DIGITAL NATURALIST-AI ENABLED TOOL FOR BIO-DIVERSITY RESEARCHERS

A REPORT

NMS0001: PROFESSIONAL READLINES FOR INNOVATION , EMPLOYABLITIY AND ENTREPRENEURSHIP

IV YEAR / VII SEM

TEAM ID: IBMPNT2022TMID07192

Submitted by

DEEPASHRI	(130719205012)
AJAY	(130719205001)
ARUNKUMAR	(130719205007)
DEIVEEGA DHANA ROSHINI	(130719205015)

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

JERUSALEM COLLEGE OF ENGINEERING

(An Autonomous Institution, Affiliated to Anna University, Chennai)

ANNA UNIVERSITY: CHENNAI 600 025

DECEMBER 2022

BONAFIDE CERTIFICATE

Certified that this Report titled "**DIGITAL NATURALIST-AI ENABLED TOOL FOR BIO-DIVERSITY RESEARCHERS**" is the bonafide work of **DEEPASHRI(130719205012),AJAY(130719205001),ARUNKUMAR(130719205007),DEIVEEGA DHANA ROSHINI (130719205015)** who carried out the work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

Dr. K.SUNDARAMOORTHY
Professor and Head
Dept. of InformationTechnology
Jerusalem College of Engineering
Pallikaranai,Chennai 600100.

Mr. D.SUDHAGAR
Associate Professor
Dept. of Information Technology
Jerusalem College of Engineering
Pallikaranai, Chennai 600100.

Submitted to the project viva-voce Examination held on ————————

Internal Examiner **External Examiner**

ABSTRACT

This project focuses on helping researchers, naturalists, and many more people who are involved in exploring nature. It gives the species/botanical names, medicinal values, extinct/endangered species, and information about the flora and fauna to help the people who seek it. This is achieved by deep learning concepts. This provides information for both the flora and fauna. Being able to identify the flora and fauna around us often leads to an interest in protecting wild spaces. This project act as a user-friendly guide to Researchers, Naturalist and Tourist by predicting the image and provides useful information of flora and fauna precisely.

TABLE OF CONTENTS

1.INTRODUCTION

1.1 Project Overview

The ever-growing number of digital sensors in the environment has led to an increase in the amount of digital data being generated. This includes data from satellites, weather stations, data from “internet of things” devices, and data collected by members of the public via smartphone applications, to name but a few. These new sources of data have contributed to the era of “Big Data” characterized by large volumes of data, of numerous types and quality, being generated at an increasing speed. This presents challenges and opportunities across a number of domains, including water management, camera trapping, and acoustic analysis. Automated identification of plants and animals have improved considerably in the last few years. In total, nine deep learning systems implemented by three different research teams were evaluated with regard to nine expert botanists of the French flora. Therefore, we created asmall set of plant observations that were identified in the field and revised by experts in order to have a near-perfect golden standard. The main outcome of this work is that the performance of state-of-the-art deep learning models is now close to the most advanced human expertise. This shows that automated plant and animal identification systems are now mature enough for several routine tasks, and can offer very promising tools for autonomous ecological surveillance systems. Artificial intelligence (AI) 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 analyse hundreds ofthousands of visual data every day. Deep learning models (some of the most advanced AI algorithms) are developed with training datasets that allow them to capture discriminant visualpatterns. Their performances are then strongly correlated to the quality and completeness of the datasets on which they are trained. Unbalanced, biased, or otherwise poor-quality training datasets will lead to underperforming algorithms in real conditions. During the learning phases,particular attention must be given to any relevant limitations of the training data, and the gap between these and the test data on which the developed algorithms will be evaluated.

1.2 Purpose :

To better understand the complexities of natural ecosystems and better manage and protect them, it would be helpful to have detailed, large-scale knowledge about the number, location, and behaviours of animals in natural ecosystems. Having accurate, detailed, and up-to- date information about the location and behaviour of animals in the wild would improve our ability to study and conserve ecosystems. We investigate the ability to automatically, accurately, and inexpensively collect such data, which could help catalyze the transformation of many fields of ecology, wildlife biology, zoology, conservation biology, and animal behaviour into “big data” sciences. Motion- sensor “camera traps” enable collecting wildlife pictures inexpensively, unobtrusively, and frequently. However, extracting information from these pictures remains an expensive, time- consuming, manual task. We demonstrate that such information can be automatically extracted by deep learning, a cutting-edge type of artificial intelligence. We train deep convolutional neural networks to identify, count, and describe the behaviours of 48 species in the 3.2 million-image Snapshot Serengeti dataset. Our deep neural networks automatically identify animals with >93.8% accuracy, and we expect that number to improve rapidly in years to come.

2.LITERATURE SURVEY

2.1 Existing problem

Problem Statement 1

Customer States a Problem that, "As a Naturalist, I'm trying to study the patternsof nature, identifies a different kind of flora and fauna in nature. But at some point traditional approaches may become inefficient or even impossible given the volume, diversity, and heterogeneity of these data. Because, Artificial Intelligence(AI) techniques have profoundly transformed our ability to extract information from visual data. Which makes me feel, AI can also be used to extract information from big data in order to address various challenges faced by society natural resource".

ProblemStatement 2

Customer States another Problem that,"As a Influencer, I'm trying to do the interpretation of automatically collected observation data is a popular and fast-growing research field at Biodiversity. But, automatically the collection of data will only reach its full potential if data analysis can be automated to a certain degree. Because framework will include AI models that have been trained by expert taxonomists, thus providing a highg level of accuracy. Which makes me feel biodiversity data from natural collections openly accessible and easier to analyse".

2.2 References

1.Plant identification: Experts vs. machines in the era of deep learning: deeplearning techniques challenge floraexperts. Bonnet P., Goeau H., Hang S.T., Lasseck M., Sulc M., Malécot V., Jauzein P., Melet J.C., You C., Joly A..2018.

2.Plant Species Identification Using Computer Vision Techniques: A SystematicLiterature Review JanaWäldchen, Patrick Mäder Published 7 January 2017

3.AI Naturalists Might Hold the Key to Unlocking Biodiversity Data in Social Media Imagery Tom A August, Oliver L Pescott, Alexis Joly, Pierre Bonnet Patterns 2020

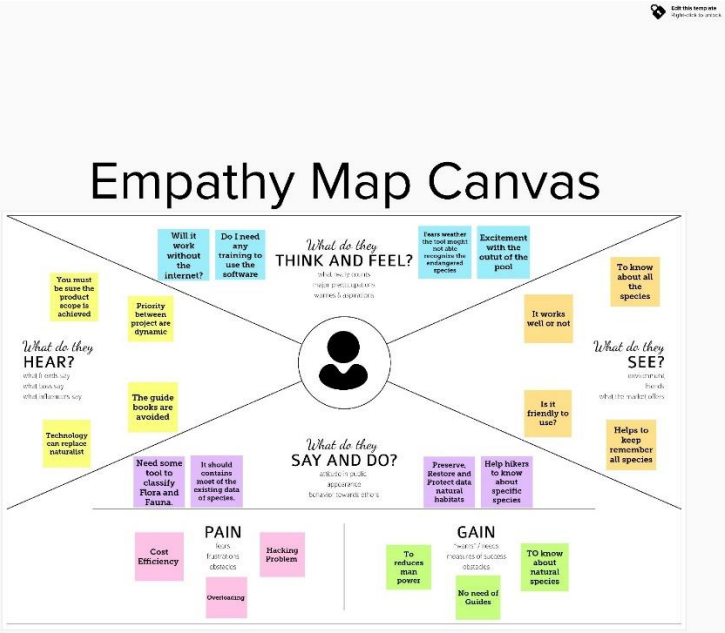
4.Automated plant identification using artificial neural networks Jonathan Y. Clark, D. Corney, H. L. Tang Computer Science 2012 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)

2.3 Problem Statement Definition

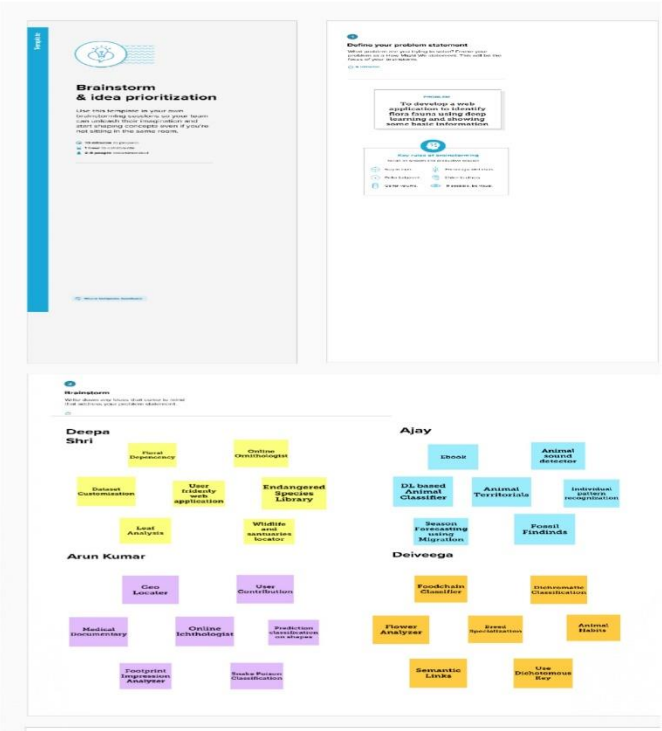
Problem to be solved is, At some point in time, traditional approaches become inefficient or even impossible given the volume, diversity, and heterogeneity of these data.
Solution for the problem , this project focuses on helping researchers, naturalists, and many more people who are involved in exploring nature. It gives the species/botanical names, medicinal values, extinct/endangered species, and information about the flora and fauna to help the people who seek it. This is achieved by deep learning concepts.

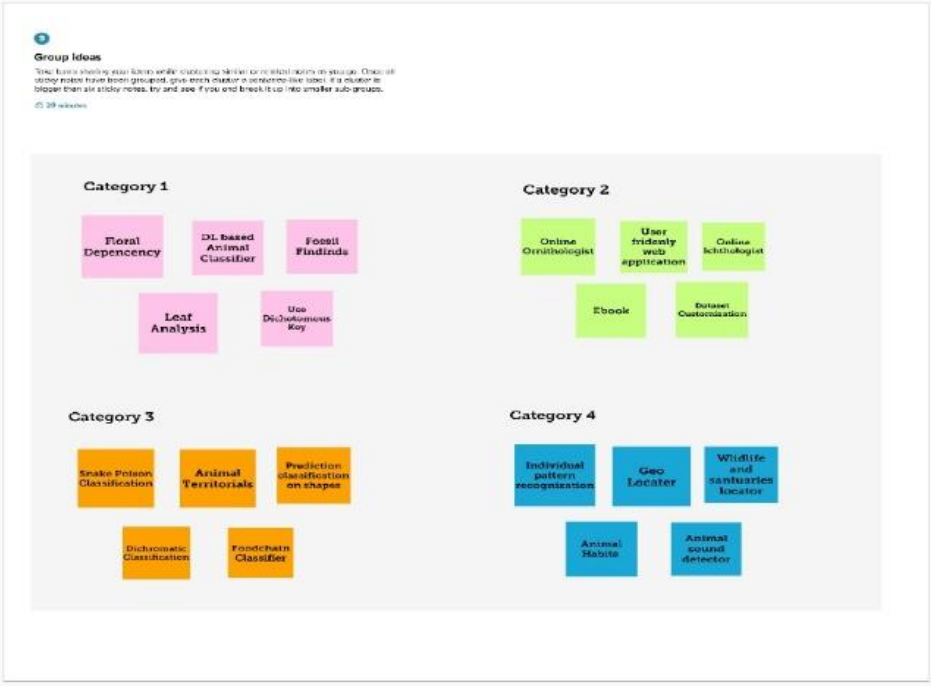
3. IDEATION & PROPOSED SOLUTION

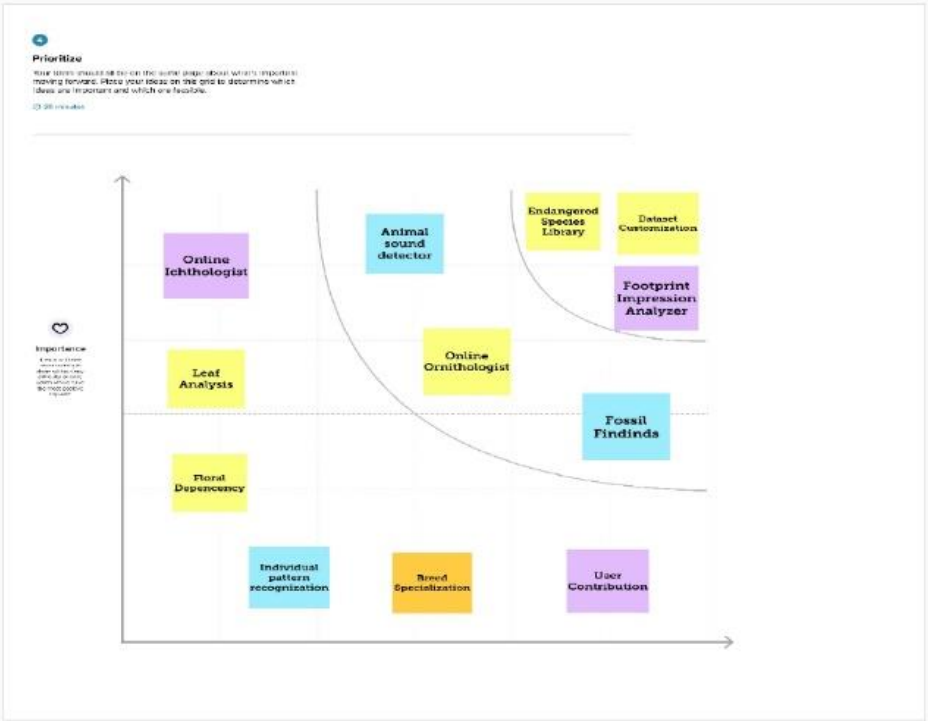
3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming



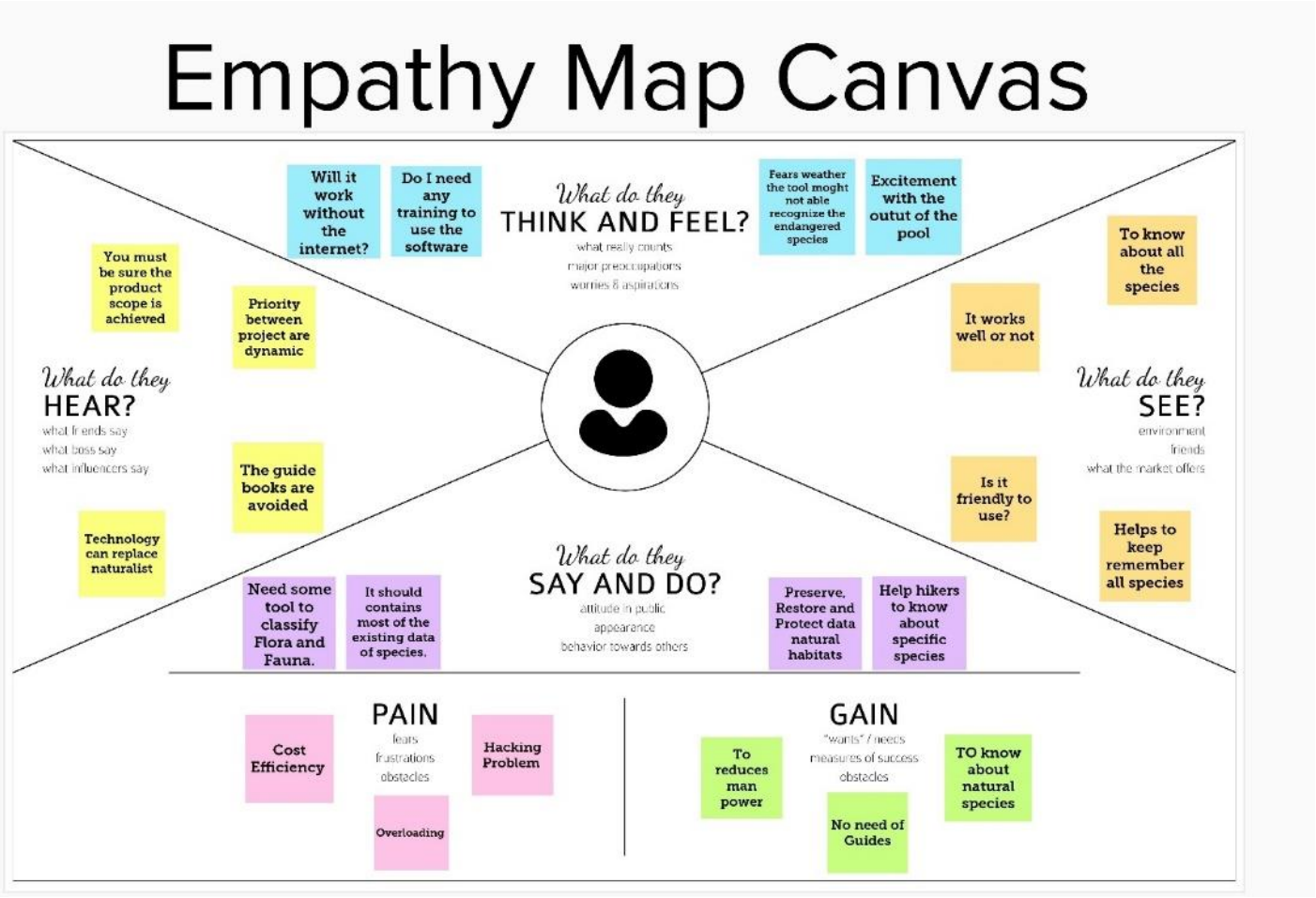

9
Group Ideas
Category 1
Floral Depencency
DL based Animal Classifier
Fossil Findinds
Leaf Analysis
Use Dichotomous Key
Category 2
Online Ornithologist
User fridenly web application
Online Ichthologist
Ebook
Dataset Customization
Category 3
Snake Poison Classification
Animal Territorials
Prediction classification on shapes
Dichromatic Classification
Fondchain Classifier
Category 4
Individual pattern recognization
Geo Locater
Wildlife and Santuaries locater
Animal Habits
Animal sound detector


4
Prioritize
Importance
Online Ichthologist
Animal sound detector
Endangered Species Library
Dataset Customization
Footprint Impression Analyzer
Online Ornithologist
Leaf Analysis
Fossil Findinds
Floral Depencency
Individual pattern recognization
Breed Specialization
User Contribution

3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	At some point in time, traditional approaches become inefficient or even impossible given the volume, diversity, and heterogeneity of these data.
2.	Idea / Solution description	This project focuses on helping researchers, naturalists, and many more people who are involved in exploring nature. It gives the species/botanical names, medicinal values, extinct/endangered species, and information about the flora and fauna tohelp the people who seek it. This is achieved by deep learning concepts.
3.	Novelty / Uniqueness	This provides information for both the flora and fauna.
4.	Social Impact / Customer Satisfaction	Being able to identify the flora andfauna around us often leads to an interest in protecting wild spaces.
5.	Business Model (Revenue Model)	It can make money through subscription-based. Partnership with many laboratories and scientists around the world.
6.	Scalability of the Solution	This application can be accessed anywhere and anytime by users with the help of the internet.

3.4 Problem Solution fit



4. REQUIREMENT ANALYSIS

4.1 Functional requirement

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	■ Registration through Gmail ■ Registration through Mobile Number
FR-2	User Confirmation	● Confirmation via Email ● Confirmation via OTP
FR-3	Authentication	By entering the OTP sent to the Gmail or Mobile.
FR-4	Subscriptions	Transactions via ● Net Banking or ● UPI or ● Credit card.
FR-5	Administrative functions	■ Maintaining description of flora and fauna. ■ Adding the species.
FR-6	User interfaces	> Easy to understand. > Sharing the experience with friends.

4.2 Non-functional Requirements:

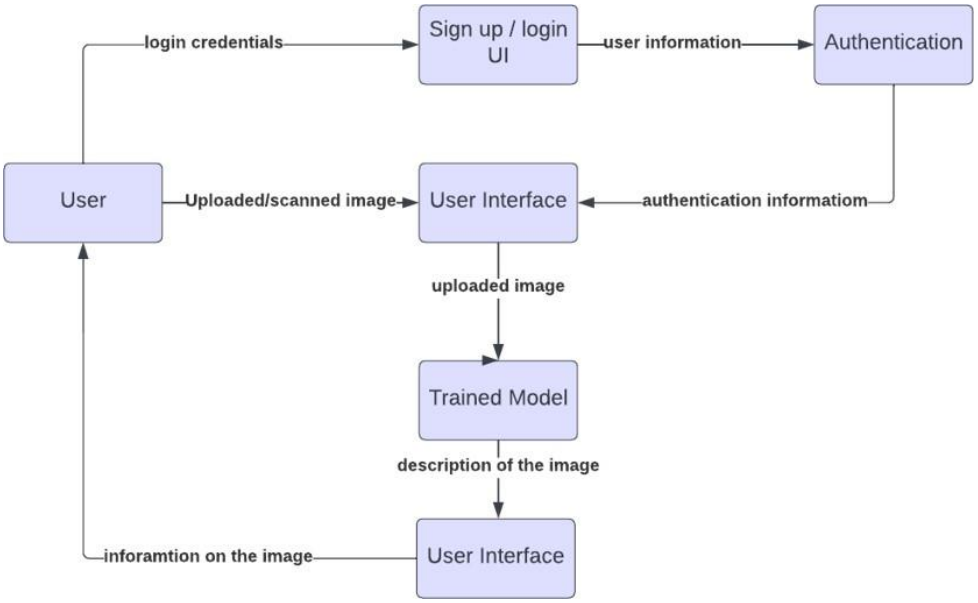
NFR No.	Non-Functional Requirement	Description
NFR-1	Usability	This project act as a user-friendly guide to Researchers, Naturalist & Tourist by predicting the image and provides useful information of flora and fauna precisely.
NFR-2	Security	● Authentication helps in maintaining the data secured. ● Subscription transactions details should be encrypted.
NFR-3	Reliability	This project provides reliability by covering the various species among different habitats.
NFR-4	Performance	■ To provide increasing in accuracy with low loss. ■ To be more efficient in prediction of flora and fauna. ■ Increased Data Augmentation
NFR-5	Availability	> Dataset is constantly updated. > Network required for cent percent prediction.
NFR-6	Scalability	It supports many users without any issues which scaled through the cloud resources.

5. PROJECT DESIGN

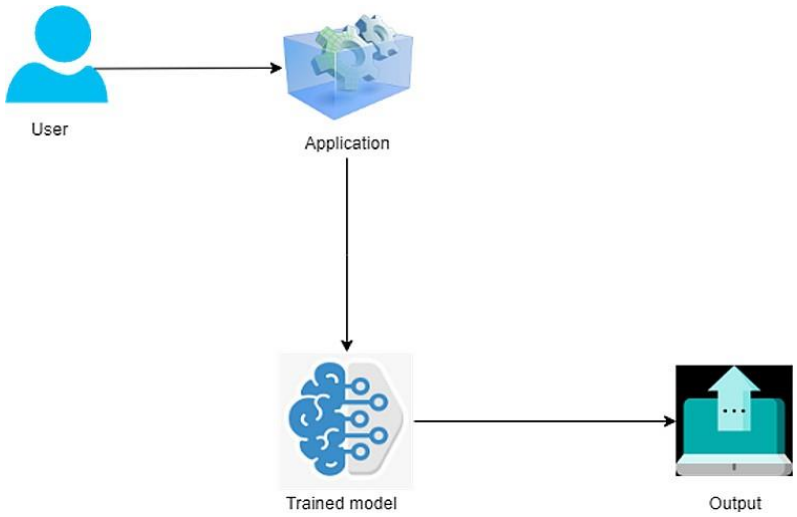
5.1 Data Flow Diagram

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

Data Flow Diagram:



Simplified Data Flow Diagram:



5.2 Solution & Technical Architecture

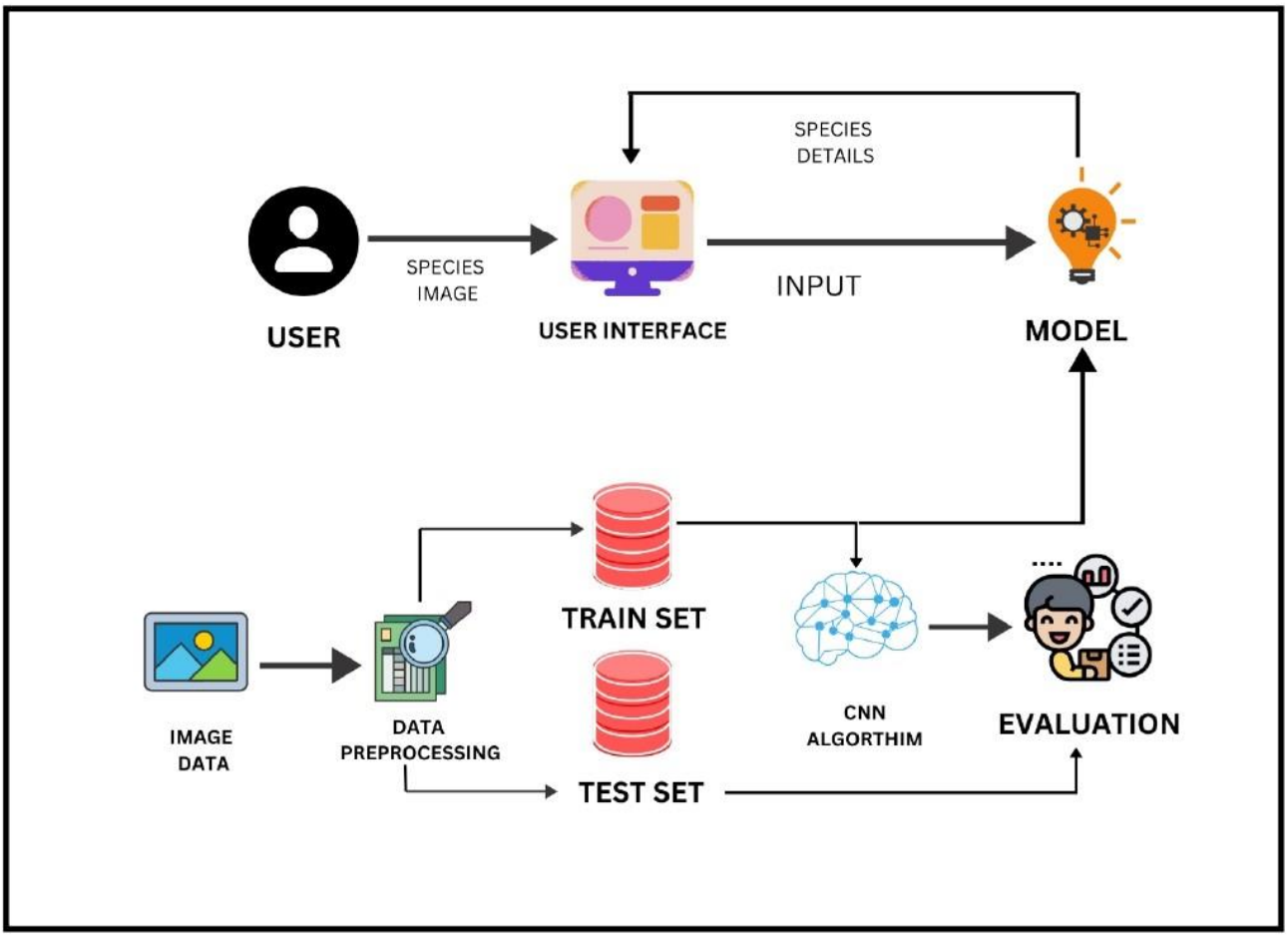


Fig. An Architecture diagram for the digital naturalist

Table-1: Components & Technologies:

S. No	Component	Description	Technology
1.	User Interface	Web UI or Website	HTML, CSS, JavaScript / React JS
2.	Application Logic-1	Model building and then training the model	Python
3.	Application Logic-2	User uploads the image for the prediction	IBM Watson STT service
4.	Application Logic-3	Getting the relevant data from the database and providing to the user	IBM Watson Assistant
5.	Database	Image of all the variety species along with detailed information of each species	MySQL / NoSQL
6.	Cloud Database	Gets the data from database and feed them to model for prediction and also used to retrieve the data required for user.	IBM Cloudant, IBM DB2
7.	File Storage	User Login credentials, Images and their data, code and API keys	IBM Block Storage
8.	External API-1	To get data from the database when user gives the image asthe input	IBM Storage API
9.	External API-2	To collect the username and password of the specific user	Secure Authentication API
10.	Machine Learning Model	To predict the both flora & fauna through the image whichis given as input and also it gives detailed information of the particular species	Image Recognition Model(Detecting the species and identifying the model)
11.	Infrastructure (Server / Cloud)	To deploy the 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 framework (Web Server GatewayInterface)
2.	Security Implementations	To Authenticate the species data in database as well as User credentials.	SHA-256, Encryptions
3.	Scalable Architecture	To scale our application in serverside by supporting clients including desktop browsers, mobile browsers etc..	IBM Auto Scaling
4.	Availability	To make application available both online and offline and also24/7 service.	IBM Cloud load balancer
5.	Performance	Designing an application which can handle wide range of requests at a time to provide accuracy in prediction as well as without any delay in time.	IBM instance

5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer or User	Sign up/Login	USN-1	As a user, I can log in/signup for the application.	I can access the application through this process.	High	Sprint-1
	Upload or scan the image	USN-2	As a user, I can upload or scan the image about which the information is needed.	I can upload the image.	High	Sprint-2
	Get information on the image	USN-3	As a user, I can get information about theimage.	I can get information aboutthe image.	High	Sprint-2

6.PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Modelling Phase	USN-1	Data Collecting and digitalizing for analysing	3	Medium	AJAY R
Sprint-1		USN-2	Adding more data to avoid overfitting	2	Medium	ARUN KUMAR B
Sprint-1		USN-3	Building a CNN model using the collected data	5	High	DEIVEEGA
Sprint-1		USN-4	Evaluating the model to check the accuracy and precision	3	High	DEEPA SHRI
Sprint-2	Development Phase	USN-5	Home page Creation – Shows the features of our application	1	Low	AJAY R
Sprint-2		USN-6	Setting up facilities for user to feed the image	2	Medium	ARUN KUMAR B
Sprint-2		USN-7	Prediction page creation – shows prediction for the user given image	4	Medium	DEIVEEGA
Sprint-2		USN-8	Model loading – API creation using flask	5	High	DEEPA SHRI
Sprint-3	Deployment Phase	USN-9	Integrating UI & backend – Connecting the front end and backend using API calls	3	Medium	AJAY R
Sprint-3		USN-10	Cloud deployment – Deployment of application using IBM Cloud	5	High	ARUN KUMAR B
Sprint-4	Testing Phase	USN-11	Functional testing – Checking the scalability and robustness of the application	5	High	DEIVEEGA, DEEPA SHRI

Velocity:

Average Velocity = Sprint duration/velocity

= 15/6

= 2.5

6.2 Sprint Delivery Schedule

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	13	6 Days	24 Oct 2022	29 Oct 2022	13	29 Oct 2022
Sprint-2	12	6 Days	31 Oct 2022	05 Nov 2022	12	05 Nov 2022
Sprint-3	8	6 Days	07 Nov 2022	12 Nov 2022	8	12 Nov 2022
Sprint-4	10	6 Days	14 Nov 2022	19 Nov 2022	10	19 Nov 2022

Velocity:

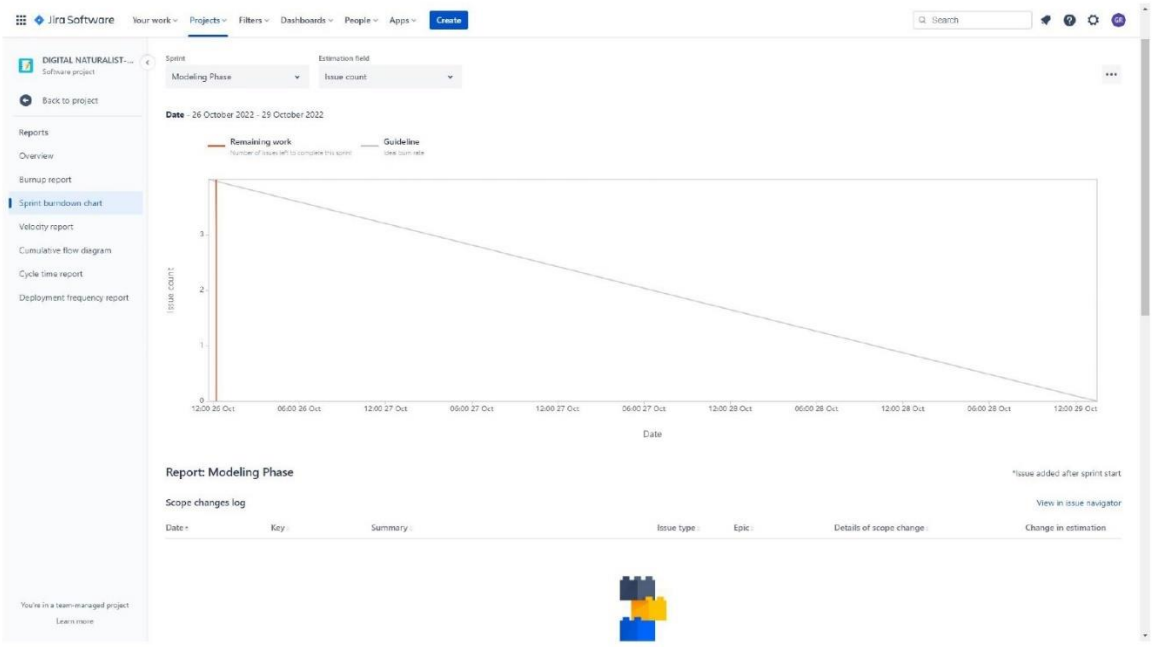
Average Velocity = 61/24 = 2.51

6.3 Reports from JIRA

Burndown Chart:

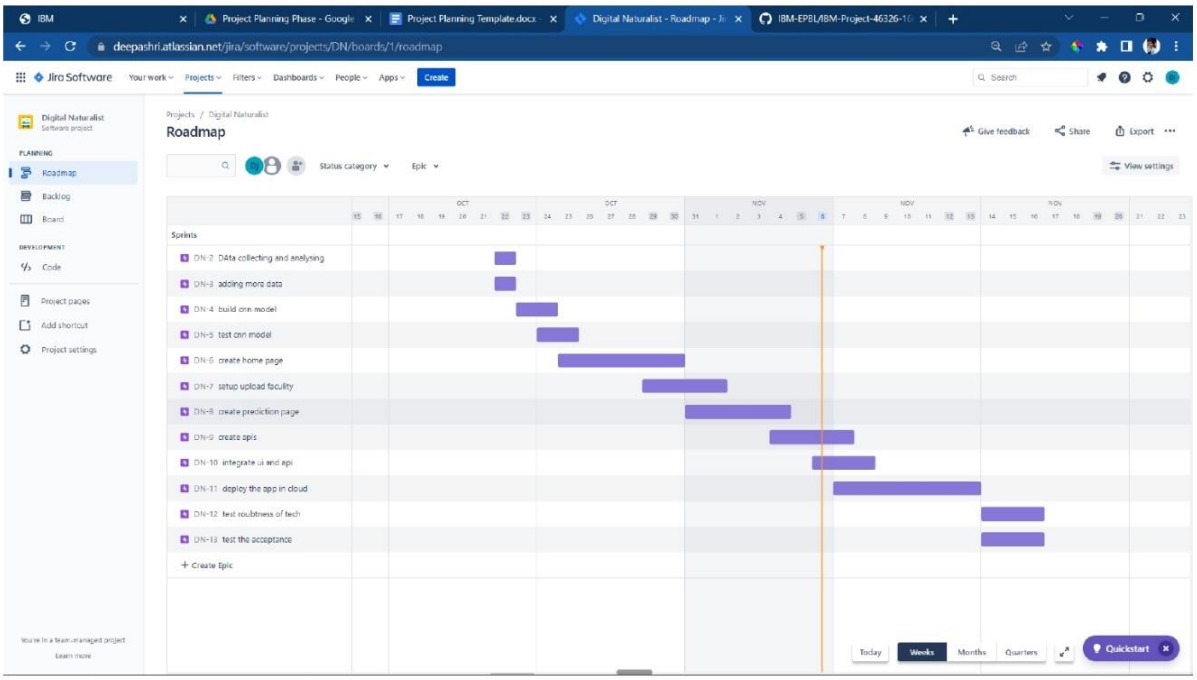
A Burndown Chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.

Burndown Chart:



Roadmap:

Roadmap:



7.CODING & SOLUTIONING

7.1 Code for web page creation:

home.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <link rel="spreadsheet" type="text/css" href="styles.css">
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet"
href="https:/     maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

<link href='https:/  fonts.googleapis.com/css?family=Josefin Sans' rel='stylesheet'>
<link href='https:/  fonts.googleapis.com/css?family=Merriweather' rel='stylesheet'>
          <script src="https:/  cdn.bootcss.com/popper.js/1.12.9/umd/popper.min.js"></script>
    <script src="https:/  cdn.bootcss.com/jquery/3.3.1/jquery.min.js"></script>
    <script src="https:/  cdn.bootcss.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>

  <title>Digital Naturalist</title>
  <style>
    @import
url('https:/          fonts.googleapis.com/css2?family=Josefin+Sans:ital,wght@1,300&family=Montserr
at&family=Raleway:wght@100&family=Roboto:wght@300&family=Sacramento&family=Sourc
e+Sans+3:wght@300;400&display=swap');
    *{
      padding:0;
      margin: 0;
      box-sizing: border-box;
      list-style: none;
      text-decoration: none;

    }
    body{
      font-family: 'Montserrat', sans-serif;
      background-image: linear-gradient( 89.8deg, rgb(13, 95, 33) 4.7%, rgba(30, 29, 29, 1) 120.
```

```
cursor: pointer;
            font-weight:bold;
  }

  .upload-label:hover{
     background:
     #3A3A3A;color:
     white;
            font-weight:bold;
  }

  .loader {
     border: 8px solid #f3f3f3; /* Light grey
     */border-top: 8px solid #161616;
     border-radius: 50%;
     width: 50px;
     height: 50px;
     animation: spin 1s linear infinite;
  }

  @keyframes spin {
     0% { transform: rotate(0deg); }
     100% { transform: rotate(360deg); }
  }
   </style>
 </head>
 <body>
   <nav>
     <img class="logo" src="../static/images/greentree-removebg-preview.png" >
     <label class="lab">Digital Naturalist</label>
     <ul>
       <li><a  href="{{url_for('home_func')}}">Home</a></li>
       <li><a href="{{url_for('aboutus_func')}}">About us</a></li>
     </ul>
   </nav>
   <div class="sec">
     <div class="content">
       <p class="topic">ARE YOU A NATURALIST ?</p><br>
```

```
    <p class="desc">
      Wanna know more about the flora and fauna around you?<br>

      Drop their picture now! </p>
   </div>
   <br>

      <div class="center">
        <a href="#section2">
          <button type="button" class="myButton" >Click here!</button>

        </a>
   </div>
  </div>


  <div class="predicting" id="section2" >
   <section id="main">
   <div style="text-align:center;width:100%;">
   <p><h3 style="color:white;" >Click on choose and upload the image<br><br></h3></p>

     </div>
     </section>
</div>
  <div style="margin-top:0%;padding-top:0%;">
     <div>
       <h4 style="font-size:19px;text-align:center; color:white; padding-bottom:30px ; ">Upload
your image</h4>
   <center><form style="border-radius: 4px;"action = "http:/ localhost:5000/" id="upload-file"
method="post" enctype="multipart/form-data">
     <label style="text-align: center;"for="imageUpload" class="upload-label">
       Choose
     </label>
     <input type="file" name="image" id="imageUpload" accept=".png, .jpg, .jpeg">
   </form></center>


   <div class="image-section" style="display: none; text-align:center;padding-left: 40%;
width:500px; height:300px;">
     <div class="img-preview">
       <div  id="imagePreview">
```

```
      </div>
    </div>
    </div>
    <br>
   <div class="image-section" style="display: none;">
    <div style="padding-left: 45%;">
      <button type="button" class="btn btn-lg upload-label" id="btn-predict">Predict!</button>
    </div>
   </div>

   <div class="loader" style="display:none;"></div>
   <div style="width:70%;text-align:justify;margin-left:20%;">
   <h4 style="color:white">
    <span id="result"> </span>
   </h4></div>

</div>

</div></div>
<script>
window.onscroll = function() {myFunction()};

$(document).ready(function () {
 / Init
 $('.image-section').hide();
 $('.loader').hide();
 $('#result').hide();

 / Upload Preview
 function readURL(input)
 {
    if (input.files && input.files[0])
       { var reader = new
       FileReader();reader.onload =
       function (e) {
          $('#imagePreview').css('background-image', 'url(' + e.target.result + ')');
          $('#imagePreview').hide();
          $('#imagePreview').fadeIn(650);
       }
       reader.readAsDataURL(input.files[0]);
    }
 }
 $("#imageUpload").change(function () {
```

```
    $('.image-section').show();
    $('#btn-predict').show();
    $('#result').text('');
    $('#result').hide();
});
    readURL(this);
;
```

```
    / Predict
    $('#btn-predict').click(function () {
        var form_data = new FormData($('#upload-file')[0]);

        /  Show loading animation
        $(this).hide();
        $('.loader').show();

        /  Make prediction by calling api /predict
        $.ajax({
            type: 'POST',
            url: '/predict',
            data: form_data,
            contentType: false,
            cache: false,
            processData: false,
            async: true,
            success: function (data) {
                /  Get and display the result
                $('.loader').hide();
                $('#result').fadeIn(600);
                $('#result').text('Prediction: '+data);
                console.log('Success!');
            },
        });
    });

  });
  </script>
</body>
  </html>
```

8.TESTING

8.1 TEST CASES

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)	BUG ID	Executed By
Webpage_TC_001	UI	Home Page	Verify whether user can view the homepage	1. Latest web browser 2. Proper Internet Connection	1. Enter the url of the website and click go 2. Verify the webpage is loading or not	no test data required	The webpage should be visible to the user	The webpage is visible	Pass	The test case passed without any issues	Y	1	Vijayalakshmi K
Webpage_TC_002	UI	Home Page	Verify whether user is able to upload the image	1. Latest web browser 2. Proper Internet Connection 3.Sample images for testing	1. Enter the url of the website and click go 2. After tha page loaded , Click on the choose button to uploadthe image	Image	The image should be uploaded successfully	The image was uploaded successfully	Pass	The test case passed without any issues	Y	2	Swetha S
Webpage_TC_003	UI	Home Page	Verify the webpage accepts proper inputs from user and displays it	1. Latest web browser 2. Proper Internet Connection 3.Sample images for testing	1. Enter the url of the website and click go 2. After tha page loaded , Click on the choose button to upload the image	Sample images for testing	The webpage should accept the image and display it to the user	The webpage accepts the image and displays it to the user.	Pass	The test case passed without any issues	Y	3	Rakshana B
Webpage_TC_004	UI	Home Page	Verify whether web components	1. Latest web browser 2. Proper Internet	1. Enter the url of the website and click go 2. Verify the webpage is loading and working properly upload and reset	Sample image for	The webpage should be stable during the upload and prediction	The webpage is responding stabley	Pass	The test case passed without any	Y	4	Rakshana B
Webpage_TC_005	UI	About us Page	Verify whether user can view the about us page	1. Latest web browser 2. Proper Internet Connection	1. Enter the url of the website and click go 2. Verify the webpage is loading or not	no test data required	The webpage should be visible to the user	The webpage is visible	Pass	The testcase passed without any issues	Y	5	Swetha K
Flask_TC_001	Functional	Flask app	Verify the flask app whether it uses the trained model	1. Latest web browser 2. Proper Internet Connection	1. Enter the url of the website and click go 2. Verify the webpage is accepting inputs and predecting according to the category of the animal	Sample images for testing	The webapp should predict the image properly	The webapp predicts the image accurately	Pass	The test case passed without any issues, but it requires more training set to predict the image accuretly	Y	6	Vijayalakshmi K
Flask_TC_002	Functional	Flask app	Verify whether the uploaded image is saved in the specified folder	1. Latest web browser 2. Proper Internet Connection 3. Storage as a folder for storing the uploaded image	1. Enter the url of the website and click go 2. After page loading try to upload the image and wait	Sample images for testing	The website should accept the image data and save it locally in a folder	The app stored the image successfully	Pass	The testcase passed without any issues, But it will be an issue in future when the storage	Y	7	Swetha S

		Flask	Verify whether the app displays the	1. Latest web browser 2. Proper Internet Connection testing	1. Enter the url of the website and click go 2. Verify the webpage inputs and predecting according to their category.	Sample images for	The web app should be able to display the prediction	The app displays the prediction		The testcase passed without			
Flask _TC_004	Functional	Flask app	Verify whether the app redirects the user to the about us page or not?	1. Latest web browser 2. Proper Internet Connection 3.Sample images for testing	1. Enter the url of the website and click go. 2. Verify the page is redirecting to about us page.	no test data required	The web app should redirect to the user to about us page	The app redirected successfully	Pass	The testcase passed without any issues	Y	9	Swetha S

8.2 USER ACCEPTANCE TESTING

Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the Digital Naturalist project at the time of the release to User Acceptance Testing (UAT).

Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

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

Test Case Analysis

This report shows the number of test cases that have passed, failed, anduntested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	5	0	1	4
Client Application	30	0	0	30
Security	1	0	0	1
Outsource Shipping	4	0	0	4
Exception Reporting	30	0	0	30
Final Report Output	3	0	0	3
Version Control	2	0	0	2

9. RESULTS

9.1 PERFORMANCE METRICS

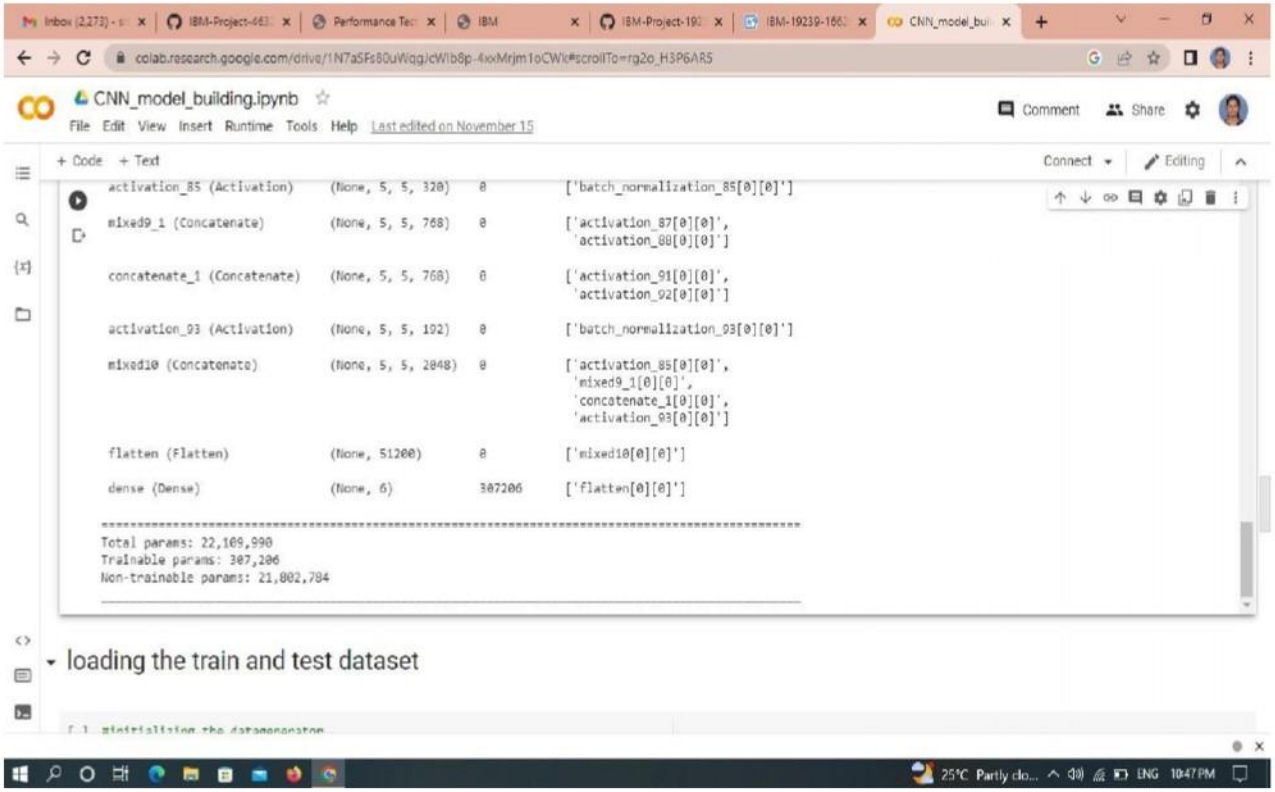
Model Performance Testing:

Project team shall fill the following information in model performance testing template.

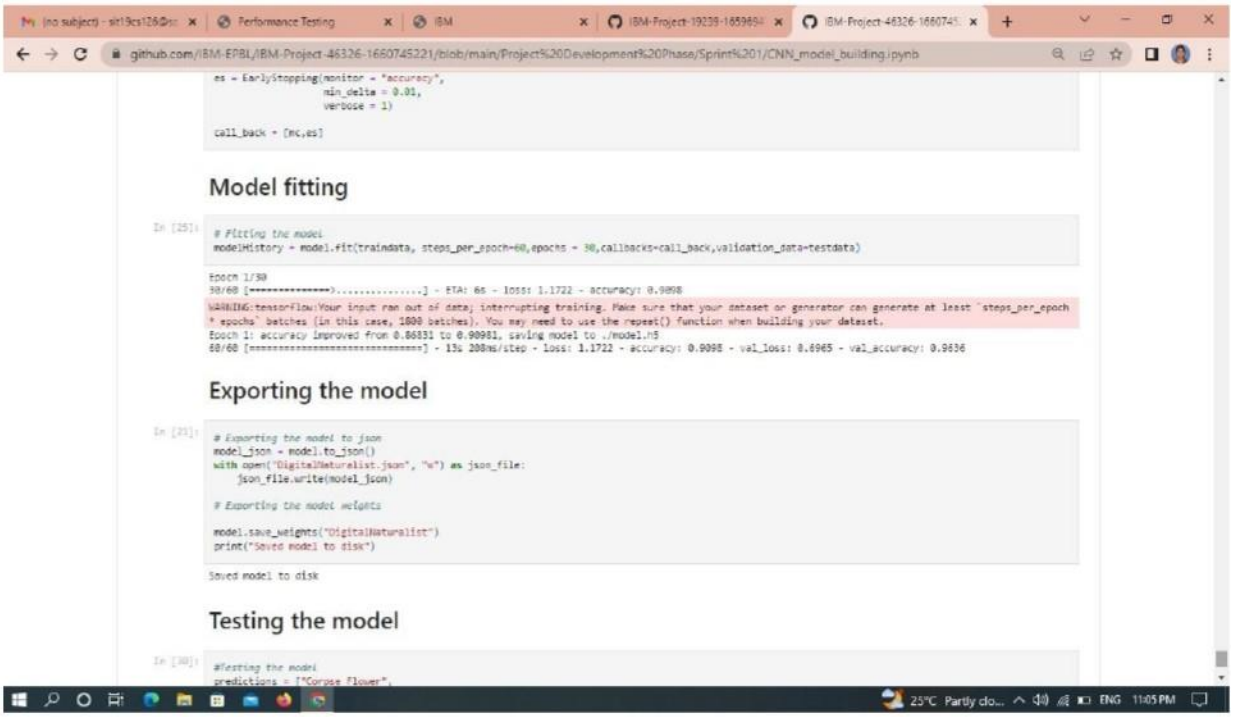
S.No.	Parameter	Values	Screenshot
1.	Model Summary	Total params: 22,109,990 Trainable params: 307,206 Non-trainable params: 21,802,784	Screenshot 1
2.	Accuracy	Training Accuracy - 90.9% Validation Accuracy - 96.3%	Screenshot 2

Screenshots - Please refer to the next page:

SCREENSHOT 1:



SCREENSHOT 2:



10.ADVANTAGES AND DISADVANTAGES

ADVANTAGES:

An advantage of naturalistic observation is that it allows the investigators to directly observe the subject in a natural setting. The method gives scientists a first-hand look at social behavior and can help them notice things that they might never have encountered in a lab setting.The observations can also serve as inspiration for further investigations. The information gleaned from naturalistic observation can lead to insights that can be used to help peopleovercome problems and lead to healthier, happier lives.

i.)Allows researchers to study behaviors or situations that cannot be manipulated in a lab due to ethical concerns. For example, it would be unethical to study the effects of imprisonment by actually confining subjects. But researchers can gather information by using naturalistic observation in actual prison settings.

ii.)Can support the external validity of research. Researchers might believe that the findings of a lab study can be generalized to a larger population, but that does not meanthey would actually observe those findings in a natural setting.

DISADVANTAGES

Naturalistic observation can be useful in many cases, but the method also has some downsides. Some of these include:

Inability to draw cause-and-effect conclusions: The biggest disadvantage of naturalistic observation is that determining the exact cause of a subject's behavior can be difficult.

Lack of control: Another downside is that the experimenter cannot control for outsidevariables.

Lack of validity: While the goal of naturalistic observation is to get a better idea of how it occurs in the real world, experimental effects can still influence how people respond. The Hawthorne effect and other demand characteristics can play a role in people altering their behavior simply because they know they are being observed. It is also important to note that naturalistic observation is a type of correlational research(others include surveys and archival research). A correlational study is a non-experimental approach that seeks to find statistical relationships between variables. Naturalistic observation is one method that can be used to collect data for correlational studies.
While such methods can look at the direction or strength of a relationship between two variables, they cannot determine if one causes the other. As the saying goes, correlation does not imply causation.

11.CONCLUSION

Naturalistic observation can play an important role in the research process. It offers anumber of advantages, including often being more affordable and less intrusive than other types of research.

In some cases, researchers may utilize naturalistic observation as a way to learn moreabout something that is happening in a certain population. Using this information, they can then formulate a hypothesis that can be tested further.

In conclusion, AI applications in Biodiversity help conserve various species of mammals, insects, birds, etc. Moreover, it helps experts understand the behaviors ofmultiple species in their ecology. Hence, it helps replicate certain necessary elementsfor their survival from their environments.

12. FUTURE SCOPE

One very time-consuming task in biodiversity research is data collection. Traditionally, a scientist might have spent hours waiting for one single observation, chasing away most timid animals and therefore distorting the data. Machine observations free researchers from tedious tasks and even make rare observations possible at all. Researchers have been using camera traps in order to monitor bigger animals like lions or antelopes. But after collecting huge amounts of images, the problem remains that the amount of information exceeds the capacity of human interpreters, with only a small percentage of the collected material being relevant at all. That's why automating the collection of data will only reach its full potential if data analysis can be automated.

APPENDIX

SOURCE CODE

Code for building flask app:

app.py

```
from_future_import division, print_functionimport os
import numpy as np
from keras.models import load_modelimport tensorflow as tf
from tensorflow.keras.preprocessing import
imagefrom flask import Flask, request,
render_template from werkzeug.utils import
secure_filename
from keras.models import
model_from_jsonglobal graph
graph=tf.compat.v1.get_default_graph()
# Define a flask
app app = Flask(_
name
```

```
json_file.close()

loaded_model =
model_from_json(loaded_model_json)
loaded_model.load_weights("final_model.h5")
print('Model loaded. Check http://127.0.0.1:5000/')
#Configure Home page.
@app.route('/
')def index():
 # Main page


   return render_template('home.html')
```

```
@app.route('/home')
def home():

 # Main page

   return render_template('home.html')
@app.route("/home",
methods=['GET','POST'])def home_func():
 # Main page

   if request.method=='POST':
      return
      redirect(url_for('home'))
   return render_template('home.html')
@app.route("/aboutus",
methods=['GET','POST'])def aboutus_func():
   # Main page

   if request.method=='POST':

      return redirect(url_for('aboutus'))
   return
   render_template('aboutus.html')
#Pre-process the frame and run
@app.route('/predict', methods=['GET',
'POST'])def upload():
   if request.method == 'POST':

      # Get the file from post
      requestf =
      request.files['image']



      # Save the file to ./uploads
```

```
    basepath = os.path.dirname(_file_
    )file_path = os.path.join(
      basepath, 'uploads',
    secure_filename(f.filename))f.save(file_path)
    img = image.load_img(file_path, target_size=(224,


      preds = np.argmax(model.predict(x),axis=1)
```

found = [" The great Indian bustard is a bustard found on the Indian subcontinent. A large birdwith a horizontal body and long bare legs, giving it an ostrich like appearance, this bird is among the heaviest of the flying birds. It belongs to Otididae family and is listed among critically endangered species.",

" The spoon-billed sandpiper is a small wader which breeds in northeastern Russia andwinters in Southeast Asia. It belongs to Scolopacidae family and is listed among critically endangered species.",

" Amorphophallus Titanum is endemic to sumantra. Due to its odor, like that of a rottingcorpse, the titan arum is characterized as a Carrion Flower or Corpse Flower. It belongs to Araceaefamily.",

" Lady's slipper, (subfamily Cypripedioideae), also called lady slipper or slipper orchid, subfamily of five genera of orchids (family Orchidaceae), in which the lip of the flower is slipper-shaped.",

" Pangolins, sometimes known as scaly anteaters, are of the order Pholidota. Often thought of as a reptile, but pangolins are actually mammals. They are the most trafficked mammals.",

" The white deer found at Seneca Army Depot are a natural variation of the white-tailed deer (Odocoileus virginianus), which usually have brown coloring. The Seneca White Deer are leucistic, meaning they lack all pigmentation in the hair, but have the normal brown-colored eyes."]

```
    text =
    found[preds[0]]
    return text
if_name_== '_main_':
```

```
app.run(threaded =
```

Code for augmenting the data:

Aug data.ipynb

```
#import  libraries


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

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):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


    # 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
  """
  Arguments: file_dir: A string representing the directory where images that we want to augment
are found.
      n_generated_samples: A string representing the number of generated samples using the
given image.
      save_to_dir: A string representing the directory in which the generated images will be
saved."""
```

Code for training the data:

model_train.ipynb

```
from google.colab import drive
drive.mount("/content/drive")

!unzip "/content/drive/MyDrive/Colab Notebooks/Digital Naturalist Dataset.zip"

#import libraries
from keras.preprocessing.image import ImageDataGenerator
import cv2
from os import listdir
import time

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_director
```

```
Naturalist/augmented data"

 #For Birds
 # augment data for the examples with label equal to GIB in Birds
 augment_data(file_dir=r'/content/drive/MyDrive/Colab Notebooks/Digital Naturalist/Digital
 Naturalist Dataset/Bird/Great Indian Bustard Bird', n_generated_samples=8,
 save_to_dir=augmented_data_path+'/Bird/GIB_AUG')
 # augment data for the examples with label equal to GIB in Birds
 augment_data(file_dir=r'/content/drive/MyDrive/Colab Notebooks/Digital Naturalist/Digital
 Naturalist Dataset/Bird/Spoon Billed Sandpiper Bird', n_generated_samples=8,
 save_to_dir=augmented_data_path+'/Bird/SPS_AUG')

 #For MAMMALS
 # augment data for the examples with label equal to GIB in Flower
 augment_data(file_dir=r'/content/drive/MyDrive/Colab Notebooks/Digital Naturalist/Digital
 Naturalist Dataset/Flower/Corpse Flower', n_generated_samples=8,
 save_to_dir=augmented_data_path+'/Flower/Corpse_AUG')
 # augment data for the examples with label equal to GIB in Flower
 augment_data(file_dir=r'/content/drive/MyDrive/Colab Notebooks/Digital Naturalist/Digital
 Naturalist Dataset/Flower/Lady Slipper Orchid Flower', n_generated_samples=8,
 save_to_dir=augmented_data_path+'/Flower/LS_Orchid_AUG')

 #For Flowers
 # augment data for the examples with label equal to GIB in Mammals
 augment_data(file_dir=r'/content/drive/MyDrive/Colab Notebooks/Digital Naturalist/Digital
 Naturalist Dataset/Mammal/Pangolin Mammal', n_generated_samples=8,
 save_to_dir=augmented_data_path+'/Mammal/LS_Pangolin_AUG')
 # augment data for the examples with label equal to GIB in Mammals
 augment_data(file_dir=r'/content/drive/MyDrive/Colab Notebooks/Digital Naturalist/Digital
 Naturalist Dataset/Mammal/Senenca White Deer Mammal', n_generated_samples=8,
 save_to_dir=augmented_data_path+'/Mammal/SW_Deer_AUG')


 end_time = time.time()
 execution_time = (end_time - start_time)
 print(f"Elapsed time: {hms_string(execution_time)}")


 # Loading Data and Preprocessing
 #Importing the libraries
```

```
#For matrix calculations and data Managememnt
import pandas as pd
import numpy as np

#Importing libraries required for the model
import tensorflow as tf
import keras
import keras.backend as K

from keras.optimizers import SGD, Adam, Adagrad,
RMSpropfrom keras.applications import *
from keras.preprocessing import *
from keras_preprocessing.image import ImageDataGenerator, img_to_array, array_to_img,
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.models import Model
from keras.utils.np_utils import to_categorical
from sklearn.model_selection import train_test_split

#For plotting charts used for data visualizationsimport
matplotlib.pyplot as plt

#Libraries for Locating and loading data
import pathlib
from pathlib import Path
import os, gc, glob, random
from PIL import Image

#Make a list of paths to all folders where you have data
#Setting path to our dataset folder
#dirName = r'C:/Users/vijay/OneDrive/Desktop/Digital Naturalist'
dirName="/content/drive/MyDrive/Colab Notebooks/Digital Naturalist/Digital Naturalist
Dataset"

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 classsubfolders
```

#Loading Images into machine understandable Data

```
#X data will includes the data generated for each image
#Y data will include a id no:, for every different boat type in out boats folder
#a different number is being assigned. That will be tha label we're classifying.
X_data = []
Y_data = []


id_no=0
found =
[]
#itering in all folders under Boats folder
for paths in subfolders:
   #setting folder path for each boat type
   files = glob.glob (paths + "/*.jpg")
   found.append((paths.split('\\')[-1],paths.split('\\')[-1]))

   #itering all files under the folder one by one
   for myFile in files:
      img = Image.open(myFile)
      #img.thumbnail((width, height), Image.ANTIALIAS) # resizes image in-place keeps ratio
      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


from keras.preprocessing.image import ImageDataGenerator
#Define arguments for ImageDataGenerator Class
train_datagen = ImageDataGenerator(rescale = 1./255,shear_range = 0.2,zoom_range =
0.2,horizontal_flip = True)
test_datagen = ImageDataGenerator(rescale = 1./255)

#Applying ImageDataGenerator functionality to trainset and testset

path="/content/drive/MyDrive/Colab Notebooks/Digital Naturalist/Digital Naturalist Dataset"

x_train=train_datagen.flow_from_directory(path,target_size = (64,64),batch_size =
32,class_mode = "categorical")
x_test=test_datagen.flow_from_directory(path,target_size = (64,64),batch_size = 32,class_mode
= "categorical")

#Data splitting into Train And Test

#to see our data
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)

#The Keras library offers a function called to_categorical() that you can use to one hot enode
#integer data. The sequence has an example of all known values
#so we can use the to_categorical() function directlyX
= X.astype('float32')/255.0
```

```
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)

#Splitting the data to Test and Train

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")

#Getting Started with Convolutional Neural Networks (CNN)
```

#MODEL BUILDING

```
early_stop_loss = EarlyStopping(monitor='loss', patience=3, verbose=1)
early_stop_val_acc = EarlyStopping(monitor='val_accuracy', patience=3, verbose=1)
model_callbacks=[early_stop_loss, early_stop_val_acc]

#Add Layers(Conv, Maxpool, Flatten, Dense, Dropout)

#defining our model, All the layers and configurations
def load_CNN(output_size):
  K.clear_session()
  model = Sequential()
  model.add(Dropout(0.4,input_shape=(224, 224, 3)))

  model.add(Conv2D(256, (5, 5),input_shape=(224, 224, 3),activation='relu'))
  model.add(MaxPool2D(pool_size=(2, 2)))
  #model.add(BatchNormalization())

  model.add(Conv2D(128, (3, 3), activation='relu'))
  model.add(MaxPool2D(pool_size=(2, 2)))
  #model.add(BatchNormalization())

  model.add(Conv2D(64, (3, 3), activation='relu'))
  model.add(MaxPool2D(pool_size=(2, 2)))
  #model.add(BatchNormalization())

  model.add(Flatten())
  model.add(Dense(512, activation='relu'))
```

```
 model.add(Dropout(0.3))
 model.add(Dense(256, activation='relu'))
 model.add(Dropout(0.3))
 model.add(Dense(128, activation='relu'))
 model.add(Dropout(0.3))

 model.add(Dense(output_size, activation='softmax'))

 return model
```

#Building Model(Summary, Compile, Fit, Predict)

```
**#Model Summary**
#Building a model based on the above defined function
model = load_CNN(6) #Number of Columns / Outputs
model.compile(loss='categorical_crossentropy',optimizer=Adam(lr=0.001),metrics=['accuracy'])
model.summary() #to print model summary
weights = model.get_weights() #to get the weights from our model


**#Fitting Model**
#some arrays to store the result of each model (model trained on each bath size)
histories_acc = []
histories_val_acc = []
histories_loss = []
histories_val_loss =
[]


model.set_weights(weights)
h=model.fit(X_train,y_train,
          batch_size=16,
          epochs=7,
          verbose=1,
          callbacks=[early_stop_loss],
          shuffle=True,
          validation_data=(X_test, y_test))

model.summary() #to print model summary

model.set_weights(weights)
```

```
h=model.fit(X_train,y_train,
        batch_size=16,
        epochs=15,
        verbose=1,
        callbacks=[early_stop_loss],
        shuffle=True,
        validation_data=(X_test, y_test))
```

#Evaluation And Model Saving

```
**#Accuracy, Loss**
#printing the keys we have for the stores values
print(h.history.keys())

histories_acc = []
histories_val_acc = []
histories_loss = []
histories_val_loss = []

#appending the data for each epoch in a arr, and for each batch size
histories_acc.append(h.history['accuracy'])
histories_val_acc.append(h.history['val_accuracy'])
histories_loss.append(h.history['loss'])
histories_val_loss.append(h.history['val_loss'])

#converting into numpy arrays
histories_acc = np.array(histories_acc)
histories_val_acc = np.array(histories_val_acc)
histories_loss = np.array(histories_loss)
histories_val_loss  =  np.array(histories_val_loss)

#here we have 3 columns and 6 rows each,ever row representss differnet bath size,#every
column represent different epoch scores.
print('histories_acc',histories_acc ,
    'histories_loss', histories_loss,
    'histories_val_acc', histories_val_acc,
    'histories_val_loss',  histories_val_loss)
```

*****Loading a Test Image & Making a Test Prediction*****

```
#Predicting the image's classes
#individual scores for each class as well as class with the highest score is printed

#making predictions ,storing result as array of probabilities of each class predicted
predictions = model.predict([X_test[8].reshape(1, 224,224,3)])
predictions


for idx, result, x in zip(range(0,6), found, predictions[0]):
  print("Label: {}, Type : {}, Species : {} , Score : {}%".format(idx, result[0],result[1],
  round(x*100,3)))

#predicting the class with max probability
ClassIndex=np.argmax(model.predict([X_test[image_number-1].reshape(1, 224,224,3)]),axis=1)
#getting the index of the class which we can pass
#to the boat_types list to get the boat type name
ClassIndex

print(found[ClassIndex[0]])

#loading Test Data
image_number = random.randint(0,len(X_test))
print(image_number)
#plotting the test image
plt.figure(figsize=(8, 8))
plt.imshow(X_test[image_number])

#loading Test Data
image_number = random.randint(0,len(X_test))
print(image_number)

#plotting the test image
plt.figure(figsize=(8, 8))
plt.imshow(X_test[image_number])
```

#Model Saving and Loading

```
h5_path=r'/content/drive/MyDrive/Colab Notebooks/Digital Naturalist/final_model.h5'
model.save(h5_path)

#saving necessary model files model_json
= model.to_json() #indent=2
```

```
with open("final_model.json", "w") as json_file:
    json_file.write(model_json)

# serialize weights to H5
model.save_weights("final_model.h5")
print("Saved model to disk")
```

:

GITHUB AND PROJECT DEMO LINK:

https://www.youtube.com/watch?v=gEhWVIZAVL0&ab_channel=ArunKumar