EMERGING METHODS FOR EARLY DETECTION OF FOREST FIRES.

NALAIYA THIRAN PROJECT REPORT

IBM-Project-9417-1659003152

TEAM ID : PNT2022TMID08407

Submitted by

G. JEEVITHA	**(814319104015)**
D. LALITHA	**(814319104022)**
V. MEGALA	**(814319104029)**
S.RAJASRI	**(814319104039)**

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINNERING

IN

COMPUTER SCIENCE AND ENGINEERING

DHANALAKSHMI SRINIVASAN ENGINEERING COLLEGE
(AUTONOMOUS)
PERAMBALUR-621212

PROJECT REPORT

1. INTRODUCTION

1.1. Project Overview

Forests are the savior of earth's ecological balance. Automatic fire detection systems, when combined with other elements of an emergency response and evacuation plan, can significantly reduce property damage, personal injuries, and loss of life from fire in the workplace. In this project, the Ai based forest fire detection is implemented by using an algorithm called YOLOv3. To balance the efficiency and accuracy, the model is fine-tuned considering the nature of the target problem and fire data.

This project is to study the detection of flames in a video by using motion and edge detection techniques. This is an improved method over all the existing ones. This method detects the edges of the flames properly by removing the noises in the flames. This paper focuses on identifying gray cycle pixels for the detection of flame. It's an optimizing technique to detect the flame, which is generated because of smoke and the spreading of fire pixels and the area of flame. These systems can be used to reduce false fire detect.

The novel system simulates the existing fire detection techniques with above given new techniques of fire detection and gives an optimized way to detect the fire in terms of less false detection of fire by giving the accurate result of fire occurrence.

To reduce the risk and prevent the forest fire they are big offerings to fight it like planes, fire brigade trucks, also extinguishers to small areas which depends upon the severity of the fire and leads to large investment by concerned agencies. Forest fires are highly destructive and are uncontrollable when it starts spreading over the area. Fire causes respiratory problems for living beings even they are living several kilometers away. The 2019, Australia wildfires almost 15 million acres have been burned and the fire also killed about one billion animals.

1.2. Purpose

Forest fires as of late have been annihilating both for normal biological system, biodiversity and woodland economy. With expanding populace weight and change in worldwide atmosphere situation, there is an expansion in level of fires that are a significant reason for declining Indian woodlands. As indicated by woodland study report of India, 50 % of backwoods regions in nation are fire inclined (going from 50 to 90 % in certain conditions of nation). Around 6 % of the woods are inclined to extreme fire harms. The reason for this planned framework is to

manufacture a dependable fire location framework so as to know dynamic status of backwoods temperature in specific conditions. It is about the sensors and dynamic checking framework to dodge a significant fire and genuine harm to woods.

2. LITERATURE SURVEY

2.1 Existing Problem

The existing system for detecting fire are smoke alarms and heat alarms. The main disadvantage of the smoke sensor alarm and heat sensor alarms are that just one module is not enough to monitor all the potential fire prone places. The only way to prevent fire is to cautious at the time. Even if they are installed in every nook and corner, it just is not sufficient for an efficient output consistently. As the number of smoke sensor requirement increase the cost will also increase to its multiple. The proposed system can produce consistent and highly accurate alerts within seconds of accident of the fire. It reduces cost because only one software is enough to power the entire network of surveillance. Research is active on this field by data scientists and machine learning researchers. The real challenge is to minimize the error in detection of fire and sending alerts at the right time.The idea of this research is to fabricate a system through IoT sensors, which is arbitrarily spread in the forest and to make a self-sorted out powerful system between the sensors to cover all the enormous territories in the forest that will used to maintain a strategic distance from the fire harm whenever. The capacity of the sensor is to identify fire in the inclusion region between the time intermission of each 5-10 minutes. At the point when the fire is recognized the entirety of the sensor in the region will be dynamic and order to stop the normal assignment. The concept is to build early fire detector using Arduino which is connected with different IoT sensors. Putting all efforts to develop a smarter system by connecting it to a webpage and monitoring the developed system statistics controlled by the Arduino programming. The use of latest technology can help to prevent the catastrophic accidents in forests. The aim is to early detect the fireplace in forest by considering the several factor like smoke, temperature, humidity, flame and based on the data we get from this programming, the forest department will be able to take an appropriate decision and the rescue team will be able to arrive on time at exact location. Consider, if it is a large region and it produces more carbon monoxide than the ordinary vehicle traffic. Surveillance of the danger areas and an early detection of fireplace can appreciably shorten the response time and additionally decrease the practicable injury as nicely as the fee of firefighting. Known rule applies here: 1 minute – 1 cup of water, 2 minutes - 100 liters of water, 10 minutes - 1000 liters of water. The goal is to notice the fireplace as quicker as possible, its actual localization and early notification to the fire devices. When fire starts then the flammable texture may likewise issues fuel to the hearth focal spot. The spot at that point will expand and more extensive. The first phase of start is alluded as "surface fire‟‟ stage. This may feed on abutting bushes and the fire will turn into higher and transforming into "crown fire‟‟. Generally, at this stage the hearth transforms into wild and injury

which end up being extreme that could stay for quite long time while depending on atmosphere conditions and the territory. Forest fire detection using optimized solar–powered ZigBee wireless sensor networks- In this paper, they have developed system for Forest Fire Detection which overcomes the demerits of the Existing technologies of Forest Fire Detection. It can be ensured that the system developed can be implemented on a large scale with its promising results. The system is provided with low-power elements, higher versions of Zigbee, Maximum power point tracking Algorithm is used in order to make the system run for longer periods efficiently. Forest fires are a very serious problem in many countries, and global warming may contribute to make this problem worse. Experts agree that, in order to prevent these tragedies from happening, it is necessary to invest in new technologies and equipment that enable a multifaceted approach. This paper describes a WSN for early detection of forest fires. This network can be easily deployed at areas of special interest or risk. There are two types of nodes from the physical structure point of view: SNs, to collect data from the environment, and CNs, to gather data from the SNs and transmit the information to a Control Centre. The nodes also can be in different functioning modes. This enables a proper and seamless configuration of the network, provides redundancy, and ensures there will be full temporal and geographical coverage in the deployment zone. The information gathered is related not only to early detection purposes but also to environment monitoring to maximize the WSN usage. This environmental data can also be employed to firefighting preventive tasks such as vegetation modelling, microclimate studies, and propagation model parametrization.

Characteristics	Flaming	Smoldering
Emissions	Light gases Particles high in EC	Hydrocarbons, PAH's, mercaptans, partially oxidized gases, particles lower in EC
Flames	visible	not visible
Extent of reaction (combustion efficiency)	Reactions tend to go to completion (90-95%)	Incomplete combustion reactio (60-90%)
O_2 concentrations	>= 15%	>= 5%
Temperature	>300°C (peak of 1800°K)	< 300°C
Combustion efficiency [1]	About 90-95%	About 60-90%

TABLE 1. Comparison of different techniques

In this paper, a forest fire detection algorithm is proposed. The algorithm uses YCbCr color space since it effectively separates luminance from chrominance and is able to separate high temperature fire center pixels because the fire at the high temperature center region is white. The final results show that the proposed system has good detection rates and fewer false alarms, which are the main crucial problems of the most existing algorithms. The presences of fire in video streams are indicated by semantic events. Most of the existing systems can only be used for the videos obtained from stationary cameras and videos obtained from the controlled lightening conditions. These existing automatic fire detection systems cannot be used for video

streams obtained from mobile phones or any hand held devices. It was decried as a global tragedy. Lit by farmers, the fires raged through villages, destroyed ecosystems and pumped climate-warming pollution into the atmosphere.

2.2 References

• V.Parthipan, D.Dhanasekaran," Preventing and Monitoring of Framework for Forest Fire Detection and Data Analysis Using Internet of Things (IoT)", IJEAT, ISSN: 2249 – 8958, Volume-8, Issue-3S February 2019.

• Shijo Joseph, K. Anitha, M.S.R. Murthy," Forest fire in India: a review of the knowledge base",J For Res(2009) 14:127-134.

• Ranjth E, Padmabalaji D, Sibisubramanian S, Radhika S," An IOT Based Forest Fire Detection and Prevention System using Raspberry PI 3", IRJET, eISSN: 2395- 0056, p-ISSN: 2395-0072, Volume: 06 Issue: 03, Mar 2019.

• M. Trinath Basu, Ragipati Karthik, J. Mahitha, V. Lokesh Reddy,"IoT Based Forest Fire Detection System", IJET, 7(2.7) (2018) 124-126.

• T. Saikumar, P. Sriramya, „"IoT Enabled Forest Fire Detection and Alerting the Authorities", IJRTE, ISSN: 2277-3878, Volume-7, Issue-6S4, April 2019. 6. Wolfgang KRÜLL, Robert TOBERA, Ingolf WILLMS, , Helmut ESSEN, Nora von WAHL," Early forest fire detection and verification using optical smoke, gas and microwave sensors", / Procedia Engineering 45 (2012) 584 – 594.

• Stanmir Zivanovic, Darko Zigar, Dejan Krstic, "The Role of Early Detection of Forest Fire in Environmental Protection", Safety Engineering, Vol 3, N2 (2013) 93-97. • Antonio Molina-Pico, David Cuesta-Frau, Alvaro Araujo, Javier Alejandre, Alba Rozas,"Forest Monitoring and Wildland Early Fire Detection by a Hierarchical Wireless Sensor Network", Volume 2016 |Article ID 8325845 | 8 pages, Source: Hindawi.

• Ahmad A. A. Alkhatib, "A Review on Forest Fire Detection Techniques", Volume 2014, Article ID 597368,12 Pages, Source: Hindawi.

• Yasar Guneri Sahin, Turker Ince,"Early Forest Fire Detection Using RadioAcoustic Sounding System", Sensors 2009, 9(3), 1485-1498, Source: MDPI.

• Jijitha R., Shabin P." A Review on Forest Fire Detection", HBRP Publication Page 1-8 2019, Volume 2 Issue

• R. Chandrasekharan, Ashiq M.I, Dr. V. Prakash," Forest Fire Detection using Temperature Sensors Powered by Tree and Auto Alarming using GSM", IJRSI, Volume II, Issue III, March 2015, ISSN 2321 – 2705.

• U. Arun Ganesh, M. Anand, S. Arun, M. Dinesh, P. Gunaseelan and R. Karthik," Forest Fire Detection Using Optimized Solar – Powered Zigbee Wireless Sensor Networks", IJSER, Volume 4, Issue 6, June-2013, ISSN 22295518.

• D.Vignesh Kirubaharan, A.John Clement Sunder, S.M.Ramesh, P.Dhinakar," Forest Fire Prediction and Alert System Using Wireless Sensor Network", IJARECE,Volume 3, Issue 11, November 2014.

- Vinay Chowdary, Mukul Kumar Gupta,"Automatic Forest Fire Detection and Monitoring Techniques: A Survey", Source: researchgate.net

2.3. Problem Statement Definition

Forest fires lead to destruction of forest wealth and not only that it also destroys the flora and fauna which causes harm to biodiversity. Forest are great resources and to preserve them is a major challenge. As, they are irreparable damage to the ecosystem, so forest fire detection and prevention is utmost important and best way to tackle this problem. But the forest fire early detection and prevention is another major challenge faced all over the world. Several methods for controlling and monitoring of fires have been proposed. In earlier days, manned observation towers were used but this technique was inefficient and failed. After that satellite and camera imaging technologies were tried but this also proved inefficient and ineffective. For example, cameras were installed at different sites in forest but these provide only line of sight pictures. For a very large areas alert system is required as it is really tedious task to monitor all the images.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

1. An empathy map is a collaborative tool teams can use to gain a deeper insight into their customers. Much like a user persona, an empathy map can represent a group of users, such as a customer segment. The empathy map was originally created by Dave Gray and has gained much popularity within the agile community.
2. An empathy map is an effective visualization template that helps analyze the behavior and emotions of customers and users. Empathy maps not only detect the behaviors but highlight possible mediums for brands to communicate with their customers in a better way
3. Empathy maps can also be used to collect data directly from the users. Used alongside user interviews, survey answers, etc., you can also have a user fill in an empathy map themselves. This often reveals aspects of the user that may have remained unsaid or not thought of.
4. Each of the four quadrants comprises a category that helps us delve into the mind of the user. The four empathy map quadrants look at what the user says, thinks, feels, and does.

Think and Feel?

Detection of forest fire and smoke in wildland areas is done through remote sensing-based methods such as satellites, high-resolution static cameras fixed on the ground

Hear?

These events are mainly caused by the actions of humans, but different nature and environmental phenomena

See?

WILDFIRE
MAGAZINE
WILDLAND FIRE
FIRE ECOLOGY
WORLD ON FIRE

Say and Do?

To control the encroachment or development of undesirable plants and encourage desirable food plants

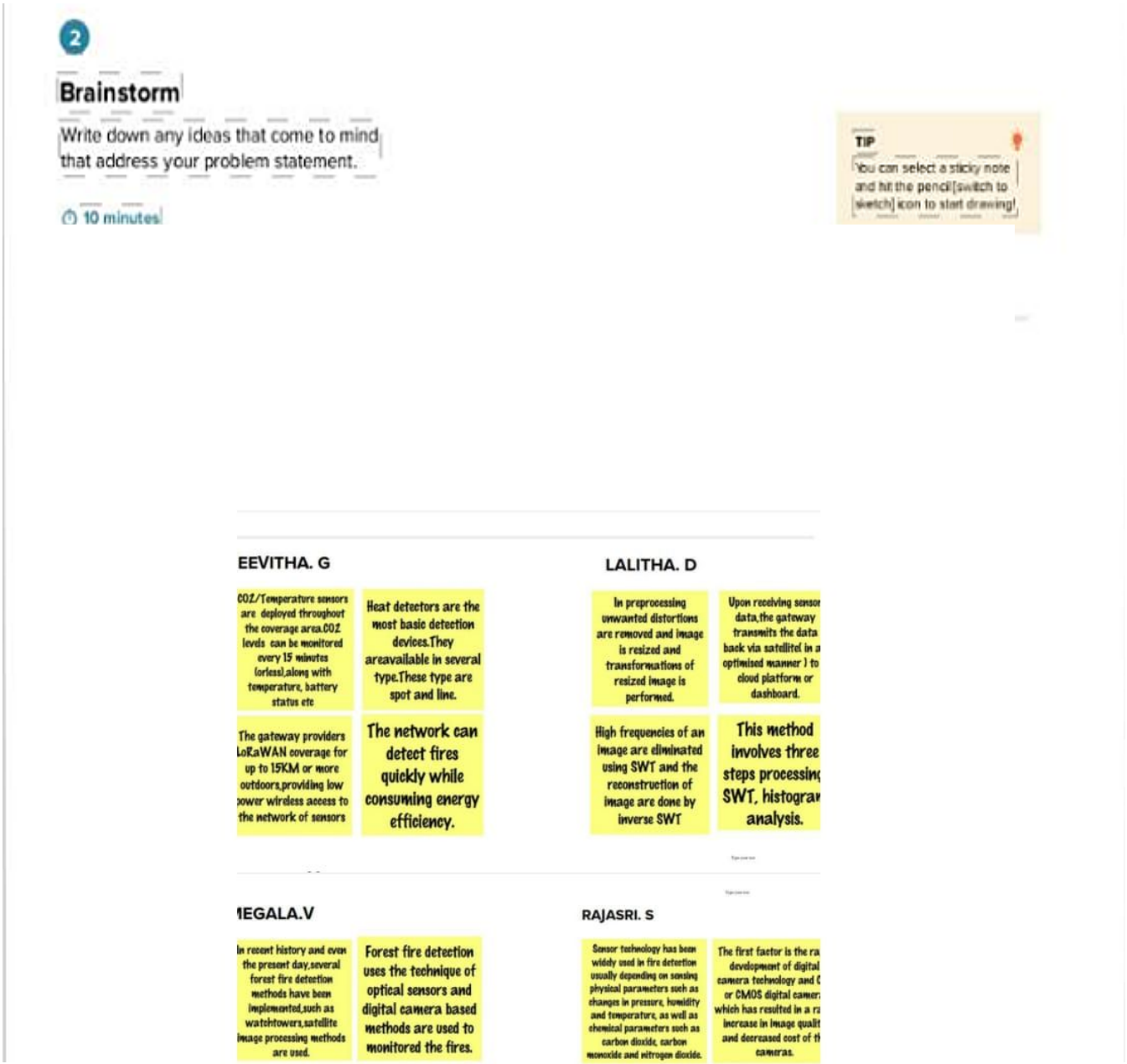
Pains

Pyrophobia is an intense fear of fire
It can cause anxiety, panic attacks or an avoidance of any situation with the potential for fire

Gains

Recycle nutrients bound up in litter.
Reduce competition,
allowing existing trees to grow larger.

3.2 Ideation & Brainstorming



3.3 Proposed Solution

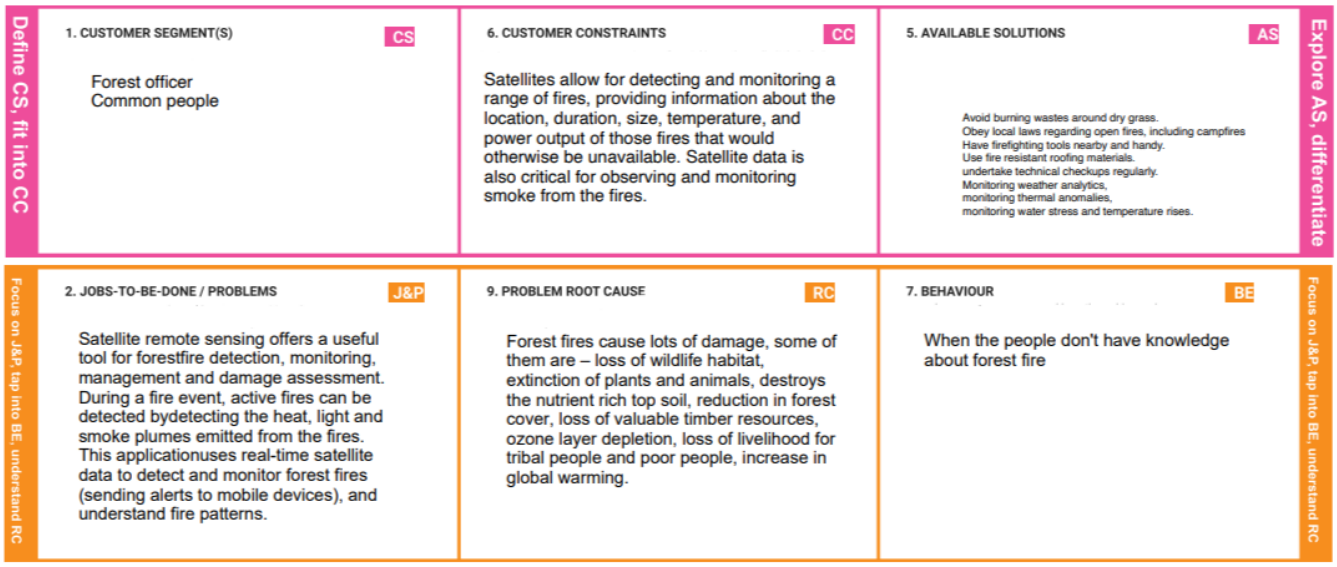
S.No	Parameter	Description
1.	**Problem Statement (Problem to be solved)**	AI based Emerging methods for early detection of forest fires
2.	**Idea/Solution Description**	A solution is needed that detects fires early by detecting smoke, hydrogen and other gases released by pyrolysis in the early stages of a wildfire, buying firefighters valuable time to extinguish the fire before it spreads out of control. Sensing solutions from Bosch Sensortec can help to reduce wildfires.
3.	**Novelty/ Uniqueness**	Remote sensing Machine learning Wildfire prediction Data mining using Artificial intelligence
4.	**Social Impact/ / Customer Satisfaction**	The most important factors in the fight against the forest fires include the earliest possible detection of the fire event , the proper categorisation of the fire and fast response from the fire services . Several different types of forest fires are known , including ground fires , surface fires and crown / tree fires . Each of these types of forest fires is specific and the proper counteractions against it must be considered and implemented to successfully fight it . Over the years the detection of forest fires has been conducted in different ways , ranging from the use of forest outposts to fully automated solutions .
5.	**Business Model (Revenue Model)**	The annual losses from forest fires in India for the entire country have been moderately estimated at Rs 440 crores (US$ 107

3.4 Problem Solution fit

Project Title: EMERGING METHODS FOR EARLY FOREST FIRE DETECTION

Project Design Phase-I - Solution Fit Template

team id:PNT2022TMID08407



4.REQUIREMENT ANALYSIS

4.1 Functional Requirement

1. The system shall take training sets of fire images and recognize whether there is a fire or the beginning of a fire (smoke) or if there is no fire
2. The system shall send a notification to the admin when it recognizes a fire in the image given
3. The system shall take real inputs of camera images and determine whether the image contains a fire or not
4. The system shall be able to take images with a variety of sizes and convert it to one fixed image to be used throughout the application.
5. The system shall run as a service on either a Windows or Linux operating system. 6. In the event that the computer on which the system is running shuts down, the system service should start automatically when the computer restarts

4.2 Non Functional Requirement

1. The system shall provide following facility that will allow web pages that the user is permitted to access. The system must support the following facility:
   a. Send alert message
   b. Customer data management

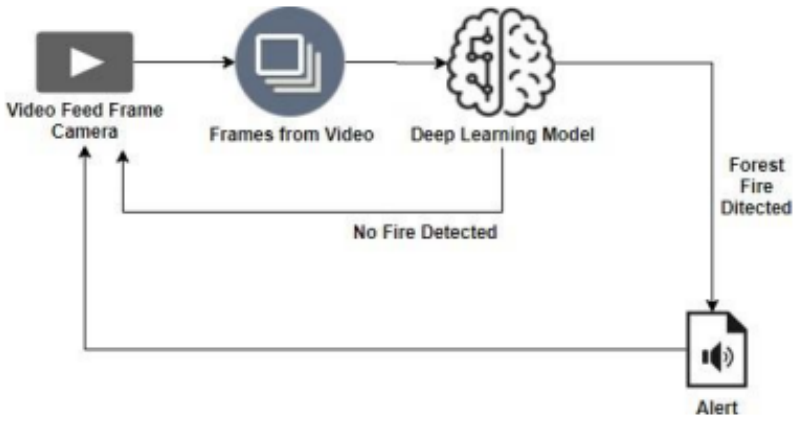
2. The system shall allow the user's status to be stored for the next time he returns to the web site. This will save the user x minutes per visit by not having to reenter already supplied data.

3. The system shall provide information about event log of forest.

5. PROJECT DESIGN

5.1 Data Flow Diagrams

It is difficult to predict and detect Forest Fire in a sparsely populated forest area. it is more difficult if the prediction is done using ground-based methods like Camera or Video-Based approach. Satellites can be an important source of data prior to and also during the Fire due to its reliability and efficiency. The various real-time forest fire detection and prediction approaches, with the goal of informing the local fire authorities

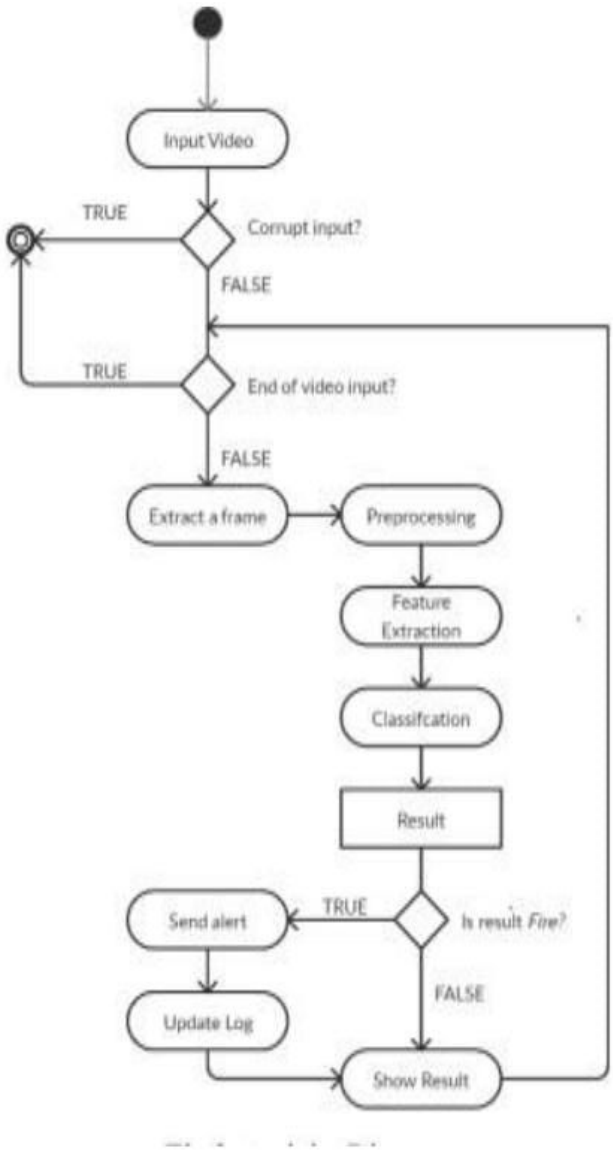


Architecture Diagram

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system.

A neat and clear DFD can depict the right amount of the system requirement graphically.

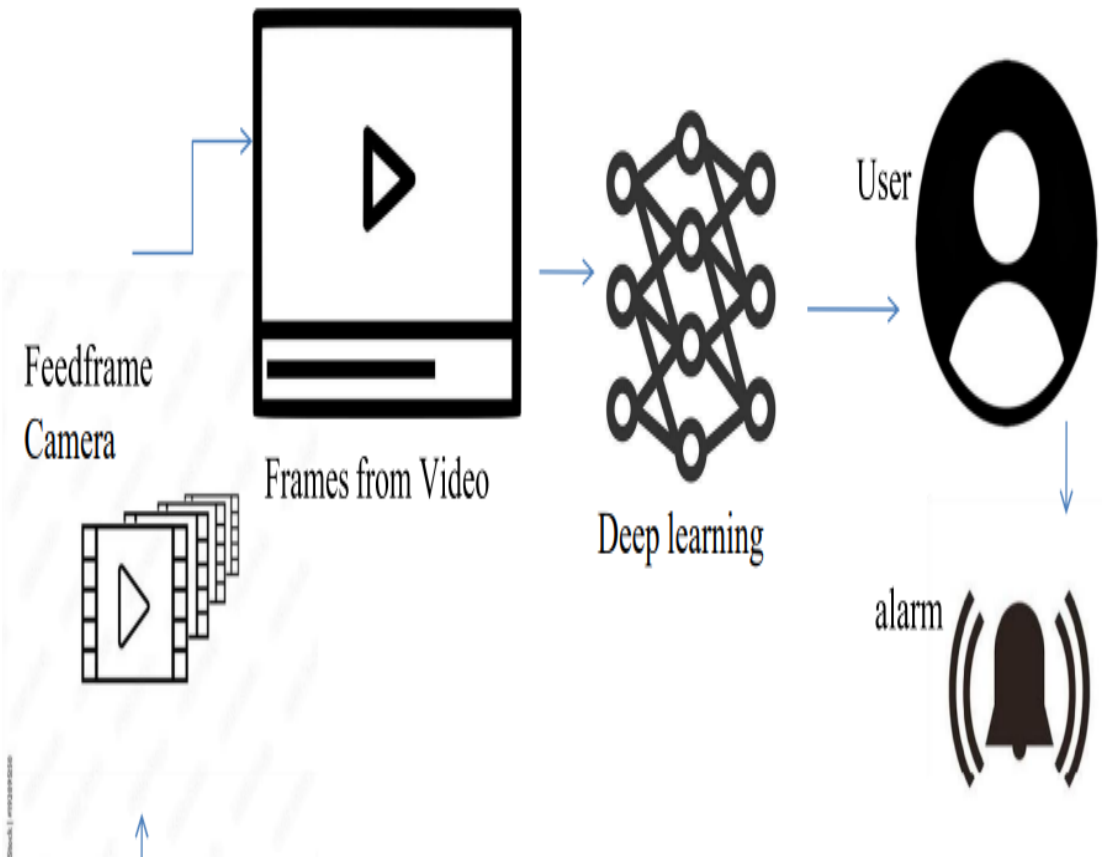
It shows how data enters and leaves the system, what changes the information, and where data is stored.



Dataflow diagram

5.2 Solution & Technical Architecture

TEAM ID:PNT2022TMID08407



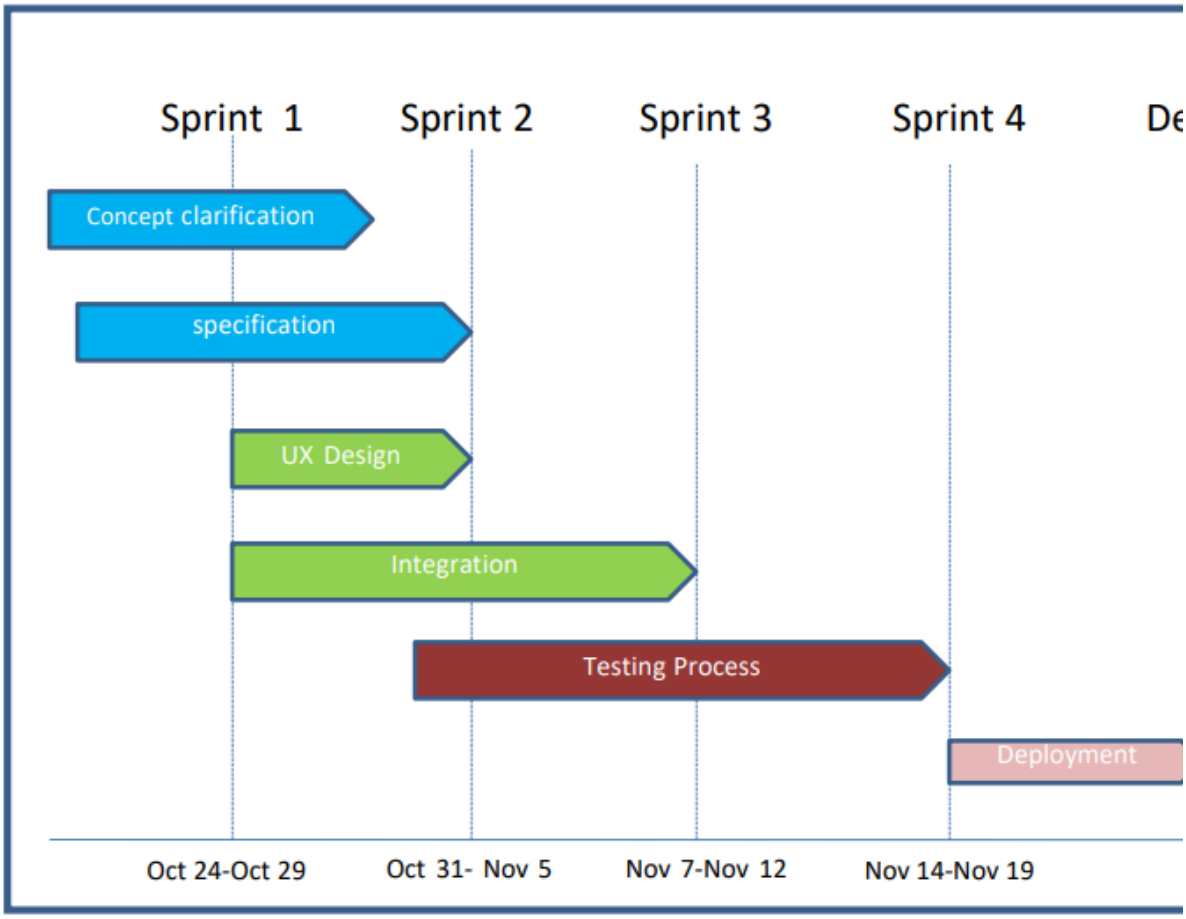
Technical Architecture

5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story I Task	Acceptance criteria	Priority	Release
Environmentalist	Collect the data	USN-1	As an Environmentalist.it is necessary to collect the data of the forest which includes temperature,humidity,wind and rain of the forest	It is necessary to collect the right data else the prediction may become wrona	High	Sprint-1
		USN-2	Identify algorithms that can be used for prediction	To collect the algorithm to identify the accuracy level of each algorithms	Medium	Sprint-2
	Implement Algorithm	USN-3	Identify the accuracy of each algorithms	Accuracy of each algorithm-calculated so that it is easy to obtain the most accurate output	High	Sprint-2
		USN-4	Evaluate the Dataset	Data is evaluated before processing	Medium	Sprint-1
	Evaluate Accuracy of Algorithm	USN-5	Identify accuracy,precision,recall of each algorithms	These values are important for obtaining the riaht output	High	Sprint-3

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation



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	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

6.3 Reports from JIRA

JIRA has categorized reports in four levels, which are –

- Agile
- Issue Analysis
- Forecast & Management
- Others

VELOCITY: SPRINT - 1

Sprint duration = 5 days

Velocity of team = 20 points

Average Velocity (AV) = $\frac{\text{Velocity}}{\text{Sprint duration}}$

AV = 20/5 = 4

Average Velocity = 4

VELOCITY: Sprint 1 - 4

Sprint duration = 20 days

Velocity of team = 80 points

Average Velocity (AV) = $\frac{\text{Velocity}}{\text{Sprint duration}}$

AV = 80/20 = 4

Total Average Velocity = 4

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

7.1 Feature 1

```
import warnings
warnings.filterwarnings("ignore")
import numpy as np
import sys
import tensorflow as tf
from distutils.version import StrictVersion
from collections import defaultdict
from object_detection.utils import ops as utils_ops

##import serial
##ser = serial.Serial(port = "COM7", baudrate = '9600')

# This is needed since the notebook is stored in the object_detection folder.
sys.path.append("..")


if StrictVersion(tf.__version__) < StrictVersion('1.9.0'):
  raise ImportError('Please upgrade your TensorFlow installation to v1.9.* or later!')


from utils import label_map_util
from utils import visualization_utils as vis_util

MODEL_NAME = 'inference_graph'
PATH_TO_FROZEN_GRAPH = MODEL_NAME + '/frozen_inference_graph.pb'
PATH_TO_LABELS = 'training/labelmap.pbtxt'

detection_graph = tf.Graph()
with detection_graph.as_default():
  od_graph_def = tf.GraphDef()
  with tf.gfile.GFile(PATH_TO_FROZEN_GRAPH, 'rb') as fid:
    serialized_graph = fid.read()
    od_graph_def.ParseFromString(serialized_graph)
    tf.import_graph_def(od_graph_def, name='')
```

```
category_index = label_map_util.create_category_index_from_labelmap(PATH_TO_LABELS,
use_display_name=True)

def run_inference_for_single_image(image, graph):
    if 'detection_masks' in tensor_dict:
        # The following processing is only for single image
        detection_boxes = tf.squeeze(tensor_dict['detection_boxes'], [0])
        detection_masks = tf.squeeze(tensor_dict['detection_masks'], [0])
        # Reframe is required to translate mask from box coordinates to image coordinates and fit
the image size.
        real_num_detection = tf.cast(tensor_dict['num_detections'][0], tf.int32)
        detection_boxes = tf.slice(detection_boxes, [0, 0], [real_num_detection, -1])
        detection_masks = tf.slice(detection_masks, [0, 0, 0], [real_num_detection, -1, -1])
        detection_masks_reframed = utils_ops.reframe_box_masks_to_image_masks(
            detection_masks, detection_boxes, image.shape[0], image.shape[1])
        detection_masks_reframed = tf.cast(
            tf.greater(detection_masks_reframed, 0.5), tf.uint8)
        # Follow the convention by adding back the batch dimension
        tensor_dict['detection_masks'] = tf.expand_dims(
            detection_masks_reframed, 0)
    image_tensor = tf.get_default_graph().get_tensor_by_name('image_tensor:0')

    # Run inference
    output_dict = sess.run(tensor_dict,
                           feed_dict={image_tensor: np.expand_dims(image, 0)})

    # all outputs are float32 numpy arrays, so convert types as appropriate
    output_dict['num_detections'] = int(output_dict['num_detections'][0])
    output_dict['detection_classes'] = output_dict[
        'detection_classes'][0].astype(np.uint8)

    output_dict['detection_boxes'] = output_dict['detection_boxes'][0]
    output_dict['detection_scores'] = output_dict['detection_scores'][0]
    if 'detection_masks' in output_dict:
        output_dict['detection_masks'] = output_dict['detection_masks'][0]
    if output_dict['detection_classes'][0] == 1 and  output_dict['detection_scores'][0] > 0.70:
      print('FIRE')
    winsound.Beep(frequency, duration)

  if output_dict['detection_classes'][0] == 2 and  output_dict['detection_scores'][0] > 0.70:
      print('FIRE')
```

```
 winsound.Beep(frequency, duration)

  if ((output_dict['detection_classes'][0] == 1 or output_dict['detection_scores'][0] < 0.70) and
(output_dict['detection_classes'][0] == 2 or  output_dict['detection_scores'][0] < 0.70)):
     print('No Fire')

   return output_dict
```

7.2 Feature 2

```
import cv2
cap = cv2.VideoCapture(0)

try:
   with detection_graph.as_default():
      with tf.Session() as sess:
            # Get handles to input and output tensors
            ops = tf.get_default_graph().get_operations()
            all_tensor_names = {output.name for op in ops for output in op.outputs}
            tensor_dict = {}
            for key in [
              'num_detections', 'detection_boxes', 'detection_scores',
              'detection_classes', 'detection_masks'
            ]:
               tensor_name = key + ':0'
               if tensor_name in all_tensor_names:
                  tensor_dict[key] = tf.get_default_graph().get_tensor_by_name(
                 tensor_name)

            while True:
               ret, image_np = cap.read()
               # Expand dimensions since the model expects images to have shape: [1, None,
None, 3]
               image_np_expanded = np.expand_dims(image_np, axis=0)
               # Actual detection.
               output_dict = run_inference_for_single_image(image_np, detection_graph)
               # Visualization of the results of a detection.
               vis_util.visualize_boxes_and_labels_on_image_array(
                  image_np,
                  output_dict['detection_boxes'],
                  output_dict['detection_classes'],
                  output_dict['detection_scores'],
```

```
                category_index,
                instance_masks=output_dict.get('detection_masks'),
                use_normalized_coordinates=True,
                line_thickness=8)
            cv2.imshow('Frame', cv2.resize(image_np,(800,600)))
            if cv2.waitKey(1) == ord('q'):
                cap.release()
                cv2.destroyAllWindows()
                break
except Exception as e:
    print(e)
    cap.release()
```

8.TESTING

8.1 Test Cases

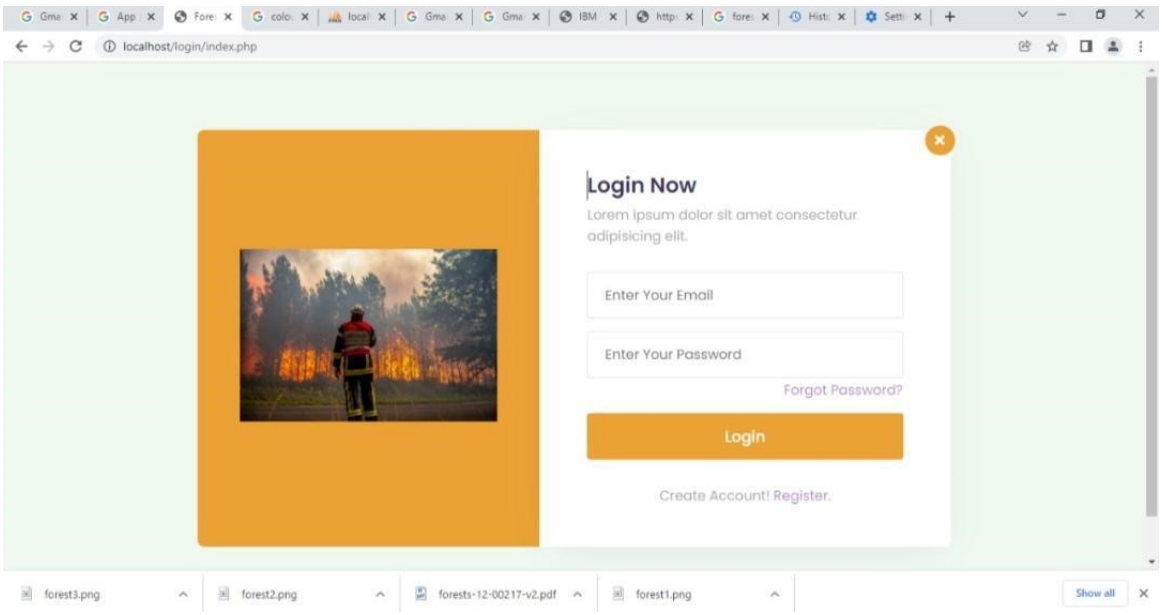


Fig 1. Login page

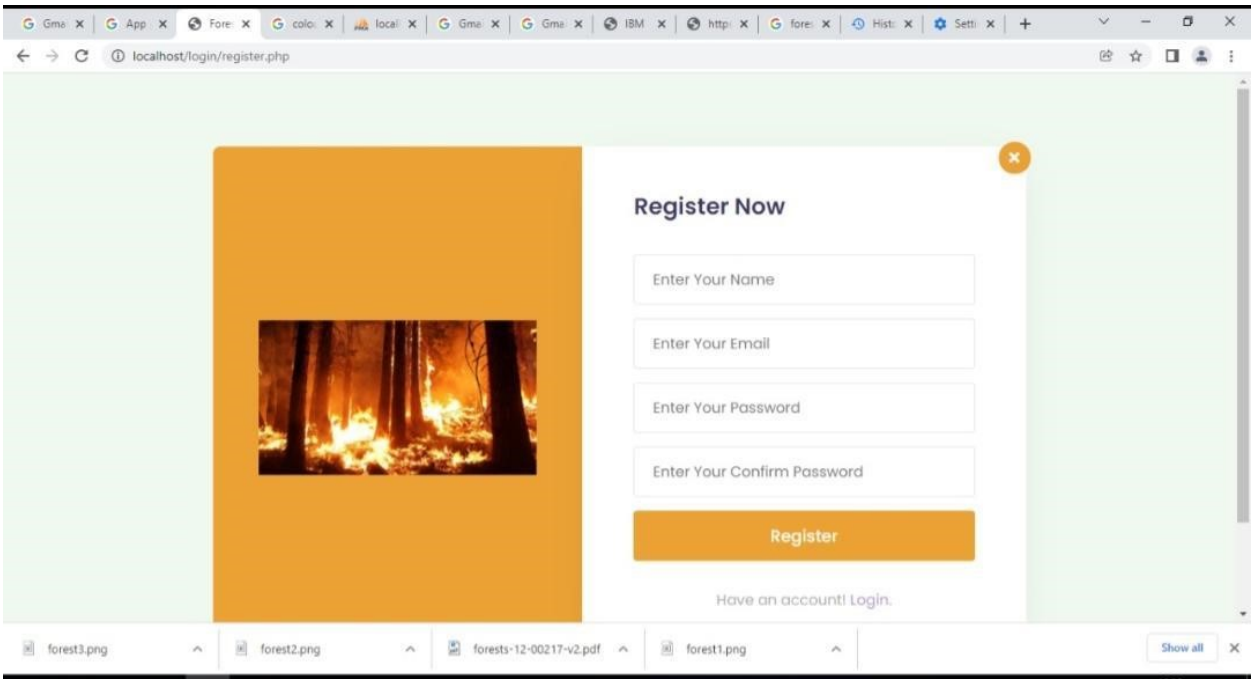


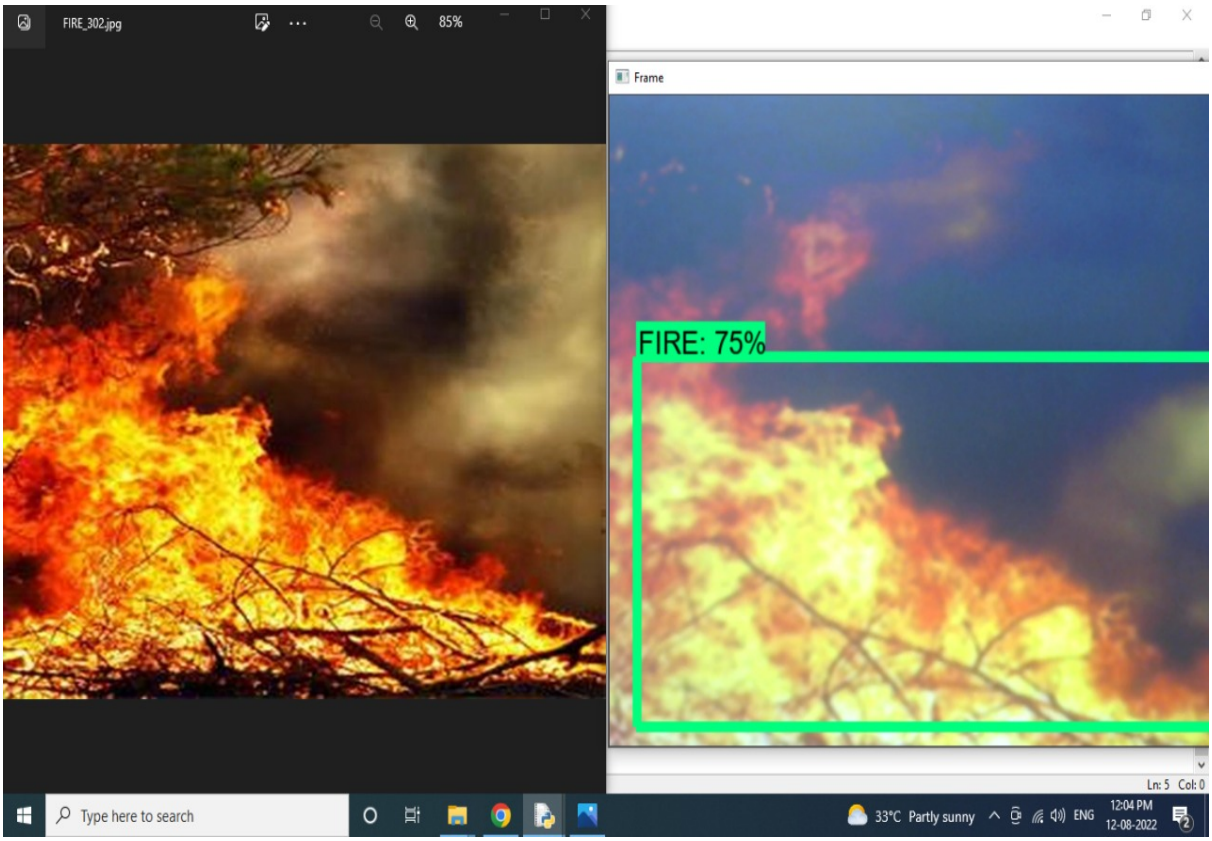
Fig 2. Register Page

8.2 User Acceptance Testing

1. This sort of testing is carried out by users, clients, or other authorised bodies to identify the requirements and operational procedures of an application or piece of software.
2. The most crucial stage of testing is acceptance testing since it determines whether or not the customer will accept the application or programmer.
3. It could entail the application's U.I., performance, usability, and usefulness. It is also referred to as end-user testing, operational acceptance testing, and user acceptance testing (UAT).

9. RESULT

9.1 Performance Metrics



Value obtained from three sensor, if any Infrared ray detected, it gives output as IR detected, Sensor activated! Similarly, if there is any temperature change it will show Abnormal temperature and its intensity. For any smoke detection it output as Smoke detected and sensor value.Above image is result obtained from the trained ML model showing count for damaged and intact homes.

10. ADVANTAGES & DISADVANTAGES

Advantages:

1. It refreshes the habitat zones: Fire clears out plants and trees to make more natural resources available to the habitat. Fewer trees mean more water becomes available for the remaining plants and animals that call the area their home. New grass and shrubs are food sources for a number of animals as well. A ground cover that comes back after a fire becomes a new micro-habitat. Everything is refreshed with a fire.

2. Low-intensity fires don't usually harm trees: The bark of a tree is like an armored shell against fire, pests, and other things that could damage them. Most forest fires burn at low-temperature levels when conditions are optimal and this causes minimal damage to the trees of the forest when it occurs. The end result is a clearing of the ground floor of the forest while the trees are able to continue standing majestically.

3. Decreases the Wastes on Forests: Forests have a lot of waste that ends up building up over time and these wastes can help create wildfires. If a large wildfire breaks out it might take

weeks to control it and the damage it can cause is just too extensive to understand for us. Waste such as dead leaves on the ground can be pretty useful for wildfires to feed on and small forest fires just deal with these wastes properly without going out of control.

Disadvantages:

1. A forest fire sets up the potential for soil erosion to occur: Forest fires clear the underbrush away and encourage new growth, but there is a period of time between the fire and the new growth where the forest is vulnerable.

2. Forest fires always bring death in some form: Maybe it's just the weak plants of the forest that are killed during a fire, but there is always some sort of death that happens when a fire occurs. Sometimes it is the firefighters who are tasked with stopping the fire. It could be animals or pets.

3. Uncontrolled fires can cause localized air pollution: Despite the amount of global development that has occurred, there are many forests that are difficult or nearly impossible to reach. Fires in these areas are left to burn in an uncontrolled fashion and this creates air pollution which can affect the local environment and make it difficult to breathe.

11. CONCLUSION

In our project we propose a fire detection algorithm which is free from sensors as ordinary fire detection systems contain. The objective of this project was to create a system which would be able to detect fire as early as possible from a live video feed.

System is expected to detect fire while it is still small and has not grown to mammoth proportions. Also, the hardware is minimal and has been already existent in places, thus saving capital. It also saves cost by getting rid of expensive temperature and heat sensors etc.

Based on the results produced, the system has proven to be effective at detecting fire. This system is an amalgamation of various fire detection algorithms. Less error rate can be implemented at a large scale like in big factories, houses.

The system can be made weatherproof Smoke detection along with fire detection can be added as a feature of System Optimization and Delay Reduction Ie. Lesser latency may be achieved.

12. FUTURE SCOPE

Future Scope In future, we are planning to install smart water tank system in dense forest where reachability of resources and firefighters is difficult. In addition to that we will be updating the system with more features and reliability. We will also include a high pitch sound system that will keep away the animals from the site of fire.The proposed system can be developed to more advanced system by integrating wireless sensors with CCTV for added protection and precision. The algorithm shows great promise in adapting to various environment.

System can be used to detect forest fires and may be embedded on a drone or any other UAV for surveillance purpose of property. The system have military applications.

13. APPENDIX

Source Code:

```
import warnings
warnings.filterwarnings("ignore")
import numpy as np
import sys
import tensorflow as tf
from distutils.version import StrictVersion
from collections import defaultdict
from object_detection.utils import ops as utils_ops

##import serial
##ser = serial.Serial(port = "COM7", baudrate = '9600')

# This is needed since the notebook is stored in the object_detection folder.
sys.path.append("..")


if StrictVersion(tf.__version__) < StrictVersion('1.9.0'):
  raise ImportError('Please upgrade your TensorFlow installation to v1.9.* or later!')
```

```
from utils import label_map_util
from utils import visualization_utils as vis_util

MODEL_NAME = 'inference_graph'
PATH_TO_FROZEN_GRAPH = MODEL_NAME + '/frozen_inference_graph.pb'
PATH_TO_LABELS = 'training/labelmap.pbtxt'

detection_graph = tf.Graph()
with detection_graph.as_default():
  od_graph_def = tf.GraphDef()
  with tf.gfile.GFile(PATH_TO_FROZEN_GRAPH, 'rb') as fid:
    serialized_graph = fid.read()
    od_graph_def.ParseFromString(serialized_graph)
    tf.import_graph_def(od_graph_def, name='')

category_index =
label_map_util.create_category_index_from_labelmap(PATH_TO_LABELS,
use_display_name=True)

def run_inference_for_single_image(image, graph):
    if 'detection_masks' in tensor_dict:
        # The following processing is only for single image
        detection_boxes = tf.squeeze(tensor_dict['detection_boxes'], [0])
        detection_masks = tf.squeeze(tensor_dict['detection_masks'], [0])
        # Reframe is required to translate mask from box coordinates to image
coordinates and fit the image size.
        real_num_detection = tf.cast(tensor_dict['num_detections'][0], tf.int32)
        detection_boxes = tf.slice(detection_boxes, [0, 0], [real_num_detection,
-1])
        detection_masks = tf.slice(detection_masks, [0, 0, 0],
```

```
[real_num_detection, -1, -1])
      detection_masks_reframed =
utils_ops.reframe_box_masks_to_image_masks(
          detection_masks, detection_boxes, image.shape[0], image.shape[1])
      detection_masks_reframed = tf.cast(
          tf.greater(detection_masks_reframed, 0.5), tf.uint8)
      # Follow the convention by adding back the batch dimension
      tensor_dict['detection_masks'] = tf.expand_dims(
          detection_masks_reframed, 0)
  image_tensor =
tf.get_default_graph().get_tensor_by_name('image_tensor:0')

  # Run inference
  output_dict = sess.run(tensor_dict,
                         feed_dict={image_tensor: np.expand_dims(image, 0)})

  # all outputs are float32 numpy arrays, so convert types as appropriate
  output_dict['num_detections'] = int(output_dict['num_detections'][0])
  output_dict['detection_classes'] = output_dict[
      'detection_classes'][0].astype(np.uint8)

  output_dict['detection_boxes'] = output_dict['detection_boxes'][0]
  output_dict['detection_scores'] = output_dict['detection_scores'][0]
  if 'detection_masks' in output_dict:
      output_dict['detection_masks'] = output_dict['detection_masks'][0]
  if output_dict['detection_classes'][0] == 1 and
output_dict['detection_scores'][0] > 0.70:
    print('FIRE')

  if output_dict['detection_classes'][0] == 2 and
output_dict['detection_scores'][0] > 0.70:
    print('FIRE')
```

```
   if ((output_dict['detection_classes'][0] == 1 or
output_dict['detection_scores'][0] < 0.70) and
(output_dict['detection_classes'][0] == 2 or
output_dict['detection_scores'][0] < 0.70)):
      print('No Fire')

   return output_dict



import cv2
cap = cv2.VideoCapture(0)

try:
   with detection_graph.as_default():
      with tf.Session() as sess:
            # Get handles to input and output tensors
            ops = tf.get_default_graph().get_operations()
            all_tensor_names = {output.name for op in ops for output in
op.outputs}
            tensor_dict = {}
            for key in [
              'num_detections', 'detection_boxes', 'detection_scores',
              'detection_classes', 'detection_masks'
            ]:
               tensor_name = key + ':0'
               if tensor_name in all_tensor_names:
                  tensor_dict[key] = tf.get_default_graph().get_tensor_by_name(
                 tensor_name)

            while True:
```

```
            ret, image_np = cap.read()
            # Expand dimensions since the model expects images to have
shape: [1, None, None, 3]
            image_np_expanded = np.expand_dims(image_np, axis=0)
            # Actual detection.
            output_dict = run_inference_for_single_image(image_np,
detection_graph)
            # Visualization of the results of a detection.
            vis_util.visualize_boxes_and_labels_on_image_array(
               image_np,
               output_dict['detection_boxes'],
               output_dict['detection_classes'],
               output_dict['detection_scores'],
               category_index,
               instance_masks=output_dict.get('detection_masks'),
               use_normalized_coordinates=True,
               line_thickness=8)
            cv2.imshow('Frame', cv2.resize(image_np,(800,600)))
            if cv2.waitKey(1) == ord('q'):
               cap.release()
               cv2.destroyAllWindows()
               break
except Exception as e:
   print(e)
   cap.release()
```

MODEL MAIN:

```
from __future__ import absolute_import
from __future__ import division
from __future__ import print_function
```

```
from absl import flags

import tensorflow as tf

from object_detection import model_hparams
from object_detection import model_lib

flags.DEFINE_string(
    'model_dir', None, 'Path to output model directory '
    'where event and checkpoint files will be written.')
flags.DEFINE_string('pipeline_config_path', None, 'Path to pipeline config '
                    'file.')
flags.DEFINE_integer('num_train_steps', None, 'Number of train steps.')
flags.DEFINE_boolean('eval_training_data', False,
                     'If training data should be evaluated for this job. Note '
                     'that one call only use this in eval-only mode, and '
                     '`checkpoint_dir` must be supplied.')
flags.DEFINE_integer('sample_1_of_n_eval_examples', 1, 'Will sample one of
'
                     'every n eval input examples, where n is provided.')
flags.DEFINE_integer('sample_1_of_n_eval_on_train_examples', 5, 'Will
sample '
                     'one of every n train input examples for evaluation, '
                     'where n is provided. This is only used if '
                     '`eval_training_data` is True.')
flags.DEFINE_string(
    'hparams_overrides', None, 'Hyperparameter overrides, '
    'represented as a string containing comma-separated '
    'hparam_name=value pairs.')
flags.DEFINE_string(
    'checkpoint_dir', None, 'Path to directory holding a checkpoint.  If '
    '`checkpoint_dir` is provided, this binary operates in eval-only mode, '
```

```
    'writing resulting metrics to `model_dir`.')
flags.DEFINE_boolean(
    'run_once', False, 'If running in eval-only mode, whether to run just '
    'one round of eval vs running continuously (default).'
)
FLAGS = flags.FLAGS



def main(unused_argv):
  flags.mark_flag_as_required('model_dir')
  flags.mark_flag_as_required('pipeline_config_path')
  config = tf.estimator.RunConfig(model_dir=FLAGS.model_dir)

  train_and_eval_dict = model_lib.create_estimator_and_inputs(
      run_config=config,
      hparams=model_hparams.create_hparams(FLAGS.hparams_overrides),
      pipeline_config_path=FLAGS.pipeline_config_path,
      train_steps=FLAGS.num_train_steps,
      sample_1_of_n_eval_examples=FLAGS.sample_1_of_n_eval_examples,
      sample_1_of_n_eval_on_train_examples=(
          FLAGS.sample_1_of_n_eval_on_train_examples))
  estimator = train_and_eval_dict['estimator']
  train_input_fn = train_and_eval_dict['train_input_fn']
  eval_input_fns = train_and_eval_dict['eval_input_fns']
  eval_on_train_input_fn = train_and_eval_dict['eval_on_train_input_fn']
  predict_input_fn = train_and_eval_dict['predict_input_fn']
  train_steps = train_and_eval_dict['train_steps']

  if FLAGS.checkpoint_dir:
    if FLAGS.eval_training_data:
      name = 'training_data'
      input_fn = eval_on_train_input_fn
```

```
    else:
      name = 'validation_data'
      # The first eval input will be evaluated.
      input_fn = eval_input_fns[0]
    if FLAGS.run_once:
      estimator.evaluate(input_fn,
                         num_eval_steps=None,
                         checkpoint_path=tf.train.latest_checkpoint(
                             FLAGS.checkpoint_dir))
    else:
      model_lib.continuous_eval(estimator, FLAGS.checkpoint_dir, input_fn,
                                train_steps, name)
  else:
    train_spec, eval_specs = model_lib.create_train_and_eval_specs(
        train_input_fn,
        eval_input_fns,
        eval_on_train_input_fn,
        predict_input_fn,
        train_steps,
        eval_on_train_data=False)

    # Currently only a single Eval Spec is allowed.
    tf.estimator.train_and_evaluate(estimator, train_spec, eval_specs[0])


if __name__ == '__main__':
  tf.app.run()
```

GITHUB LINK :

https://github.com/IBM-EPBL/IBM-Project-9417-1659003152

DEMO LINK :

https://drive.google.com/file/d/1Js-rpjq6vNTG25Vw8H-jtR6jUsEDmkqK/view?usp=drivesdk