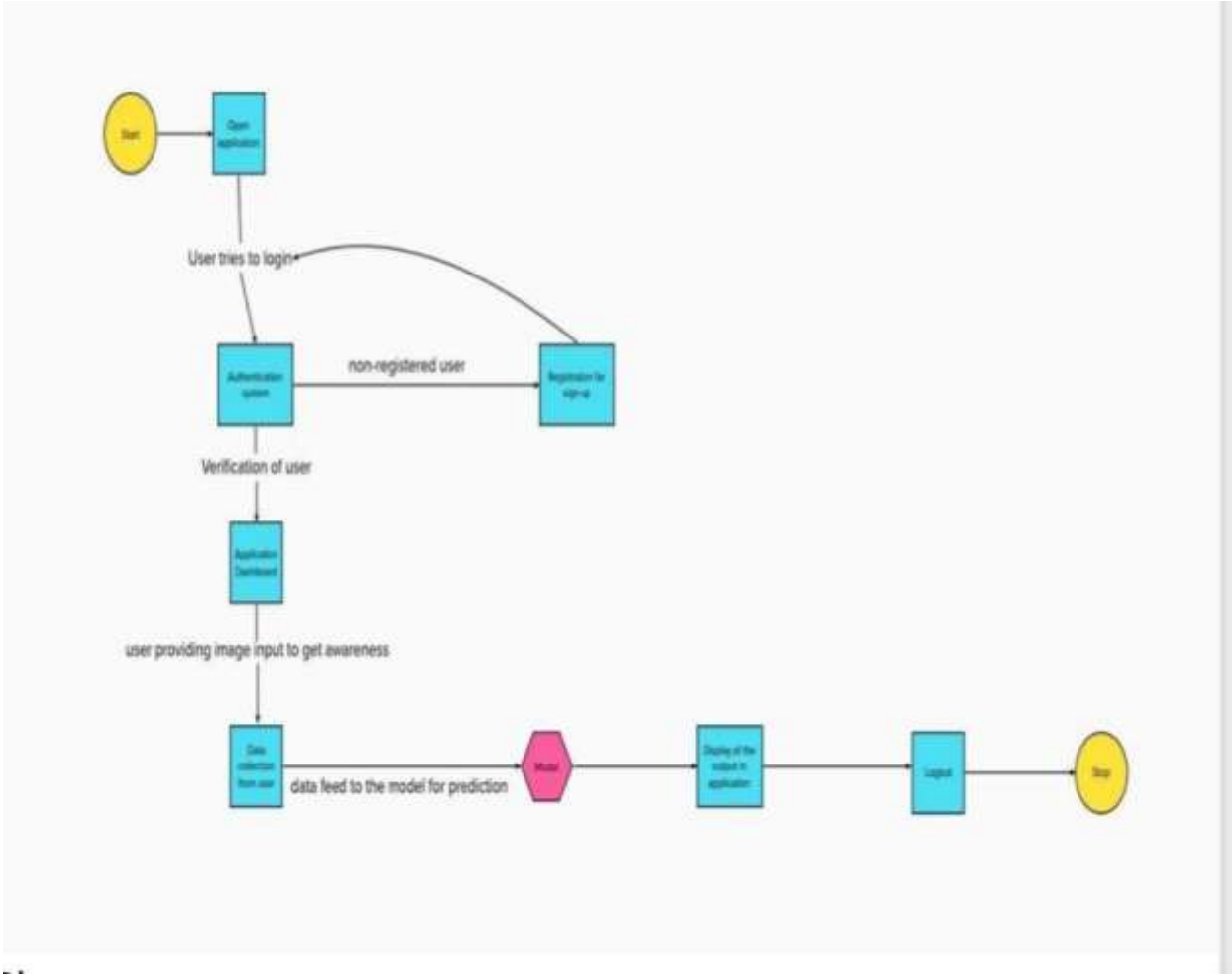
4.2 Non-Functional requirements

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 analyse about flora and fauna.
NFR-2	**Security**	Authentication process involves multilayer security to make user data and collected data more secured, 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 application.
NFR-5	**Availability**	Our application possess full-time service (either offline or online) and dataset is constantly updated.

5. PROJECT DESIGN

5.1 Data Flow Diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. Data flowcharts can range from simple, even handdrawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled.

They can be used to analyze an existing system or model a new one.

5.2 Solution & Technical Architecture

Solution architecture provides the ground for software development projects by tailoring IT solutions to specific business needs and defining their functional requirements and stages of implementation. It is comprised of many subprocesses that draw guidance from various enterprise architecture viewpoints.

In solution architecture, the client needs are expanded to business needs that in one way or another are related to technology. These needs usually crystallize through reassessing existing systems and finding out how they benefit or harm the organization in the long run. Solution architecture can be seen as a support system that provides structure and reduces the scope of complexity when developing and rolling out new systems and applications.

Technical Architecture (TA) is a form of IT architecture that is used to design computer systems. It involves the development of a technical blueprint with regard to the arrangement, interaction, and interdependence of all elements so that system-relevant requirements are met.

This digital transition required not only skilled developing teams but first and foremost IT architects. In their roles as IT strategists and planners, they map out a target architecture and make sure that all IT decisions align with business goals and requirements. This is largely due to the highly dynamic nature of IT, and its widespread adoption throughout all industries and businesses that have developed their own practices.

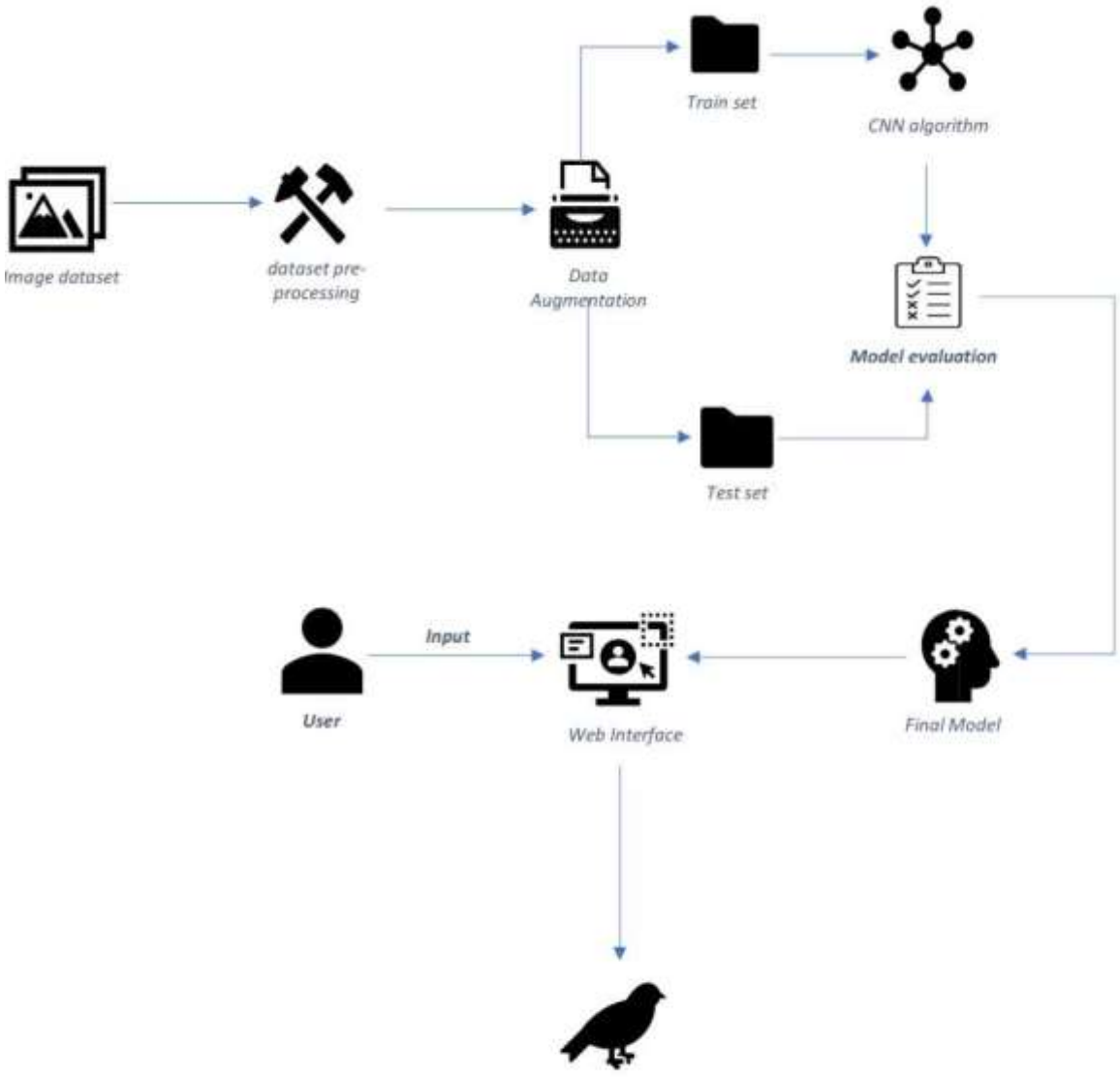


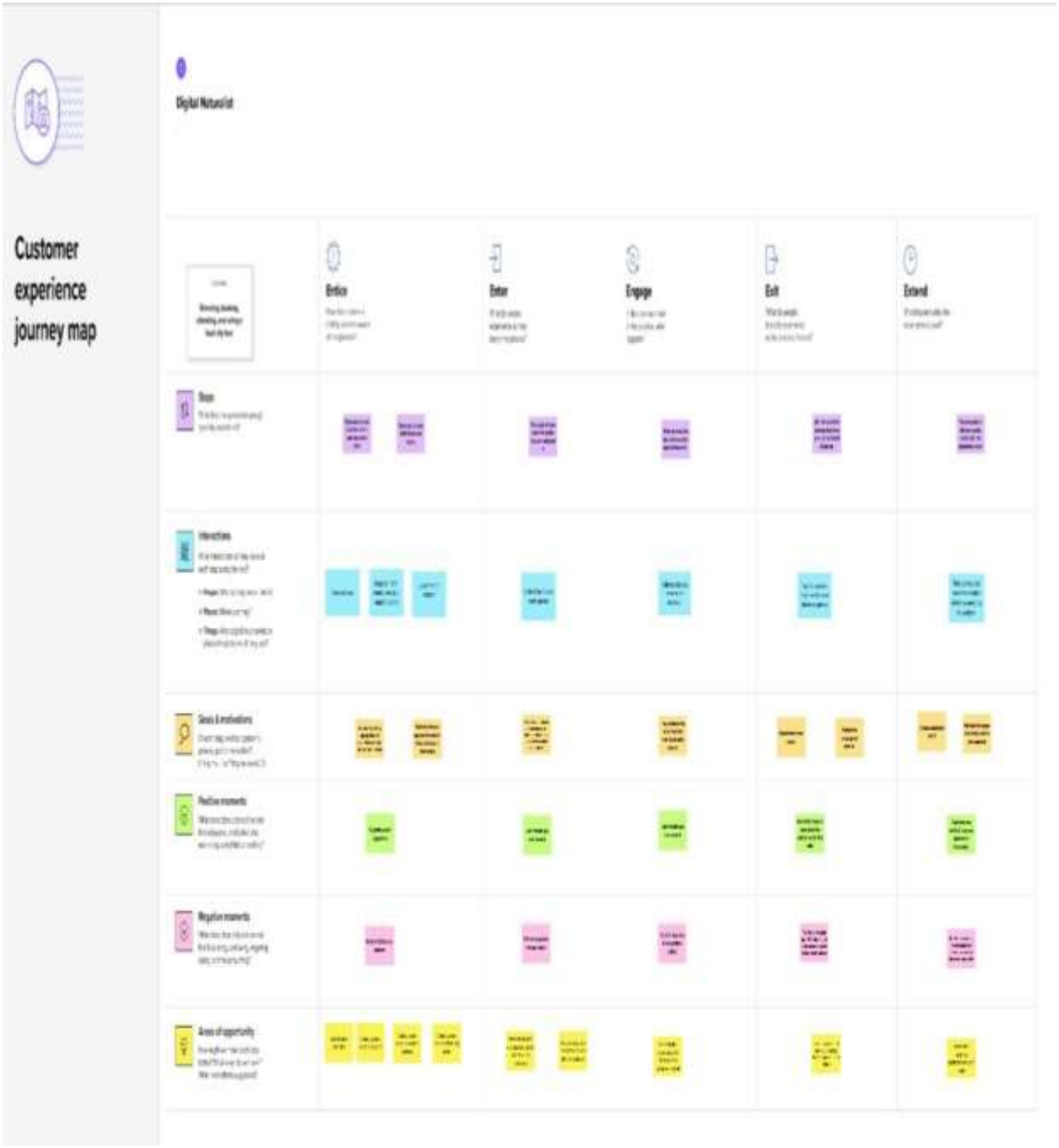
Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	Web UI or Website or Web app	HTML, CSS, React.js
2.	Application Logic-1	Model building and training	PY
3.	Application Logic-2	Getting image or text data from user for prediction	IBM Watson STT service
4.	Application Logic-3	Fetch the relevant data from the database andproject them to user	IBM Watson Assistant
5.	Database	Image and text data of all the species along withdetailed view of each species	NoSQL (MongoDB)
6.	Cloud Database	Fetch data from database and feed them to model for prediction and also used to retrieve the data required for user.	IBM Cloudant
7.	File Storage	Image data, login credentials, code (backendand frontend) and API keys	IBM Block Storage
8.	External API-1	To get data from the database when user givethe image input	IBM Storage API
9.	External API-2	To get the username and password of the specific user	Secure Authentication API
10.	Machine Learning Model	To predict the species (flora or fauna) through the image input and also it gives detailed view of the particular species	SDIM
11.	Infrastructure (Server / Cloud)	To deploy our application in cloud server	Cloud Foundry

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Application is built by using flask	WSGI fiamewoik (Web Seivei Gateway Inteiface)
2.	Security Implementations	For authenticating the user data and protectingthe data about species in database	SHA-256 and Encryptions
3.	Scalable Architecture	To scale our application in server side by supporting clients including desktop browsers,mobile browsers etc	IBM Auto Scaling
4.	Availabil ity	To make application available both online andoffline and also 24/7 service.	IBM Cloud load balancer
5.	Performance	Designing an application that can handle wide range of requests at a time without any delay and to provide accuracy in pred	IBM instance

5.3: Customer Journey Map



6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Model Building Phase	USN-1	Collecting and digitalizing data for analysis	3	Medium	Sivaranjani T
Sprint-1	Model Building Phase	USN-2	Data Augmentation and Feature Engineering	4	High	Vedhavalli V
Sprint-1	Model Building Phase	USN-3	Building the model using transfer learning approach	4	High	Nirosha M
Sprint-1	Model Building Phase	USN-4	Evaluating the model to check the accuracy and precision	4	High	Sivaranjani T
Sprint-1	Model Building Phase	USN-5	Class Prediction	3	Medium	Vedhavalli V
Sprint-2	Development Phase	USN-6	User database creation – contains the details of user	4	High	Nirosha M
Sprint-2	Development Phase	USN-7	Web page Creation	4	High	Sivaranjani T
Sprint-2	Development Phase	USN-8	Login and register page creation - Login through email and password along with otp verification	3	Medium	Vedhavalli V
Sprint-3	Development Phase	USN-9	Area to obtain user input	3	Medium	Sivaranjani T
Sprint-3	Development Phase	USN-10	Model loading - API creation using flask.	4	High	Vedhavalli V
Sprint	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Story Points**	**Priority**	**Team Members**
Sprint-3	Development Phase	USN-11	Prediction page creation – shows prediction for user input along with description about the species	2	Low	Vedhavalli V
Sprint-4	Deployment Phase	USN-12	Connecting the frontend and backend using API calls	4	High	Sivaranjani T
Sprint-4	Deployment Phase	USN-13	Cloud deployment – Deployment of application using IBM cloud	4	High	Vedhavalli V
Sprint-4	Testing Phase	USN-14	Functional testing – Checking scalability and robustness of the application	3	Medium	Nirosha M
Sprint-4	Testing Phase	USN-15	Non-functional testing – Checking for user acceptance and integration	3	Medium	Sivaranjani T

6.2 Sprint Delivery Schedule

Project Tracker, Velocity & Burndown Chart:

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	18	6 Days	24 Oct 2022	29 Oct 2022	18	30 Oct 2022
Sprint-2	11	6 Days	31 Oct 2022	05 Nov 2022	11	5 Nov 2022
Sprint-3	9	6 Days	07 Nov 2022	12 Nov 2022	9	10 Nov 2022
Sprint-4	14	6 Days	14 Nov 2022	19 Nov 2022	14	15 Nov 2022

6.3 Reports from JIRA

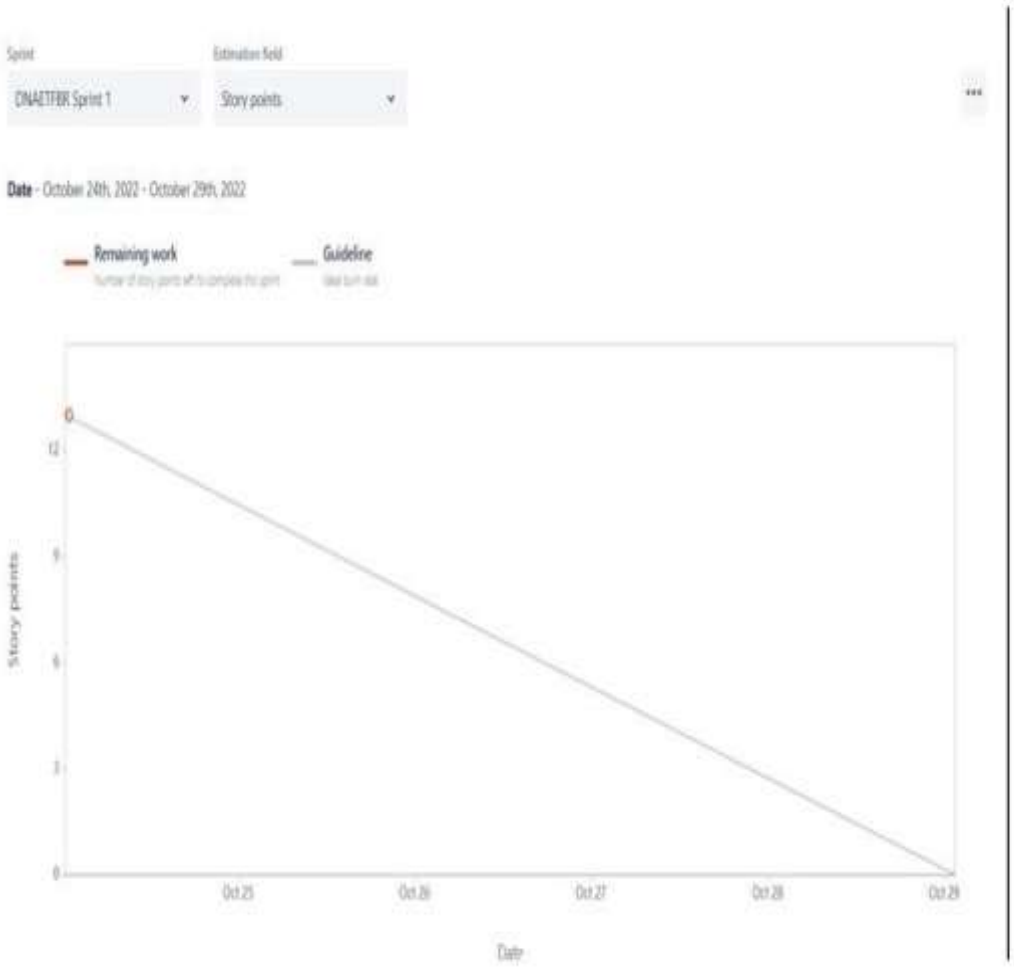
Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

Average velocity = 9/4 = 2.25

Burndown Chart:

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.



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

7.1 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 imagefile to the model via a webpage in order to get knowledge about the species that they are looking for.

7.2 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.

8. TESTING

8.1 User Acceptance Testing

Introduction:

Effectively documenting incidents during the testing process is the key to improving software or processes before a system is released. Sometimes, the testers themselves document issues they encounter; but more often, a UAT coordinator verifies, consolidates, and classifies reported issues before assigning them to the appropriate groupto address. Then, that IT coordinator again validates and prioritizes the technical issues before handing them off to an IT developer to investigate further and resolve. During the course of UAT, it is inevitable that issues will be discovered. It is shocking howoften documented issues contain insufficient data to facilitate a quick and thorough investigation.

Deliverables of UAT:

Every interviewer very quickly stated that UAT is to assure quality. Project managers also stated that it can double as a training exercise for business users as well as ensuring that the requirements set match the functionality that is desired from the system. People managers expressed thatone of the most important deliverables is the decision to go forward with the update or new system; the ”green” or ”red” light. Individual contributors expressed that UAT and inclusion goes hand in hand. That the testers feel included in the development and actually have a say in what works and what doesn’t. Individual contributors stated that they felt that UAT has been done enough when the tests they are running are all success full but that it is a gut-feeling or intuition that says when they are content with the testing. They alsostated many perks of UAT such as: learning the new system, cooperation between departments, learning something new, feeling valued by the company and inclusion in decision making. Project Managers stated that the organization at large sometimes acted as though it had forgotten the purposeof UAT - to assure quality and usability of a release.

Data Mining:

This section representsthe actions pillar of the research. Here resultsbased on empirical insights from the system log files are presented. Results from the qualitative review of the testers use of test management tools are also presented in this section.

Time spent on Testing:

Because access was granted to the SUTs application logs it waspossible to track exactly how much time users spent on testing functionality inthe system. In the blow table 4.3.1 average times are detailed some of the users from the SUT

Test Quality:

From the production logs of the SUT a Markov-chain with 68 states(one for each application feature that was left after filtering out non-relevantstates) was created. Due to the fact that the SUT was a regular release of an existing system, and not a newly adopted software, a transition matrix could be made on a per-tester level for both the production system logs, as well as the test system logs. Variability due to changes in logging were taken into account by qualitatively examining the log files. As transition matrices for bothTEST and PROD had been computed, a similarity score could be computed to directly and in bulk estimate the quality of the testing.

Execution Flow

- C program (source code) is sent to preprocessor first.
- Expanded source code is sent to compiler which compiles the code andconverts it into assembly code.
- The assembly code is sent to assembler which assembles the code andconverts it into object code.

Usually, when possible, this testing happens in a conference or a war room sortof a set up where the users, PM, QA team representatives all sit together for a day or two and work through all the acceptance test cases.

Once all the tests are run and the results are in hand, the Acceptance Decision is made.

This is also called the Go/No-Go decision. If the users aresatisfied it's a Go, or else it's a No-go. Reaching the acceptance decision is typically the end of this phase.

Conclusion:

UAT is not about the pages, fields or buttons. The underlying assumption even before this test begins is that all that basic stuff istested and is working fine. God forbid, the users find a bug as basic as that – itis a piece of very bad news for the QA team.

This testing is about the entity that is the primary element in the business.

```
class Program
{
    0 references
    static async Task Main(string[] args)
    {
        WriteLine("Please type the username for the desired user:");
        var username = ReadLine();

        var github = new GitHubClient(new ProductHeaderValue("MyAmazingApp"));

        try
        {
            var user = await github.User.Get(username);
            WriteLine($"The user {user.Name} was succesfully retrieved!");
            WriteLine($"{user.Name} has {user.PublicRepos} public repositories. Do you want to see the list? (y/n)");
            var response = ReadLine();

            if (string.Equals(
                "Y",
                response,
                StringComparison.InvariantCultureIgnoreCase))
            {
                var repos = await github.Repository.GetAllForUser(username);
                foreach (var repo in repos.OrderBy(x => x.CreatedAt))
                {
                    WriteLine($"{repo.CreatedAt:yyyy-MM-dd} | {repo.Name}");
                }
```

9. RESULTS

9.1 Performance Metrics

COLLECTION OF PERFORMANCE MEASUREMENTS

Managing application performance requires the continuous collection of data about all relevant parts of the system starting from the end user all the way through the system. This collected data is the basis for getting a holistic end-to-end and up-to-date view of the application state including the end-user experience. In this chapter, we will discuss what data to collect, and from where and how to collect the data in order to achieve this view Most application systems are implemented in a way that, in addition to the application logic executed at the provider's site (Referred to as the back-end), parts of the application are executed at client'ssite. The client site usually constitutes a system tier accessing the back end

EXTRACTION OF PERFORMANCE-RELEVANT SYSTEM INFORMATION

The previous chapter focused on the collection of performance measurements from the relevant locations of the application system. This chapter focuses on the representation of higher the application system. While time series represent summary statistics (e.g., counts, percentile, etc.) over time, execution traces provide a detailed representation of the application-internal control flow that results from individual system requests.

From this data, architectural information, including logical and physical deployments and interactions (topology), can be extracted. For all cases, we will highlight examples and use cases in the context of APM level performance-relevant information about the system and their end-users that can be extracted from this data and that is used for APM visualization and reasoning, as detailed in the next chapters. Notably, we will focus on three commonly used representations, namely time series, execution traces, and augmented information about the architecture.
When depicting the number of users accessing a system, time series usually show a periodic pattern, e.g., based on the weekdays and the hours of the day. Other interesting patterns are spikes, for instance, indicating peaks in workload or hiccups.

EXECUTION TRACES

We concluded the previous section with the statement that timeseries are not suitable for analyzing individual requests. A data structure commonly used in APM for this purpose is an

execution trace. Informally, an execution trace is a representation of the execution flow of a request through the system–ideally starting from the end user. As an example, Figure 3 depictsa schematic execution trace.

The execution trace starts with an operation called do Filter that is commonly found as an entry point in web-based applications. It can be observed that the execution of the do Filter operation includes a sequence of additional nested operation executions, until the list operation performs asequence of calls to a database.

10. ADVANTAGES & DISADVANTAGES

Advantages:

This system allows us to Identify and learn more about species automatically once an input is given. The input image is fed into a CNN which automatically analyses and produces an prediction. This project can be accessed from anywhere through the internet thus making our project portable.

Disadvantages:

The current web app is not appropriately scaled and hence won't be able to handle high traffic. Since the dataset used is not of wide variety, we will not be able to detect a wide variety of species.

11. CONCLUSION

In this project, we have deployed a website where we can upload an image of restricted set of species and the website will browse through thousands of images and will find every information it can regarding the being in the database.

12. FUTURE SCOPE

This application can be scaled widely to include a wide variety of species and also live detection systems placed in various places in areas where wildlife is widely present can be used to track and observe wildlife and help protect them.

13. APPENDIX

Source Code

digital_nature_app.py

```
from __future__ import print_function
from __future__ import division
import os

import numpy as np
import tensorflow as tf
from PIL import Image
from flask import Flask, redirect, render_template, request
from keras.applications.inception_v3 import preprocess_input
from keras.models import model_from_json, load_model
from werkzeug.utils import secure_filename
from keras.preprocessing import image

global graph
graph=tf.compat.v1.get_default_graph()
#this list is used to log the predictions in the server console
predictions = np.array(["Seneca White Deer",
                "Pangolin",
                "Lady's slipper orchid",
                "Corpse Flower",
                "Spoon Billed Sandpiper",
                "Great Indian Bustard"
                ])

#this list contains the link to the predicted species
found = np.array([
    "Seneca White Deer",
                "Pangolin",
                "Lady's slipper orchid",
                "Corpse Flower",
                "Spoon Billed Sandpiper",
                "Great Indian Bustard"
        ])
app = Flask(__name__)
model = load_model("model.h5")

@app.route('/', methods=['GET'])
def index():
    # Home Page
    return render_template("index.html")
```

```
@app.route('/predict', methods=['GET', 'POST'])
def upload():
  if request.method== 'GET':
    return("<h6 style=\"font-face:\"Courier New\";\">No GET request herd.....</h6
>")
  if request.method== 'POST':
    # fecting the uploaded image from the post request using the id 'uploadedimg'
    f = request.files['uploadedimg']
    basepath = os.path.dirname(__file__)
    #securing the  file by creating a path in local storage
    file_path = os.path.join(basepath, 'uploads', secure_filename(f.filename))
    #Saving the uploaded image locally
    f.save(file_path)
    #loading the locally saved image
    img = tf.keras.utils.load_img(file_path, target_size=(224, 224))
    #converting the loaded image to image array
    x = tf.keras.utils.img_to_array(img)
    x = preprocess_input(x)
    #converting the preprocessed image to numpy array
    inp = np.array([x])
    with graph.as_default():
      #loading the saved model from training
      json_file = open('DigitalNaturalist.json')
      loaded_model_json = json_file.read()
      json_file.close()
      loaded_model = model_from_json(loaded_model_json)
      #adding weights to the trained model
      loaded_model.load_weights("model.h5")
      #predecting the image
      preds = np.argmax(loaded_model.predict(inp),axis=1)

      #logs are printed to the console
      print("The predicted species is " , predictions[preds[0]])
  text = "The predicted species is " + found[preds[0]]
  return render_template("index.html", RESULT = text)


if __name__ == '__main__':
  #Threads enabled so multiple users can request simutaneously
  #debud is turned off, turn on during development to debug the errors
  #applications is binded to port 8000
  app.run(threaded = True,debug=True,port="8000")
```

GitHub & Project Demo Link

Github repo link:https://github.com/IBM-EPBL/IBM-Project-30929-1660192787

Project Demo Link:https://drive.google.com/file/d/1z49ux9uVKoIhk9xksDa8tkAG_niRQ7V4/view?usp=sharing