Visvesvaraya Technological University

Belagavi-590014, Karnataka

Project Report on

"ANIMAL DETECTION IN MAN-MADE ENVIRONMENT"

Submitted in partial fulfilment of the requirements for the award of

Bachelor of Engineering

in

Computer Science and Engineering

Submitted By

AKASH BHANGI	2JI19CS005
ANKITA NAGARALE	2JI19CS010
BHAVANA KHOT	2JI19CS015
YAMETKAR SNEHAL CHANDRAKANT	2JI19CS060

Under the Guidance of

PROF. PRATIK J DESHPANDE

Department of Computer Science and Engineering

Sri Bhagawan Mahaveer Jain Educational & Cultural Trust's

Jain College of Engineering

Belagavi - 590014

Academic Year 2022-23

Sri Bhagawan Mahaveer Jain Educational & Cultural Trust's

Jain College of Engineering

Belagavi - 590014

Department Of Computer Science and Engineering

Certificate

This is to certify that the Project Final Phase entitled **"ANIMAL DETECTION IN MAN-MADE ENVIRONMENT"** is carried out by **AKASH BHANGI (2JI19CS005), ANKITA NAGARALE (2JI17CS010), BHAVANA KHOT (2JI19CS015), YAMETKAR SNEHAL CHANDRAKANT (2JI19CS060)**, bonafide student of **Jain College of Engineering, Belagavi**, in partial fulfilment for the award of **Bachelor of Engineering** in **Computer Science and Engineering** from **Visvesvaraya Technological University, Belagavi**, during the academic year **2022-23.** It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report. The Project Final Phase report has been approved as it satisfies the academic requirements in respect of Project Final Phase work prescribed for the said degree.

Prof . Pratik J Deshpande
Guide

Dr. Uttam Patil
HOD, CSE

DECLARATION

We hereby declare that entire work embodied in this report entitled " **ANIMAL DETECCTION IN MAN-MADE ENVIRONMENT**" , has been carried out by us at Department of Computer Science and Engineering , Jain College of Engineering, Belagavi under the supervision of Prof. Pratik J. Deshpande. The report has not been submitted in part or full for the award of any degree of this or any other university.

Ms. Ankita Nagarale (2JI19CS010)

Mr. Akash Bhangi (2JI19CS005)

Ms. Bhavana Khot (2JI19CS015)

Ms. YAMETKAR SNEHAL CHANDRAKANT (2JI19CS060)

Date:

Place: Belagavi

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crowned the efforts with success.

We would like to profoundly thank management of **Jain College of Engineering, Belagavi** for providing such a healthy environment for the successful completion of Project work.

We would like to express our thanks to the Director and Principal **Dr. J . SHIVAKUMAR** for his encouragement that motivated us for the successful completion of Project work.

It gives us immense pleasure to thank **Dr.Uttam Patil** , Head of Department for his constant support and encouragement.

Also, we would like to express our deepest sense of gratitude to our project guide, **Prof. Pratik J Deshpande,** Assistant Professor, Department of Computer Science & Engineering for his constant support and guidance throughout the Project work.

Also, We would like to thank all other teaching and non-teaching staff of Computer Science Department who has directly or indirectly helped us in the completion of the Project work.

Last, but not the least, we would hereby acknowledge and thank our parents who have been a source of inspiration and also instrumental in the successful completion of the Project work.

ABSTRACT

The development of a stealth animal detection system using deep learning algorithms and Python programming. The goal is to detect camouflaged animals that blend into their natural environment, which is a challenge for both humans and artificial intelligence. Image processing methods such as OpenCV are used to analyze and predict these animals. In addition, the automatic detection of animals that have strayed into human inhabited areas has important security and road safety applications. Therefore, this proposed work develops an algorithm to detect animals in man-made environments, which can be achieved by applying effective deep learning algorithms such as Convolutional Neural Networks (CNN object detection framework. Animal images are learned by a CNN to localize prominent features in the images, and such information can be automatically extracted by deep learning. The overall recognition accuracy of animals from the images is evaluated by comparing this method with the CNN method. Empirical evidence shows that the detectors are unable to generalize from training images of animals in their natural habitats to deployment scenarios of man-made environments. Animal detection and classification can help to prevent animal-vehicle accidents, trace animals and prevent theft.

List of Figures

SYSTEM DESIGN

METHODOLOGY

IMPLEMENTATION

RESULTS

CONTENTS

CHAPTER 1

INTRODUCTION

Object detection is an important field in computer vision that has seen very rapid improvements in recent years using deep learning Examining wild animals in their natural environment is an essential task in ecosystem. Due to the enormous growth in human inhabitants and the increase in hunt of economic development makes excessive exploitation of natural resources, fast, innovative and significant changes in the Earth's ecosystems. An expanding region of the land surface has been changed by human activity, modifying natural life populace, habitat and behavior. More fatally, many wild animals on the Earth have disappeared, and many species are allocated into new places where they can disturb all natural and human resources. Deep neural networks are the collection of algorithms that have placed new records in precision for several vital problems. Convolutional neural network (CNN) is a type of deep neural networks, most generally applied for investigating visual images. Compared to other image classification algorithms, CNNs employ fairly modest pre-processing. This liberty from past knowledge and human intervention in feature design is a key benefit of Convolutional neural network (CNN). They have several applications in the field of image and video recognition, recommendation systems, image classification and medical image processing. The camera traps are activated; each time motion is sensed, and a short video of animal activities are recorded with details about the surroundings (illumination levels, humidity, temperature, and location).

Camera-trap networks are vital for acquisition of wide life data without any disturbance. Moreover, camera trap networks are economically feasible, easy to deploy at larger space and have low maintenance cost; as a result, they are widely used for wildlife monitoring. Image Classification analyses, identifies and discover several properties of an image and organizes the image data into various categories using several algorithms. It mainly employs two characteristic phases of processing: training and testing. In the training phase, a unique 1 identity of each category is obtained. In the testing phase, these unique identities are used to classify the image data into categories. A neural network is a series of algorithms that analyses a set of data and recognizes the underlying relationships within the data. They are the workhorses of deep learning. Generally, in a recognition system, when an input image is provided, features are extracted from the image. These are used by the network to train itself from the training data and organizes the

data into classes. The gained knowledge from the training is used in predicting the test data based on the features and classifies them accordingly.

Camouflage is the ability to hide from predators by matching body patterns, textures and colors to the environment. Camouflage animal detection is a method of detecting animals in the foreground that are hidden in the background image. Organisms use camouflage to disguise their location, identity, and movement. This allows the prey to avoid the predator and the predator to sneak up on the prey. Our goal is to segment animals in cloaks that are hidden in the environment. This is difficult because the environment looks similar to that of animals. Camouflage is an attempt to hide the texture of a foreground object with the frame texture of a background image. Camouflage detection or de-camouflage methods are basically used to detect animals in the foreground hidden in background images.

A recognition system can be employed with identification and verification. Identification is where the given image is compared with all the other images and produces a ranked list of matches while the Verification is where the given image is compared, and the identity of the animal is confirmed or denied.

CHAPTER 2

LITERATURE SURVEY

1. “H. Yao, S. Zhang, "Coarse-to Fine Description for Fine Grained Visual Categorization," in IEEE Transactions on Image Processing, vol. 25, no. 10, pp. 4858-4872, Oct. 2016”

Yao S. et al, proposes fine-grained visual categorization, which targets to classify the objects belonging to the same species. This novel description only required the original image as input, but could automatically generate visual descriptions discriminative enough for fine-grained visual categorization. The major drawback of fine-grained visual categorization is it is computationally expensive and not suitable for large-scale image.

2. “L. Xie, J. Wang, "Fine-Grained Image Search," in IEEE Transactions on Multimedia, vol. 17”

Xie. Et al, proposes that instance search should not return only near duplicate images, but also fine-grained results, which is usually the actual intention of a user. It introduces a baseline system using fine-grained classification scores by constructing large scale database where the reference images are compressed at constant bit rate levels by JPEG encoders with different optimization methods. To distinguish subtle differences, the comparison method is utilized to rank them in subjective experiments. The major drawback of fine-grained results is duplication occurs while classifying the objects belonging to the same species. The next paper is about template matching Algorithm used for identifying small parts of an image which should match the template image.

3. “Fang, Y.et al, (2016). “Motion based animal detection in aerial videos.” Procedia Computer Science, 92, 13-17.”

Fang, Y.et al, proposes an approach for moving animal detection by taking benefit of global patterns of pixel motion. This paper used the segmented regions, another threshold was used to filter out negative candidates, which could belong to the background. The main drawback is the complexity was high.

4. “J. Tanha, "Multiclass Semi supervised Learning for Animal Behavior Recognition from Accelerometer Data," 2012 IEEE 24th International Conference on Tools with Artificial Intelligence, Athens, 2012”

J. Tanha, et al, proposes Multiclass semi-supervised learning algorithm that uses a base classifier in combination with a similarity function applied to all data to find a classifier that maximizes the margin and consistency over all data and used labelling methods. For each and every feature was labelled. The main drawback was labelling was not easy and time consuming.

5. “F. Tu, S. Yin, "Deep Convolutional Neural Network Architecture With Reconfigurable Computation Patterns," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 25, Aug.2017, Doi: 10.1109/TVLSI.2017.2688340.”

Tu, S. Yin et al, designed a DCNN acceleration architecture called deep neural architecture (DNA), with reconfigurable computation patterns which comprises of a data reuse pattern and a convolution mapping method for different models. The major drawback is its time consuming since pattern has to been compared to every part of image.

This paper presented a large-scale study of animal detection with deep learning where 8 state of the art detectors were compared in a wide range of configurations. A particular focus of the study was to evaluate their generalization ability when training and test scenarios do not match. It was shown that none of the detectors can generalize well enough to provide usable models for deployment, with missed detections on previously unseen backgrounds being the main issue. Attempts to increase recall using tracking and multimodal pooling proved ineffective. Synthetic data generation using segmentation masks to extract animals from images of natural habitats and inserting them in target scenes was shown to be an effective solution. An almost fully automated way to achieve this was demonstrated by the competitiveness of coarse unsupervised masks with precise manualizes in terms of the performance of detectors trained on the corresponding synthetic images.

CHAPTER 3

PROBLEM STATEMENT

An increasing area of land surface has been transformed by human action, altering wildlife population, habitat and behavior. More seriously, many wild species on Earth have been driven to extinction, and many species are introduced into new areas where they can disrupt both natural and human systems.

This system is necessary to address the potential dangers and threats posed by animals, especially in areas where human-animal interactions are common, such as farms, residential areas, and wildlife reserves. Traditional methods of animal detection and warning systems may be insufficient, as they are often labor-intensive, expensive, and may not be effective in detecting different types of animals accurately.

The problem identified is the need for a reliable and accurate animal detection and alarm system that can improve safety and security in areas prone to human-animal interactions. Thus monitoring wild animals is essential as it provides researchers evidences to inform conservation and management decisions to maintain diverse, balanced and sustainable ecosystems in the face of those changes.

CHAPTER 4

OBJECTIVES

- The main objective of this project is to discover camouflaged animals.
- Use the CNN model to save the image to the dataset.
- Train your dataset using OpenCV techniques.
- Upload the image stored in the record to your website.
- Perform pretreatment to remove noise and resize.
- Easily separate animals from the background by saturating the image with grayscale, binarization, andthresholding.
- Predictions from captured animals.

Overall, the objective of creating an animal detection and alarm system is to provide an advanced solution that can improve safety and security in areas where human-animal interactions are common.

CHAPTER 5

REQUIREMENTS

- **Functional Requirements**
    1. Graphical User interface with the User.
- **Software Requirements**

  For developing the application, the following are the Software Requirements:

    1. Python
    2. Django

- **Operating Systems supported**
    1. Windows 7
    2. Windows XP
    3. Windows 8

- **Technologies and Languages used to Develop**
    1. Python

- **Debugger and Emulator**
    1. Any Browser (Particularly Chrome)

- **Hardware Requirements**
    1. Processor: Pentium IV or higher
    2. RAM: 256 MB
    3. Space on Hard Disk: minimum 512MB

CHAPTER 6

SYSTEM DESIGN

6.1. EXISTING SYSTEM

- According to previous research in crop's security, developing countries, which are using traditional storage facilities for staple food crops, can't protect them, leading to 20- 30% loss of agricultural products such as rice, corn etc.
- Currently available solutions targets only insects, pests and grain pathogens. While other study states 5 to 10% loss in rice crops on average, in Asia is due to damage caused by rodents.
- This paper is oriented to accentuate the methods to solve such problems like identification of rodents, threats to crops and delivering real time notification based on information analysis and processing without human intervention. In this device, mentioned sensors and electronic devices are integrated using Python scripts. Basedon attempted test cases, we were able to achieve success in 84.8% test cases.

EXISTING BLOCK DIAGRAM

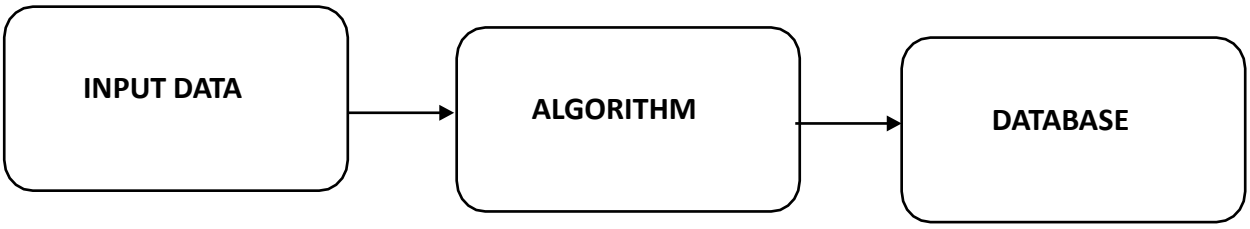

Fig 6.1: Existing Block Diagram

6.2. PROPOSED SYSTEM

- The main aim of our project is to protect the crops from damage caused by animal as well as divert the animal without any harm.
- Animal detection system is designed to detect the presence of animal and offer a warning. In this project we usedto detect the movement of the animal.
- In this project, we will monitor the entire farm at regular intervals through a camera which will be recording thesurrounding throughout the day. With the help of a deep learning model, we detect the entry of animals and we play appropriate sounds to drive the animal away.
- This report specifies various libraries and concepts of convolutional neural networks used to create the model.
- If any of the Animal is detected using CNN algorithm in the Agricultural land and send email.

PROPOSED BLOCK DIGRAM

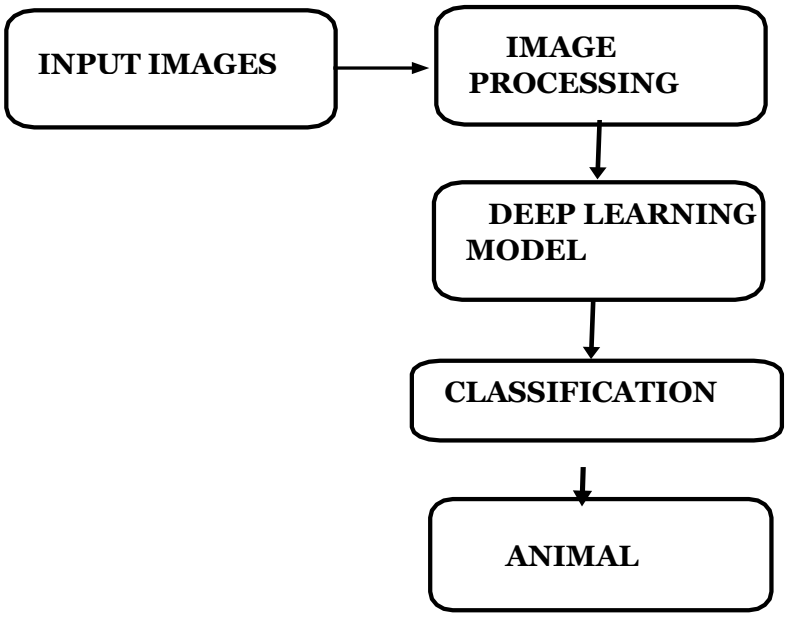

Fig 6.2: Proposed Block Diagram

6.3. ARCHITECTURE DIAGRAM

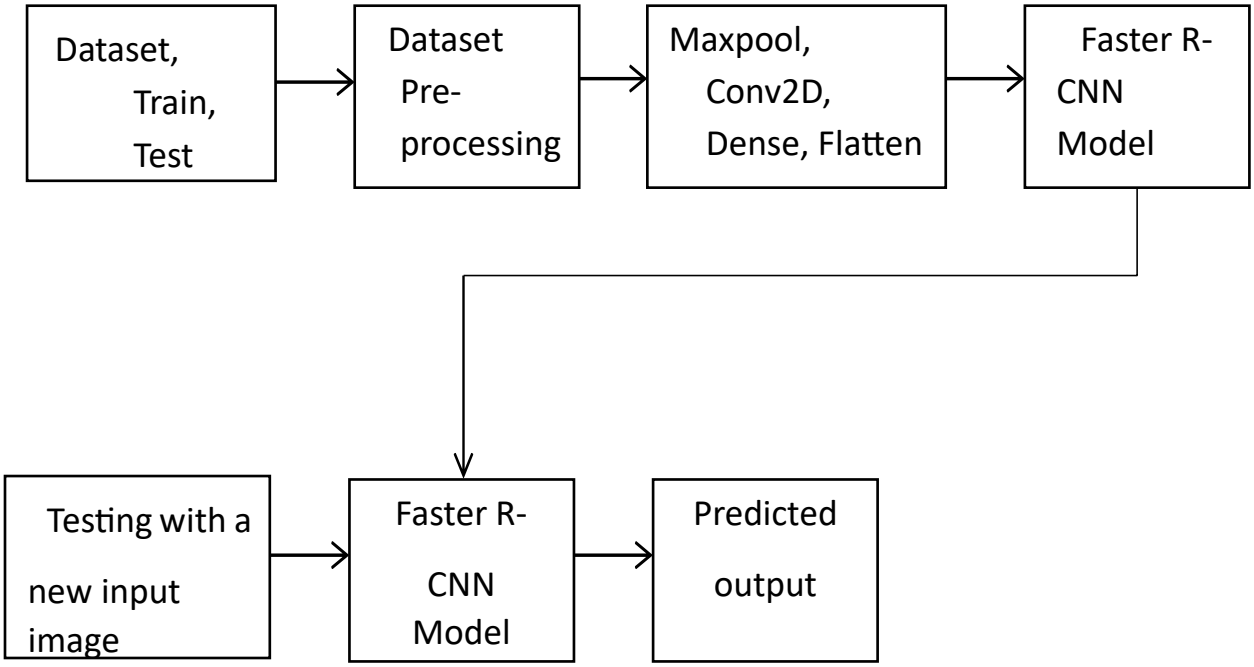

Fig 6.3: Architecture Diagram

6.4. USECASE DIAGRAM

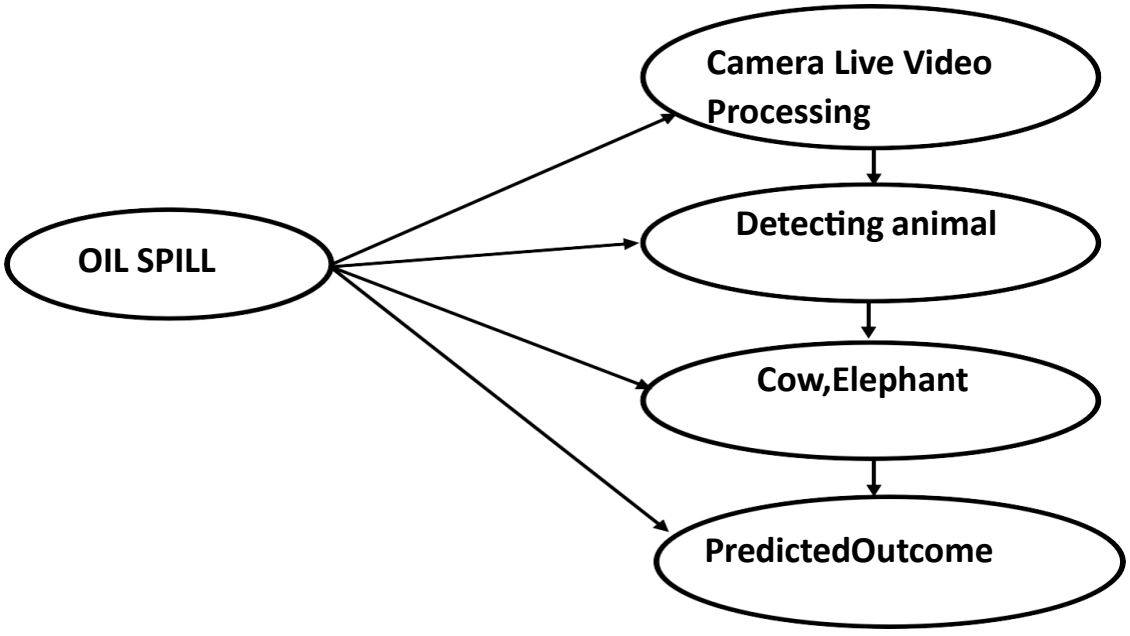

Fig 6.4: Usecase Diagram

6.5. ACTIVITY DIAGRAM

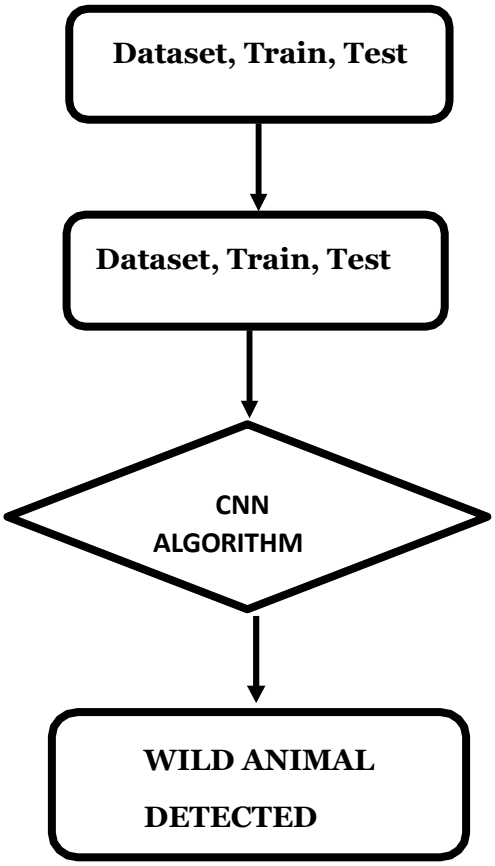

Fig 6.5: Activity Diagram

6.6. CLASS DIAGRAM

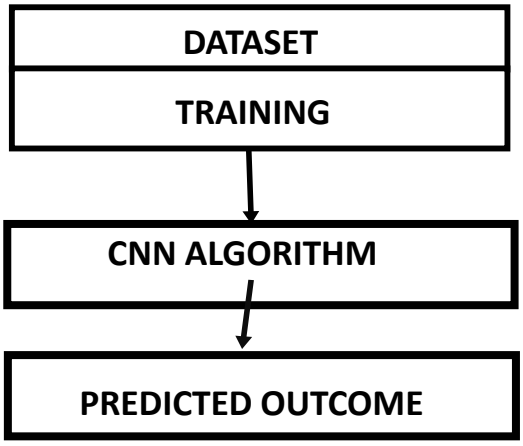

Fig 6.6: Class Diagram

6.7. SEQUENCE DIAGRAM

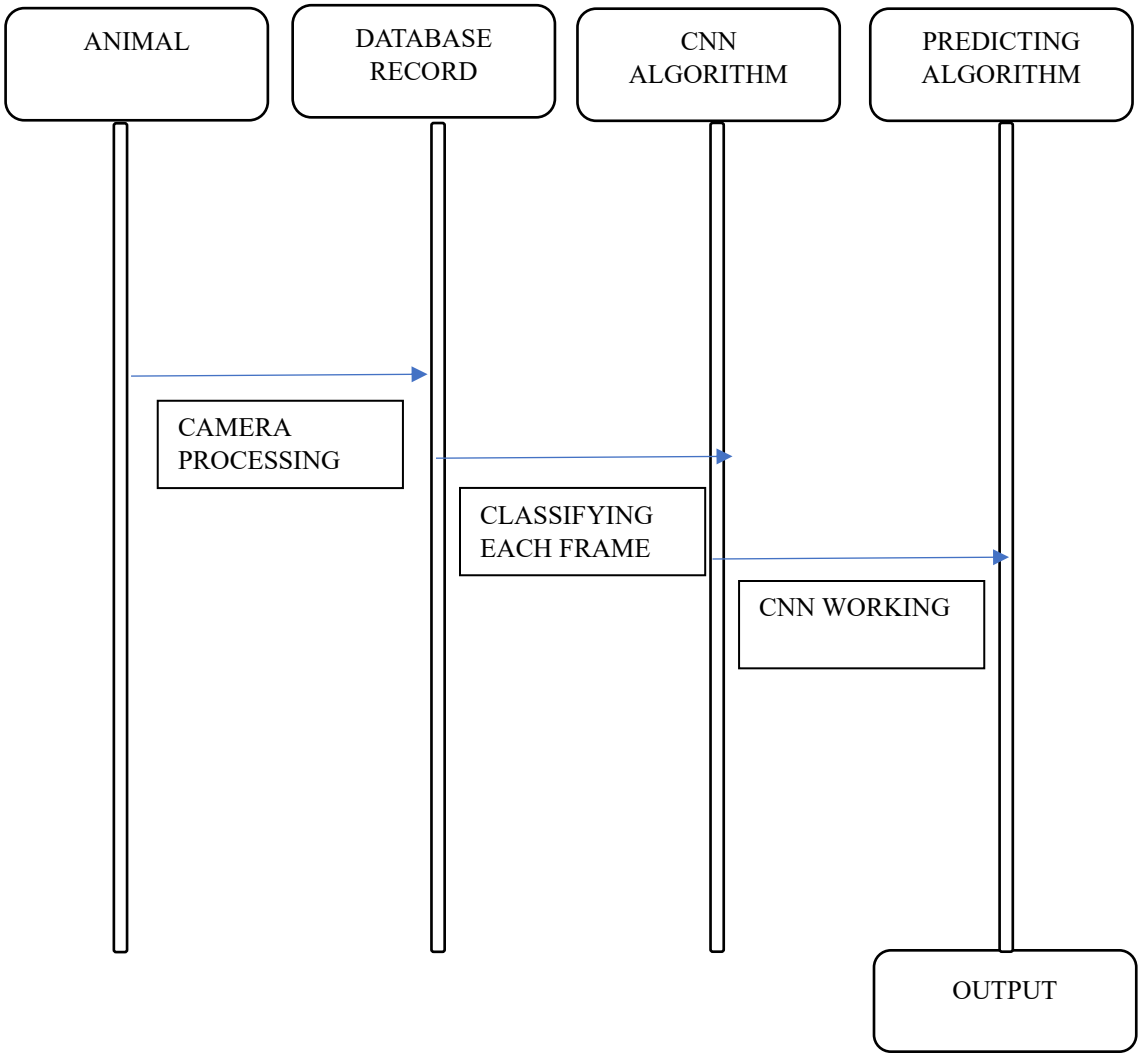

Fig 6.7: Sequence Diagram

CHAPTER 7

METHODOLOGY

7.1. FLOW CHART

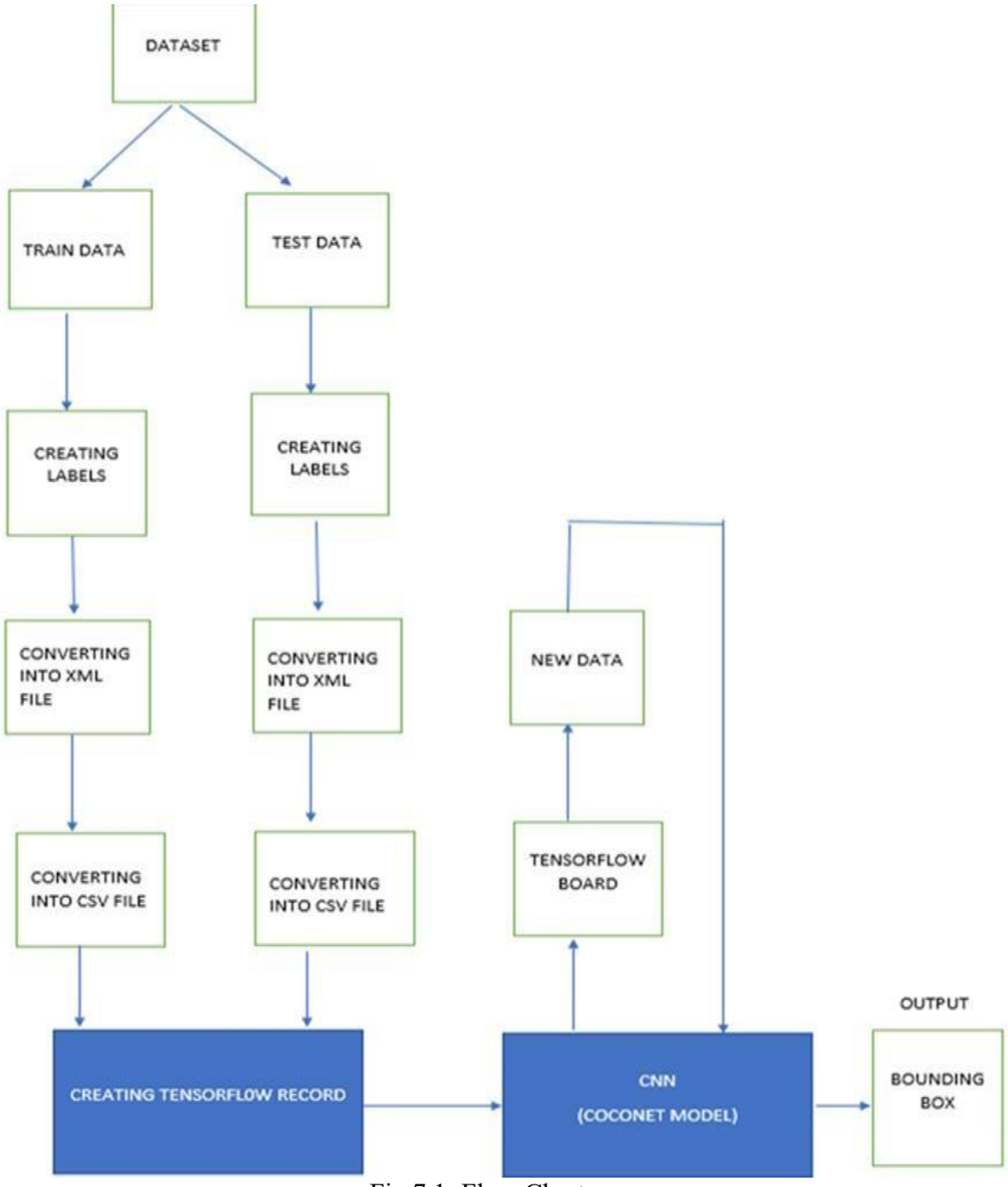

Fig 7.1: Flow Chart

Image Acquisition: You must first select the camouflaged image, then capture the features of the image, take a snapshot of the features, and load the feature image into your system. The pixel values in the image are concentrated near a small area.

Contrast Enhancement: The original image is the image passed to the system, and the system output after contrast enhancement is the enhanced image. This is the image after removing the sharp edges.

Converting RGB to HSI: RGB image size is M x N x 3, 3D consists of 3 image planes (red, green, blue). If all three components are the same, no transformation is undefined. In general, the RGB pixel range is [0,255], in which case the pixel range is [0,1]. Pixel area conversion can be performed by calculating the component. Hue, saturation, intensity.

Grayscale image: The image is converted to grayscale by converting the RGB values in the image. Converting a color image to a grayscale image with noticeable highlights is a complex process. The converted grayscale image can lose the contrast, sharpness, shadows, and texture of the shaded image. To preserve the contrast, the sharpness, shadows, and structure of the shading image suggested different calculations.

Threshold image: Automatic thresholds are a great way to extract useful pixel-encoded information while minimizing background noise. This is achieved by using a feedback loop to optimize the threshold before converting the original grayscale image to a binary image. The idea is to split the image into two parts. Background and foreground. We used unit tests to validate the applied threshold image.

Segmented image: Segmentation divides the image into separate areas, each containing pixels with similar attributes. For the analysis and interpretation of an image to be meaningful and useful, the area must be closely associated with the imaged object or feature of interest. We used unit tests to validate the segmentation of feature images and extract ROI.

ROI: Return on investment is a performance measure used to assess the performance of financing and to compare the performance of a series of exclusive investments. ROI seeks to instantly measure the amount of return on a particular financing compared to the price of the investment. We used integration tests to validate the ROI of test images and trained datasets.

Prediction: Based on the clusters and nodes identified from the test image, the trained dataset is tested and objects are extracted from the camouflage image.

Data Flow Diagram: A DFD that graphically represents a function or process that collects, manipulates, stores, and distributes data between a system and its environment, and between components of the system. The visual representation makes it an excellent communication tool between users and system designers. The structure of the DFD allows you to start with a high-level overview and extend to a detailed diagram hierarchy.

DFD is often used for the following reasons:

- System logical information flow.
- Determining the physical requirements of the system structure.
- Simple notation.

Manual and automated system requirements specifications

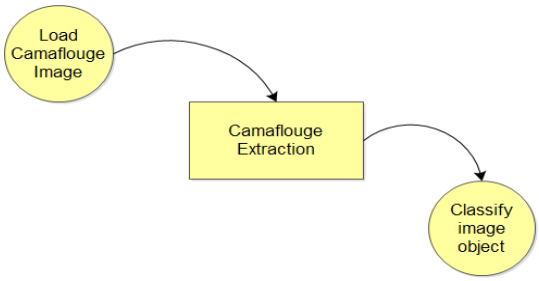

Figure 7.2: DFD of System

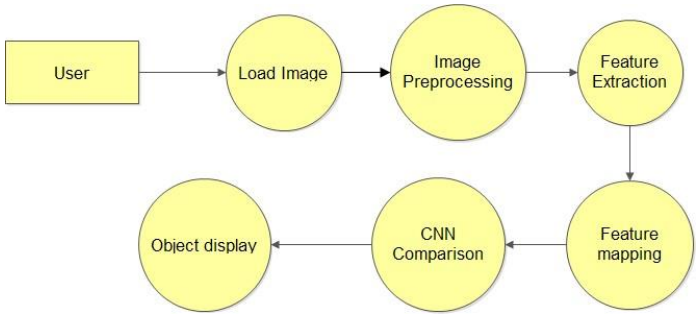

Figure 7.3: Level 1 Admin DFD

Algorithms:

CNN

When it comes to machine learning, artificial neural networks work very well. Artificial neural networks are used in various classification tasks such as images, sounds, and words. Different types of neural networks are used for different purposes. For example, use a recurrent neural network for word-order prediction, more precisely LSTM, and similarly a Convolution Neural Network for image classification.

Neural Network. A typical neural network has three types of layers:

1. **Input Layers:** This is the level that gives input to the model. The number of neurons in this layer corresponds to the total number of features in the data (pixels in the case of images).
2. **Hidden Layer:** Input from the input layer is sent to the hidden layer. There may be many hidden layers, depending on the model and data size. Each hidden layer can have a different number of neurons. This is usually more than the number of features. The output of each layer is calculated by multiplying the output of the previous layer by the learnable weight of that layer, adding a learnable bias, and then multiplying by a matrix an activation function that makes the network non-linear.
3. **Output Layer:** The output of the hidden layer is sent to logistic functions such as Sigmoid and Softmax, and the output of each class is converted into the probability value of each class.

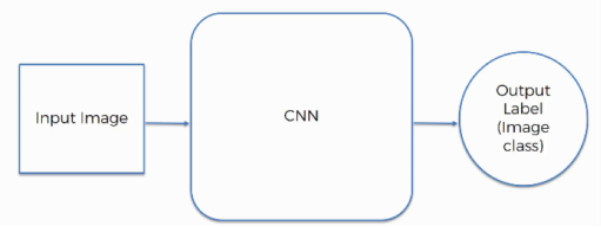

Fig 7.4: Neural Network

Steps involved in CNN Algorithm

Step1:

(a) Convolution Operation:

The first component of the attack plan is the convolution operation. This step describes a feature detector that essentially acts as a neural network filter. You will also learn about feature maps and learn about the parameters of such maps, how patterns are perceived, the level of recognition, and how the results are mapped.

(b) ReLU Layer

The second part of this step is about the normalized linear unit or ReLU. Let's talk about the ReLU layer and explore how linearity works in the context of convolutional neural networks. You don't need to understand CNN, but simple lessons to improve your skills were fine.

Step2: Pooling

This part describes pooling and gives you an accurate understanding of how it works in general. However, the nexus here is a particular type of pooling. Maximum pooling. However, it describes different approaches, including average (or total) pooling with a visual interactive tool. This ensures that the entire concept is cleared.

Step 3: Flattening

This is a simple breakdown of the flattening process, how to move from a pooled layer to a flattened layer when working with a Convolution Neural Network.

Step 4: Full Connection

This part summarizes everything described in this section. By learning this, you will have a better understanding of how Convolution Neural Networks work and how the final generated "neurons" classify images.

CHAPTER 8

IMPLEMENTATION

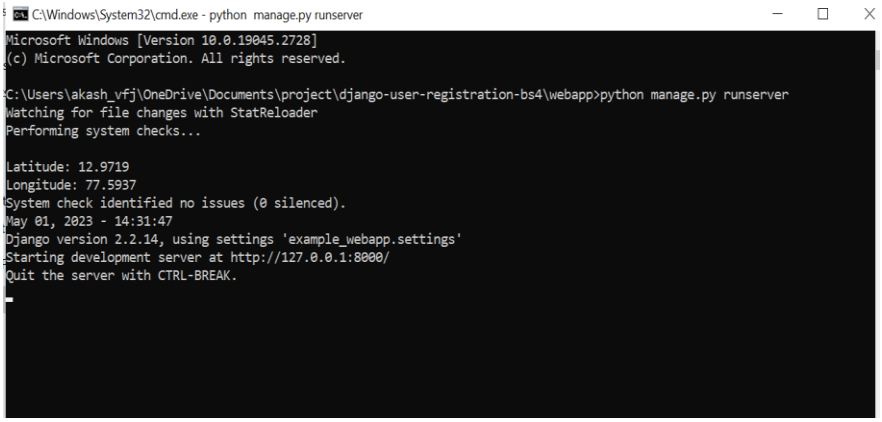

Fig 8.1: Running the project

Steps to run the program:

1. Open the terminal.
2. Run the command: python manage.py runserver
3. The terminal generates a localhost link called http://127.0.0.1:8000/, copy this linkand paste it in any browser.

```
1  from django.db import models
2  from django.contrib.auth.models import User
3  from django.db.models.signals import post_save
4  from django.dispatch import receiver
5  from io import BytesIO
6  import logging
7  from django.core.files.base import ContentFile
8  from placeholder_pics.placeholder import PlaceholderPic
9  from django_thumbs.fields import ImageThumbsField
10
11 import os
12 from django.utils import timezone
13 import datetime
14
15 current_time = datetime.datetime.now()
16 logger = logging.getLogger(__name__)
17
18
```

Fig 8.2: Importing libraries

The figure 8.2 displays the code for importing libraries. From django we are importing user, post-save, receiver.

```
    {% include "socialaccount/snippets/login_extra.html" %}
    {% else %}
    <p>{% blocktrans %}If you have not created an account yet, then please
        <a href="{{ signup_url }}">sign up</a> first.{% endblocktrans %}</p>
    {% endif %}

</div>
<div class="col col-6">


    <form class="login" method="POST" action="{% url 'account_login' %}">
        {% csrf_token %}
        {% bootstrap_form form %}
        {% if redirect_field_value %}
        <input type="hidden" name="{{ redirect_field_name }}"
               value="{{ redirect_field_value }}"/>
        {% endif %}

        <button class="btn btn-primary btn-block" type="submit">{% trans "Sign In" %}</button>
        <div class="row mt-2">
            <div class="col col-6">
                <a class="button secondaryAction" href="{% url 'account_reset_password' %}">
                    {% trans "Forgot Password?" %}</a>
            </div>
            <div class="col col-6 text-right">
                <a href="{{ signup_url }}">sign up</a>
            </div>
```

Fig 8.3: code for login page

The figure 8.3 displays the code for user login. User can login by entering email and password.

```
@csrf_exempt
def animal_detection(request):
    if request.method == "POST":
        my_var = request.POST.get("myVar")
        detection = AnimalDetectionView(my_var=my_var)
        detection.save()
        current_time = datetime.datetime.now()
        detection.end_time = current_time
        detection.save()
        # Retrieve email address of recipient from UserEmail model
        user_email = UserEmail.objects.first()
        email_receiver = user_email.email if user_email else None
        print(email_receiver)
        if email_receiver:
            context = ssl.create_default_context()
            with smtplib.SMTP_SSL('smtp.gmail.com', 465, context=context) as smtp:
                smtp.login(email_sender, email_password)
                message = f'Subject: {subject_main}\n\n{body_main}'
                smtp.sendmail(email_sender, email_receiver, message)
            detection.end_time = current_time
            detection.save()
    return render(request, 'animal/an.html')
```

Fig 8.4: code for retriever email

The figure 8.4 displays the code for retriever email, where user can the email address to whom the alert should be sent.

```
model = await tmImage.load(modelURL, metadataURL);
maxPredictions = model.getTotalClasses();

// Convenience function to setup a webcam
const flip = true; // whether to flip the webcam
webcam = new tmImage.Webcam(400, 400, flip); // width, height, flip
await webcam.setup(); // request access to the webcam
await webcam.play();
window.requestAnimationFrame(loop);

// append elements to the DOM
document.getElementById("webcam-container").appendChild(webcam.canvas);
labelContainer = document.getElementById("label-container");
console.log(labelContainer);


for (let i = 0; i < maxPredictions; i++) { // and class labels
    labelContainer.appendChild(document.createElement("hr"));
    labelContainer.appendChild(document.createElement("div"));
```

Fig 8.5: code for webcam access

The figure 8.5 displays the code for webcam. It allows user to turn on the webcam and detect the animal.

```
    //}
    else if(prediction[i].className == 'Deer'){
            playMyAudio()
            var myVar = "Deer";
            var data = {
                    myVar: myVar
                };
                sendPostRequest(data);
    }
    else if(prediction[i].className == 'Cattle'){
            playMyAudio()
            var myVar = "Cattle";
            var data = {
                    myVar: myVar
                };
                sendPostRequest(data);
    }
    else if(prediction[i].className == 'Monkey'){
            playMyAudio()
            var myVar = "Monkey";
            var data = {
                    myVar: myVar
                };
                sendPostRequest(data);
```

Fig 8.6: code for alert system

The figure 8.6 displays the code for alert sound. Which alert the system user by sound if the wild animal is detected.

```
{% load bootstrap4 %}

{% block head_title %}Web Application{% endblock %}

{% block content %}
<section style="background: white;">
    <div class="container">
        <div class="row">
            <div class="col col-12 pb-5">
                <div class="clearfix"></div>
                <table>
                    <tr>
                        <th>
                            <h1 class=" " style="margin-top: 15rem;">Animal Detection </h1>
                            <p> </p>
                        </th>
                        <td>
                            <img src="https://images3.alphacoders.com/659/thumb-1920-659401.jpg" alt="" st
                            width: 200px;height: 150px;">
                        </td>
                    </tr>
                </table>
```

Fig 8.7: Homepage html code

The figure 8.7 displays the html code of homepage.

```
            </tr>
            {% empty %}
            <tr>
              <td colspan="3">No animal detections found.</td>
            </tr>
            {% endfor %}
          </tbody>
        </table>
      </main>
      <script>
        function filterTable(colIndex) {
          var input = document.getElementById(
            colIndex === 0 ? "my-var-filter" : "start-time-filter"
          );
          var filter = input.value.toUpperCase();
          var table = document.getElementsByTagName("table")[0];
          var rows = table.getElementsByTagName("tr");
          for (var i = 1; i < rows.length; i++) {
          var cell = rows[i].getElementsByTagName("td")[colIndex];
          if (cell) {
            var text = cell.textContent || cell.innerText;
            if (text.toUpperCase().indexOf(filter) > -1) {
              rows[i].style.display = "";
            } else {
              rows[i].style.display = "none";
            }
          }
        }
      }
</script>
```

Fig 8.8: Detection page code

The figure 8.8 displays the html code of detection of animal page. Where we can also observer the probability of detecting animal.

CHAPTER 9

RESULTS

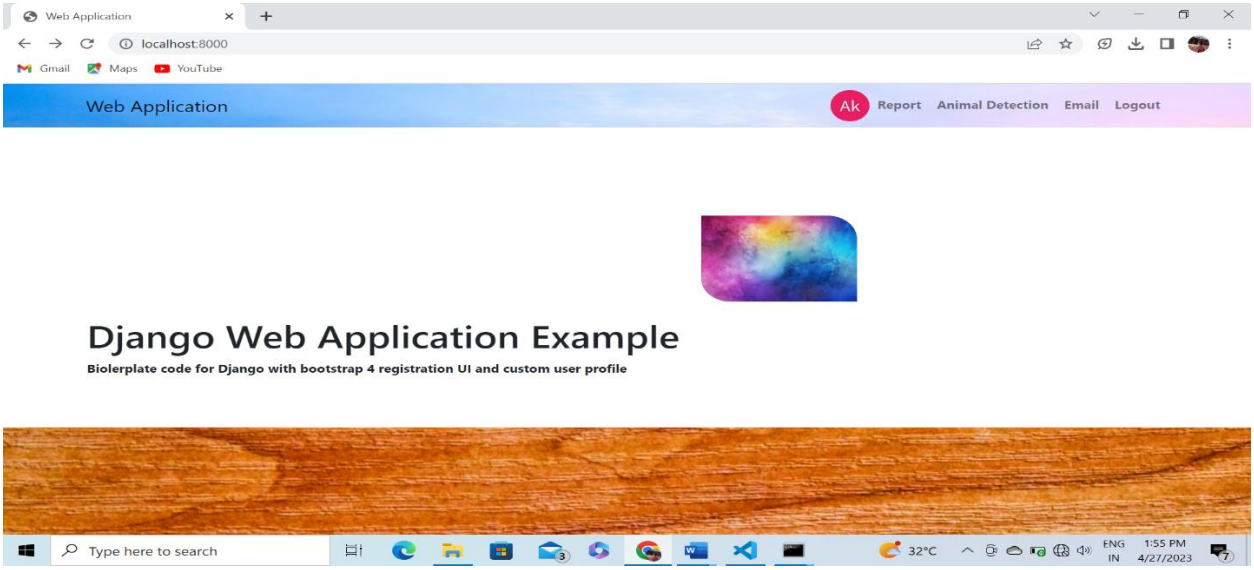

Fig 9.1: Home page

This is the home page of our project "Animal Detection".We can see the login and signup option.

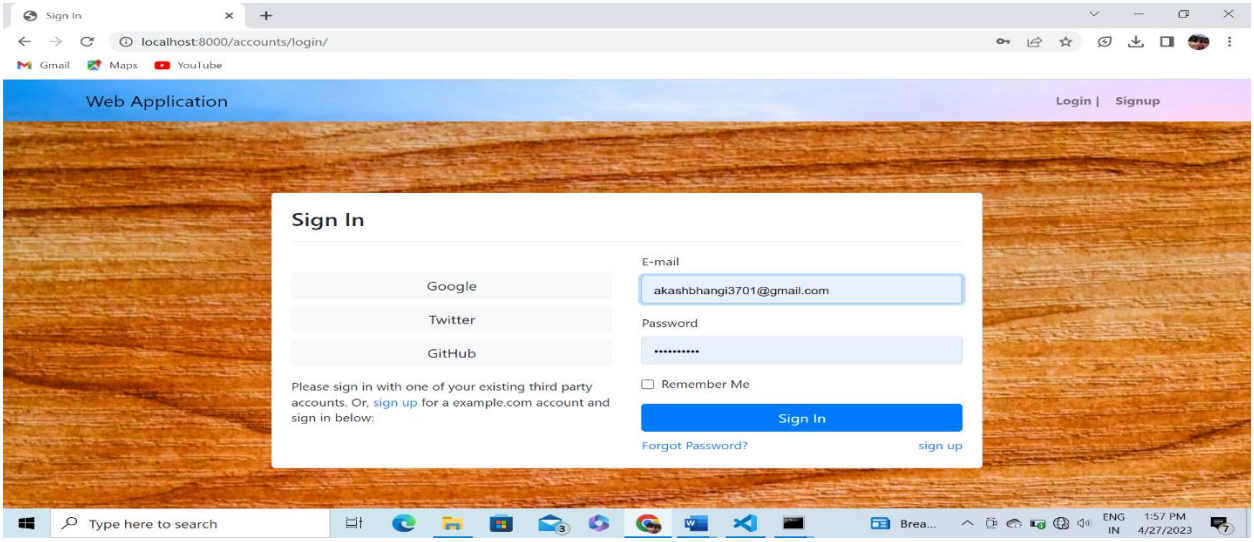

Fig 9.2: Login page

This is login page, registered user can login to account by adding email and password.

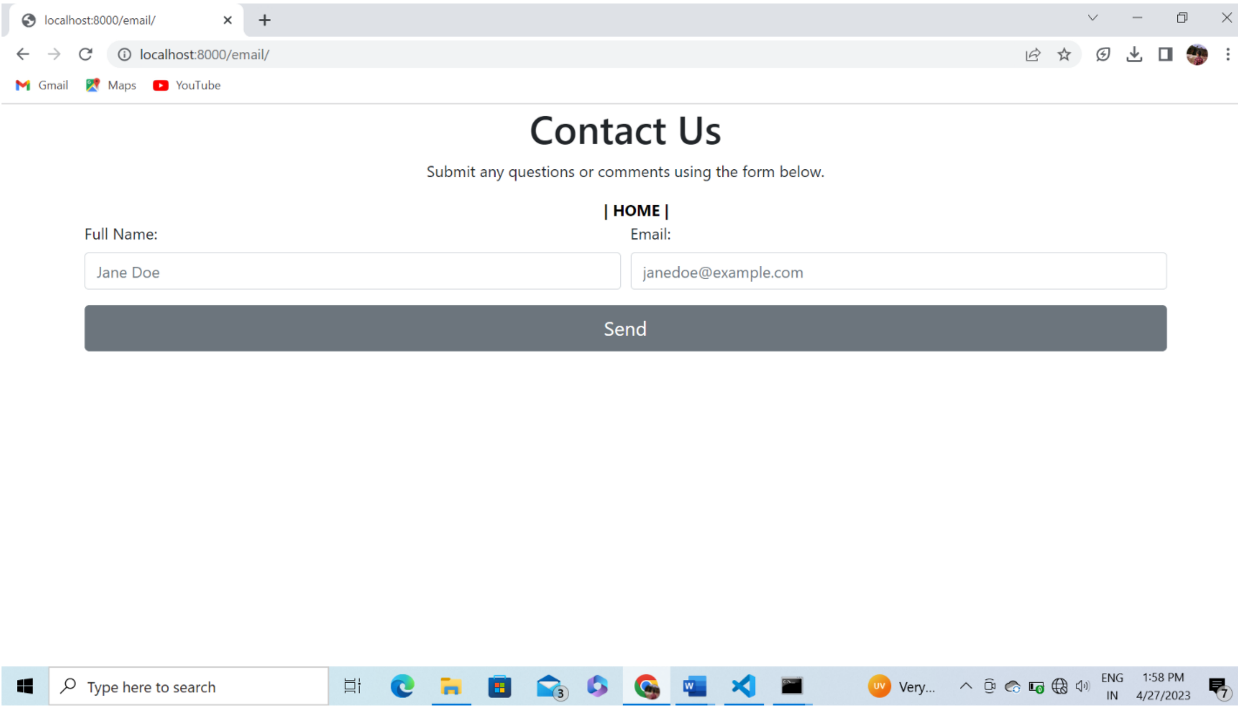

Fig 9.3: Email alert

This is the page where user can set the email address to whom the alert must be sent.

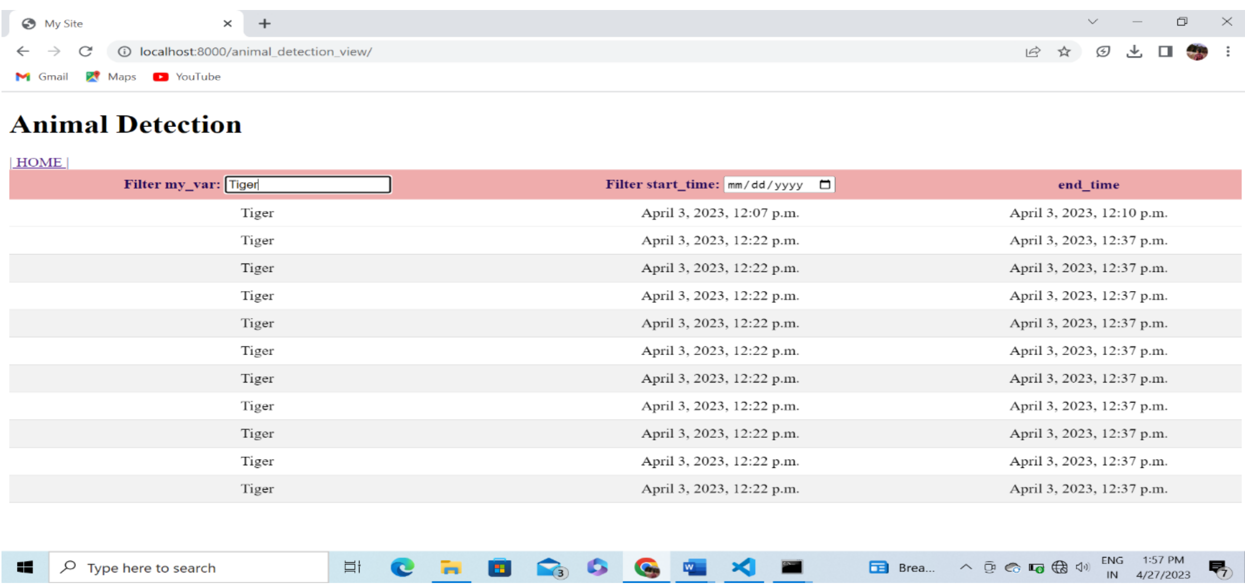

Fig 9.4: Report

This is the page where the user can see the history of detected animals.

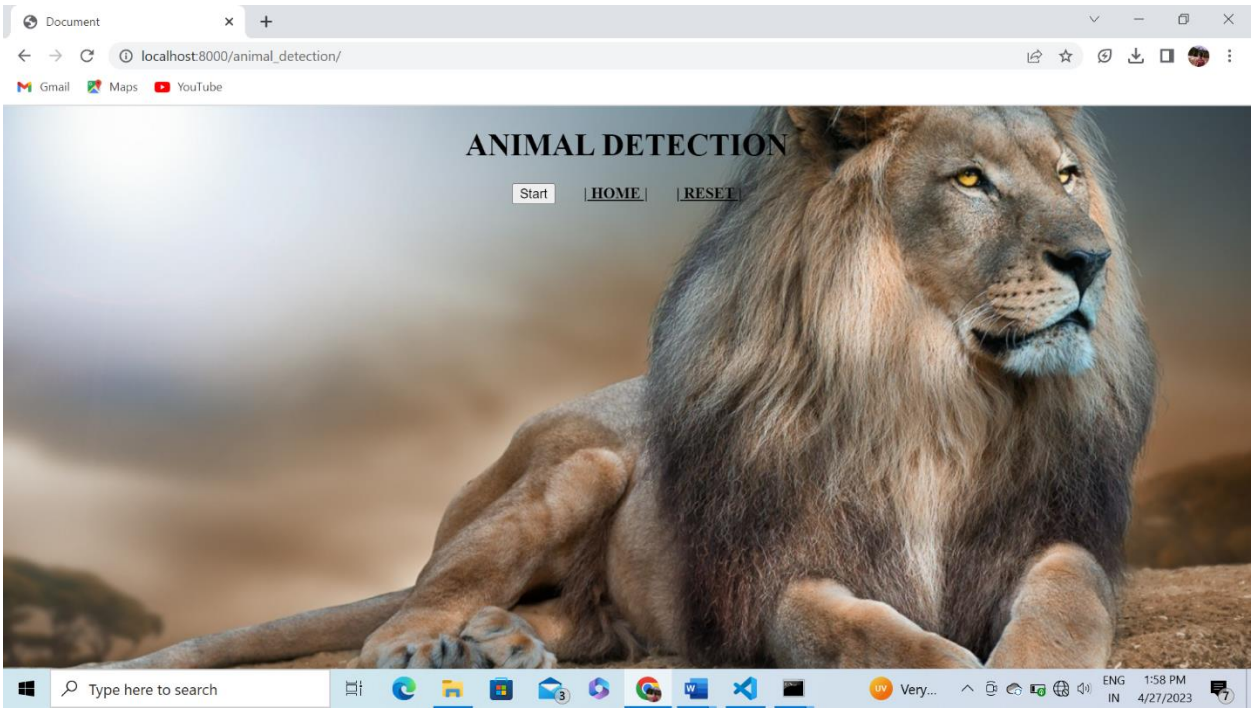

Fig 9.5: Detection page

This is the page where we can see the home, reset and start button. Start button is for turning on the camera for detection.

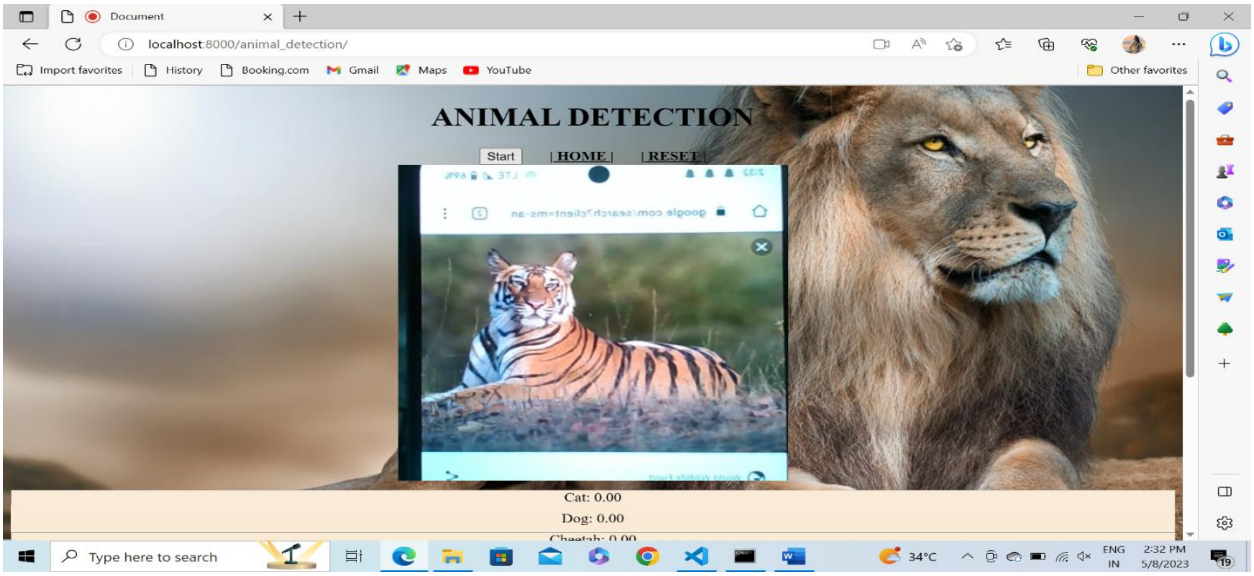

Fig 9.6: Camera detecting animal

This is the page where user can detect the animal and below the user can see the probability of detected animal out of 1.0 rating

CHAPTER 10

TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input	: identified classes of valid input must be accepted.
Invalid Input	: identified classes of invalid input must be rejected.
Functions	: identified functions must be exercised.
Output	: identified classes of application outputs must be exercised.
Systems/Procedures	: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

CHAPTER 11

SYSTEM STUDY

FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are,

- **ECONOMICAL FEASIBILITY**
- **TECHNICAL FEASIBILITY**
- **SOCIAL FEASIBILITY**

ECONOMICAL FEASIBILITY:

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY:

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

SOCIAL FEASIBILITY:

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

CONCLUSION

Most of the proposed work facilitates useful methods for dealing with problems in conditions such as disguise, and the proposed attempts are attempts to approve techniques with different types of images. The multi- resolution-based approach has a variety of characteristics and is chosen because it provides the placement you expect in your target detection application. The wavelet coefficient element from the subblock is used in this embodiment as a profile for detecting seed points. In addition, there is a smooth marking material (surface) and its measurable highlights for better recognition of objects in the camouflaged image. Conflicting aspects, slight deviations from the photo, and performance inspection of the photo invoice related to various conditions and placements are checked in sequence. In the future, the level of resizing and attenuation will depend on the size of the target. In addition, the aspect of an image, which is a mixture of many learning objects in the image, is an important exploratory task to teach you how to find objects in the ideal location. Relying on include sets and numerous destinations is too difficult for a disguise photography topic.

REFERENCE

1. H. Yao, S. Zhang, Y. Zhang, J. Li and Q. Tian, "Coarse-toFine Description for Fine Grained Visual Categorization," in IEEE Transactions on Image Processing, vol. 25, no. 10, pp. 4858-4872, Oct. 2016

2. L. Xie, J. Wang, B. Zhang and Q. Tian, "Fine-Grained Image Search," in IEEE Transactions on Multimedia, vol. 17

3. Fang, Y., Du, S., Abdoola, R., Djouani, K., & Richards, C. (2016). “Motion based animal detection in aerial videos.” Procedia Computer Science, 92, 13-17.

4. J. Tanha, M. V. Someren, M. d. Bakker, W. Bouteny, J. Shamoun-Baranesy and H. Afsarmanesh, "Multiclass Semi supervised Learning for Animal Behavior Recognition from Accelerometer Data," 2012 IEEE 24th International Conference on Tools with Artificial Intelligence, Athens, 2012

5. F. Tu, S. Yin, P. Ouyang, S. Tang, L. Liu and S. Wei, "Deep Convolutional Neural Network Architecture with Reconfigurable Computation Patterns," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 25, no. 8, pp.2220- 2233 ,Aug.2017, Doi: 10.1109/TVLSI.2017.2688340.