PROJECT REPORT

Emerging Methods for Early Detection of Forest Fires

submitted by

PNT2022TMID49273

NISHANTH D - 922319106016

NITHYA D - 922319106017

VASANTHAKUMAR V - 922319106030

NAVEEN KUMAR S - 922319106015

TABLE OF CONTENTS

CHAPTER 1 INTRODUCTION

1.1 PROJECT OVERVIEW

Machine learning and deep learning play an important role in computer technology and artificial intelligence. With the use of deep learning and machine learning, human effort can be reduced in recognizing, learning, predictions and in many more areas.

Forest fire detection is the ability of computer systems to recognise Fire from various region of forest , such as fire, smoke, and so on. This project aims to let users take advantage of machine learning to reduce manual tasks in Detecting the forest fire.

1.2 PURPOSE

The main aim of our project is detection and monitoring the forest fire To minimize the effect of fire breakout by controlling in its early stage also to protect Domestic by informing about the breakout to the respective forest department as early as possible . We have implemented the IOT technology to achieve our objective.

CHAPTER 2 LITERATURE SURVEY

2.1 EXISTING PROBLEM

Some of the relevant literary works in this field are briefed below:

The one fourth area of Karnataka is covered by forest, the forest and bio-diversity of the India are at the considerable chance and beneath enormous pressure. General causes of forest fire are extreme hot and aired weather, lightning and human carelessness. In order to protect these huge stretches of forest land, there need to be taken early caution measures to control of spreading fire

2.2 REFERENCES

1. **A Review on Early Forest Fire Detection Systems Using Optical Remote Sensing**

- P. Barmpoutis, P. Papaioannou, K. Dimitropoulos, N. Grammalidis
- Environmental Science
- Sensors
- 2020

An overview of the optical remote sensing technologies used in early fire warning systems is presented and an extensive survey on both flame and smoke detection algorithms employed by each technology is provided.

2. Forest Fire Detection System using LoRa Technology

- N. Gaitan, Paula Hojbota
- Environmental Science
- 2020

This paper proposes a system capable of quickly detecting forest fires on long wide distance using LoRa (Long Range) technology based on LoRaWAN (Long Range Wide Area Network) protocol which is capable to connect low power devices distributed on large geographical areas.

3. **Low Cost LoRa based Network for Forest Fire Detection**

- Roberto Vega-Rodríguez, Sandra Sendra, Jaime Lloret, Pablo Romero-Díaz, José Luis GarcíaNavas
- Computer Science, Environmental Science

  2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS)

- 2019

A low cost Long Range (LoRa) based network able to evaluate level of fire risk and the presence of a forest fire and the evaluation algorithm is based on the 3030-30 rule.

4. **A Survey of Machine Learning Algorithms Based Forest Fires Prediction and Detection Systems**

- F. Abid
- Environmental Science, Computer Science
- Fire Technology
- 2020

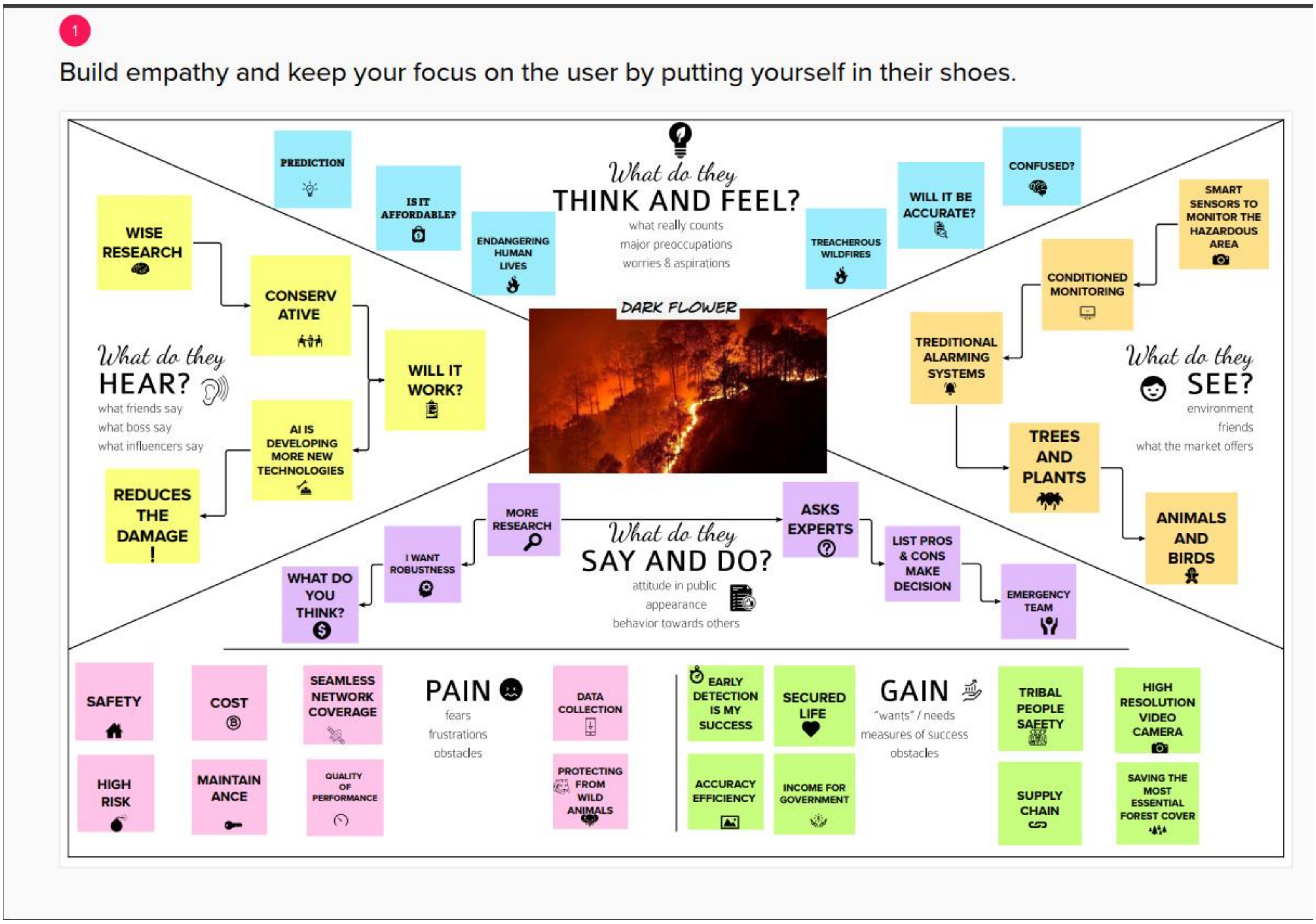
A comprehensive survey of the machine learning algorithms based forest fires prediction and detection systems is presented, highlighting the main issues and outcomes within each study.

2.3 PROBLEM STATEMENT DEFINITION

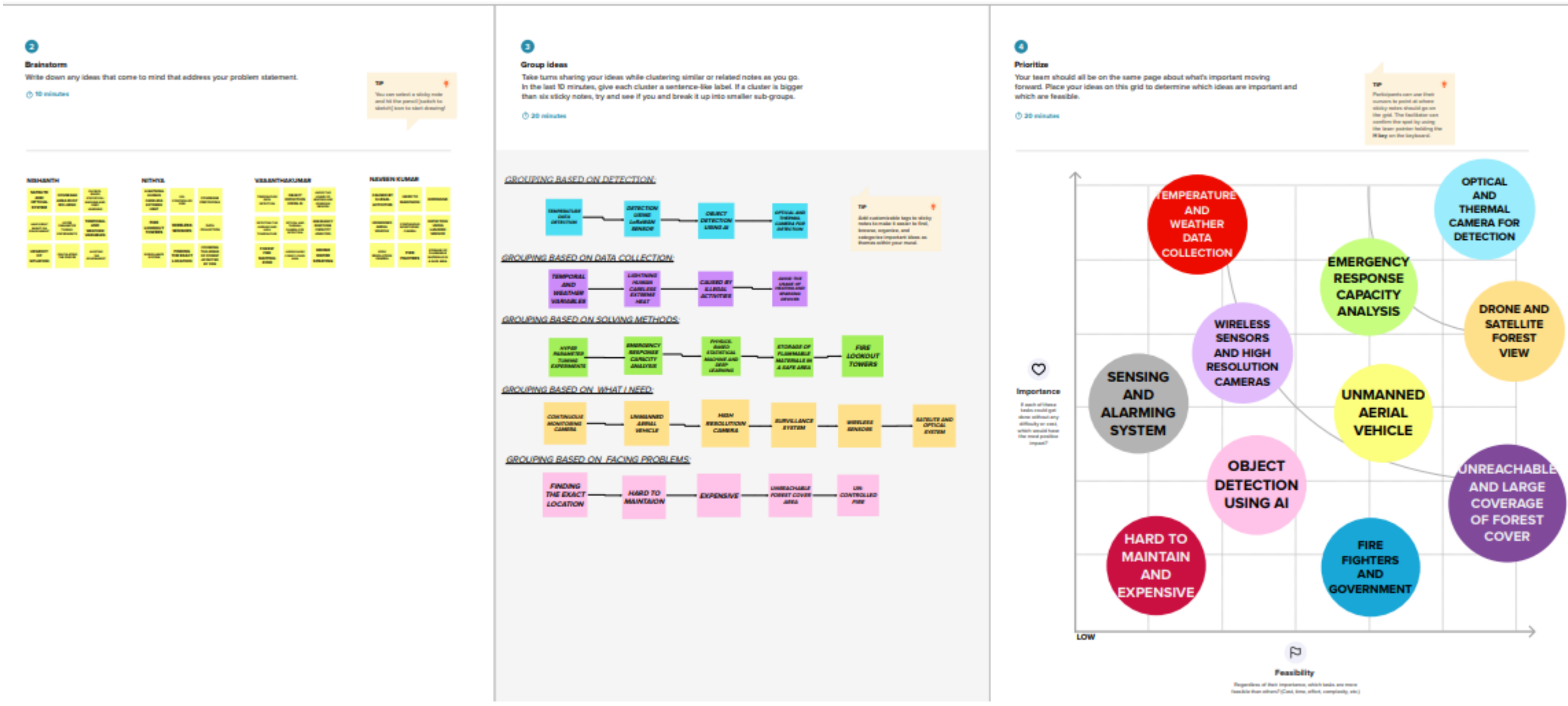
Some people know about the current issues are the most important ones because it is mostly a lot in the news but sometimes other big issues that change our lives are not mentioned in the news because they are issues that can hurt us in the long run or not really important for the modern public. One issue I can tell you about is the forest fires. Sometimes people don 't notice or now about the forest fires until it is talk in the news and it 's mostly because it has done a great damage.

CHAPTER 3 IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS



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 Sensor tech can help to reduce wildfires.
3.	**Novelty / Uniqueness**	Remote sensing Machine learning Wildfireprediction 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 categorization 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 million).
6.	**Scalability of the Solution**	Aerial-based systems gained recently a lot of attention due to the rapid development of UAV technology. Such systems provide a broader and more accurate perception of the fire, even in regions that are inaccessible or considered too dangerous for fire-fighting crews. In addition, UAVs can cover wider areas and are flexible, in the sense that they monitor different areas, as needed

3.4 PROBLEM SOLUTION FIT

CHAPTER 4 REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

FUNCTIONALREQUIREMENTS:

-Following are the functional requirements of the proposed solution

Sn. No	**Functional Requirement (Epic)**	**Sub Requirement (Story / Sub-Task)**
1.	User Registration	Registration through G-mail.
2.	User Confirmation	Confirmation through OTP. Confirmation through mail.
3.	User Login	Can login through credentials.
4.	User Feed	The live update of the forestcover is sent to user if there is any detection of fire
5.	User Profile	The workers profile created to give the forest management live track of the forest.
6.	User Alert	The user receives thequick response through alert sound or Messages,if any fire is detected.
7.	User Application	Along with the forest management team the citizens residing nearby forest can also download the application for alerts.

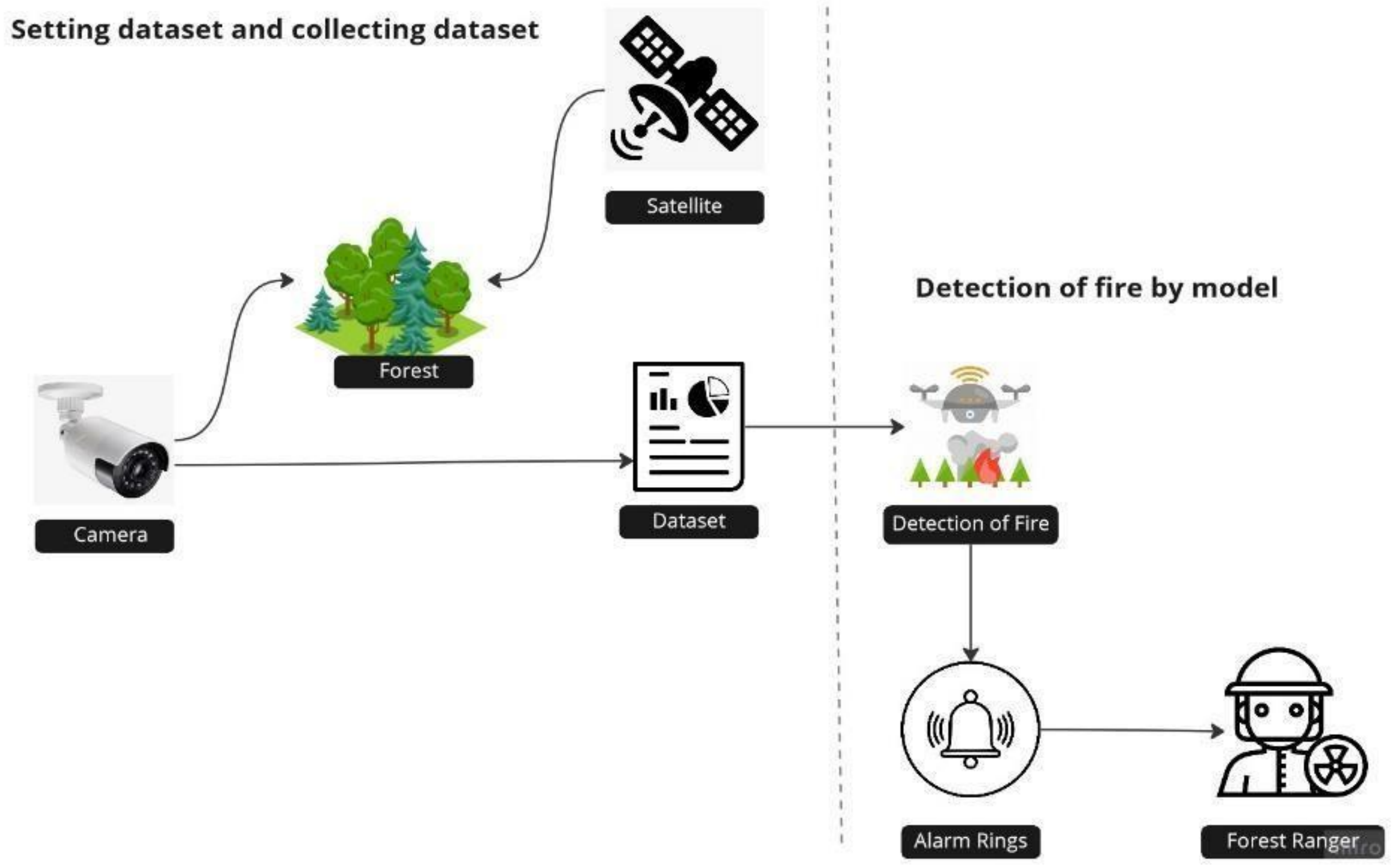
NON-FUNCTIONALREQUIREMENTS:

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

Sn. No.	**Non-Functional Requirement**	**Description**
1.	Usability	Monitoring possible danger areas and early fire detection can greatly reduce the response time and potential damage.
2.	Security	The environment is more secure.
3.	Reliability	The installment of model is safe.
4.	Performance	Model will achieve high accuracy.
5.	Availability	Build model is available all the time.
6.	Scalability	The instant alerts received by the forest team is ensured.

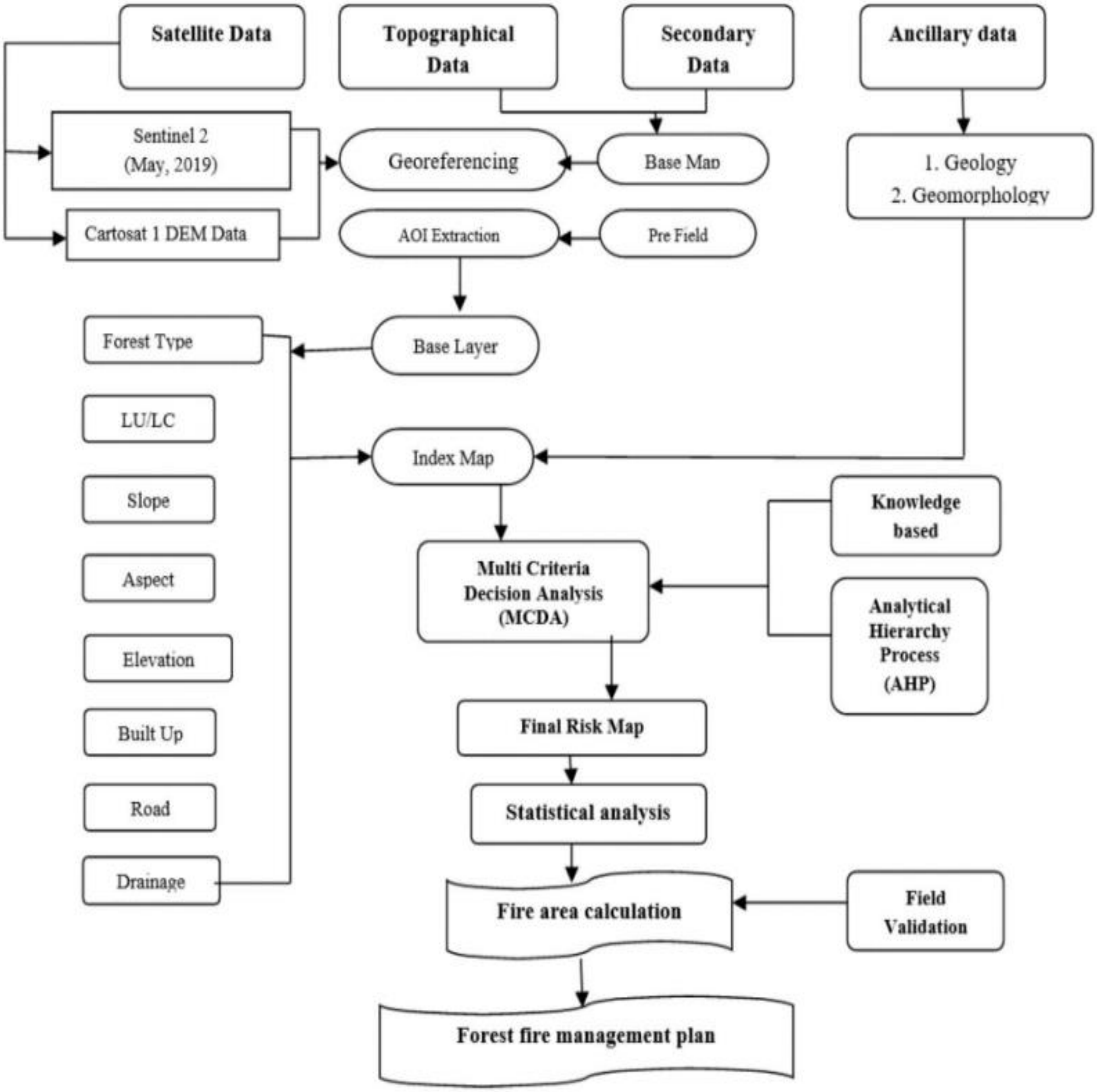
CHAPTER 5 PROJECT DESIGN

Data Flow Diagram

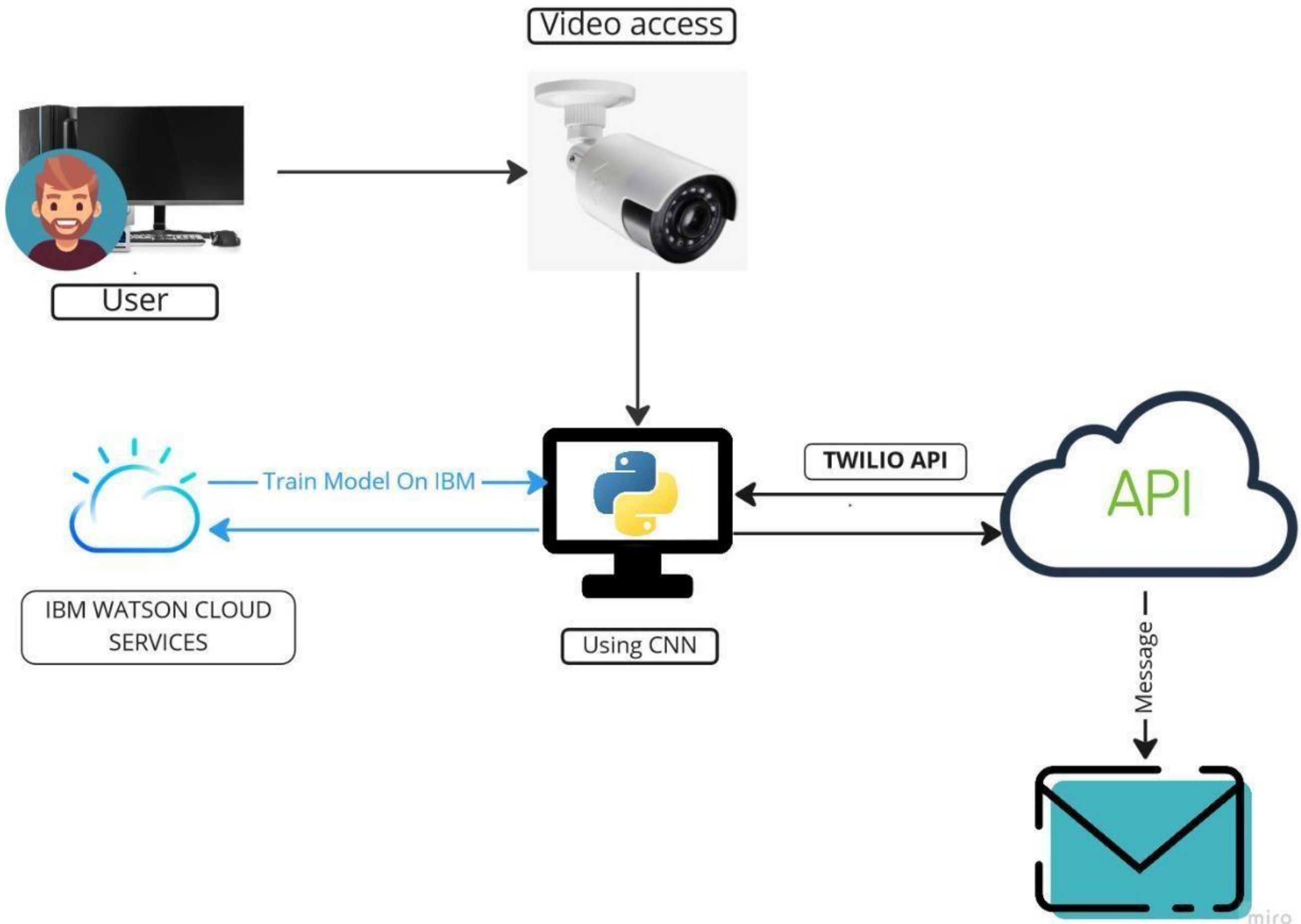


***Data Flow Diagrams*:**

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

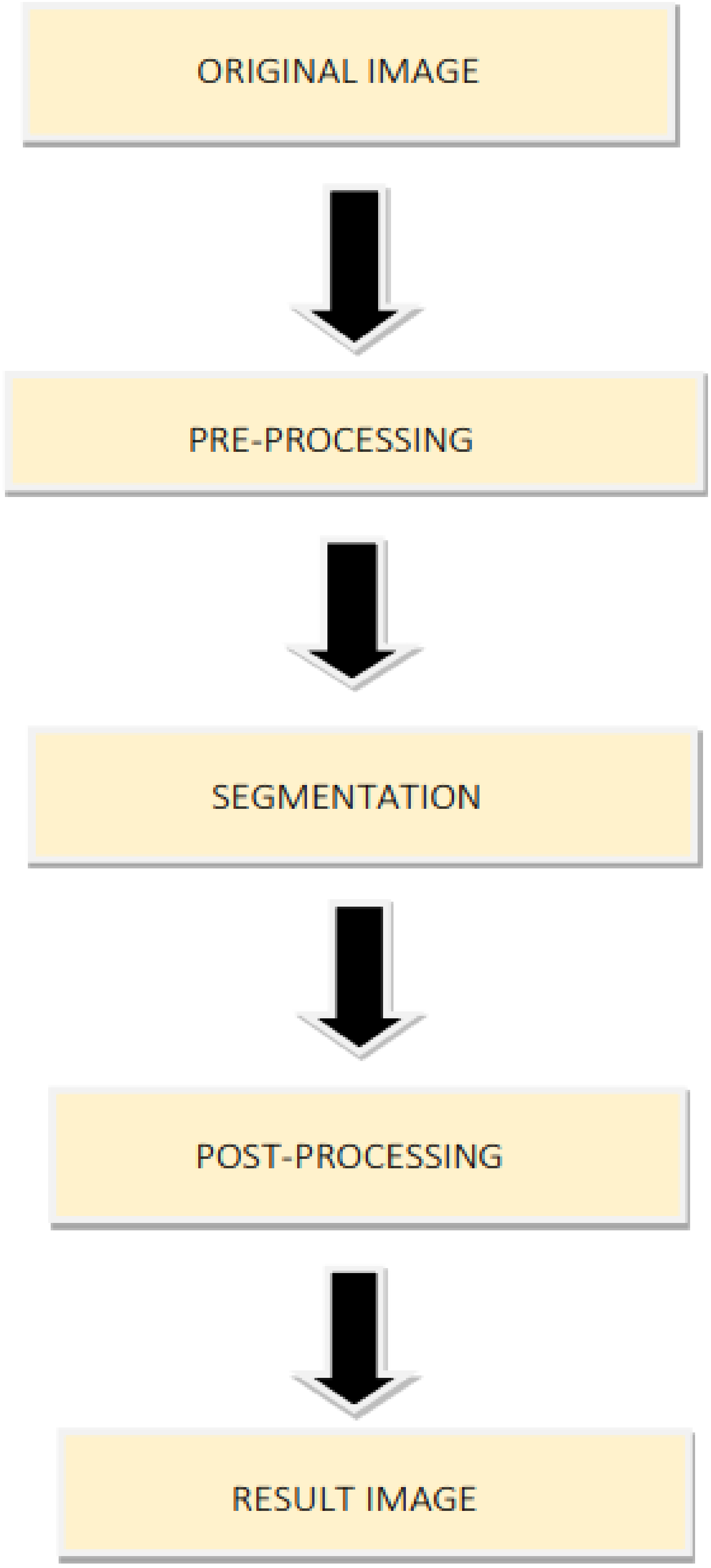


5.2 SOLUTION & TECHNICAL ARCHITECTURE

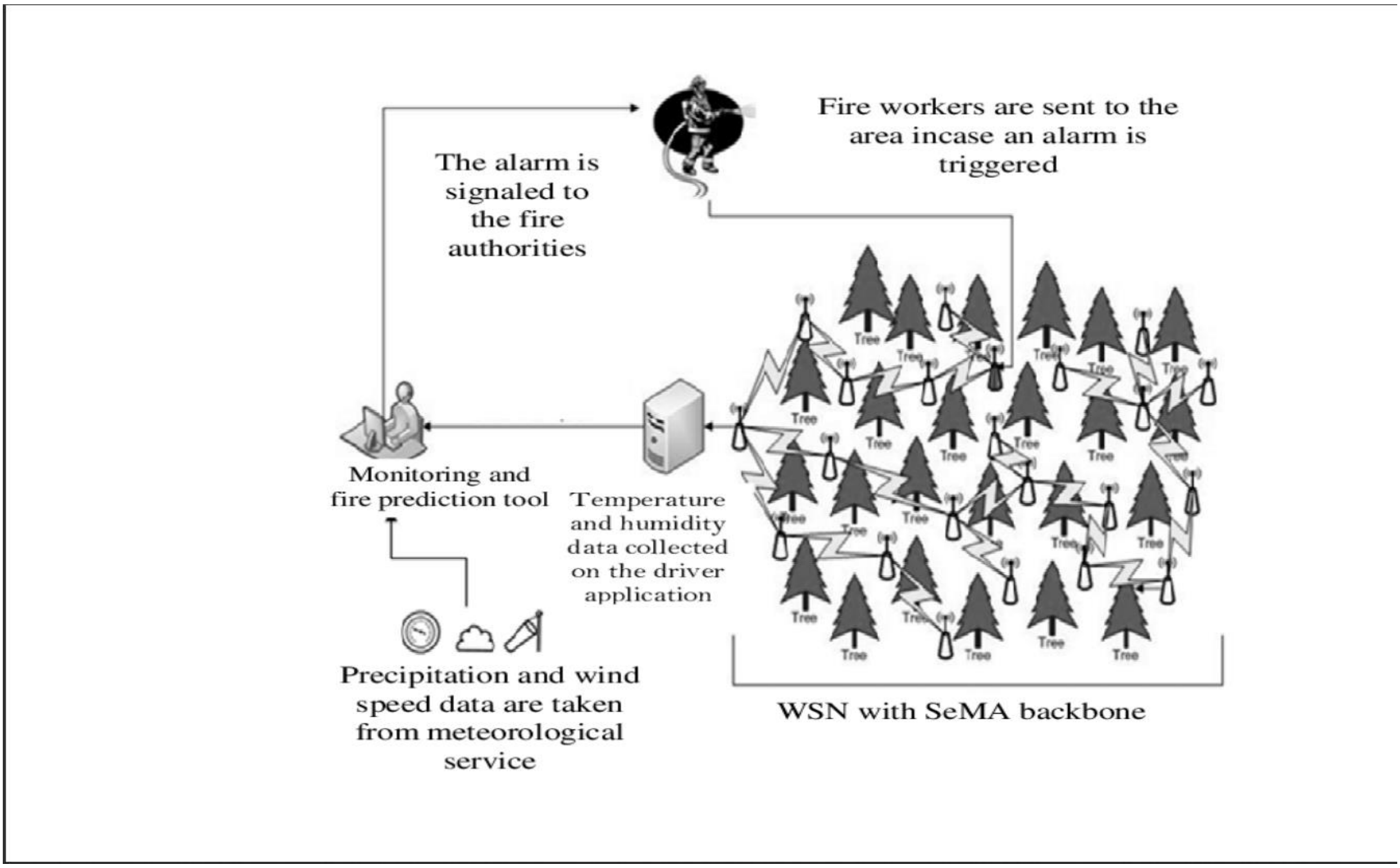


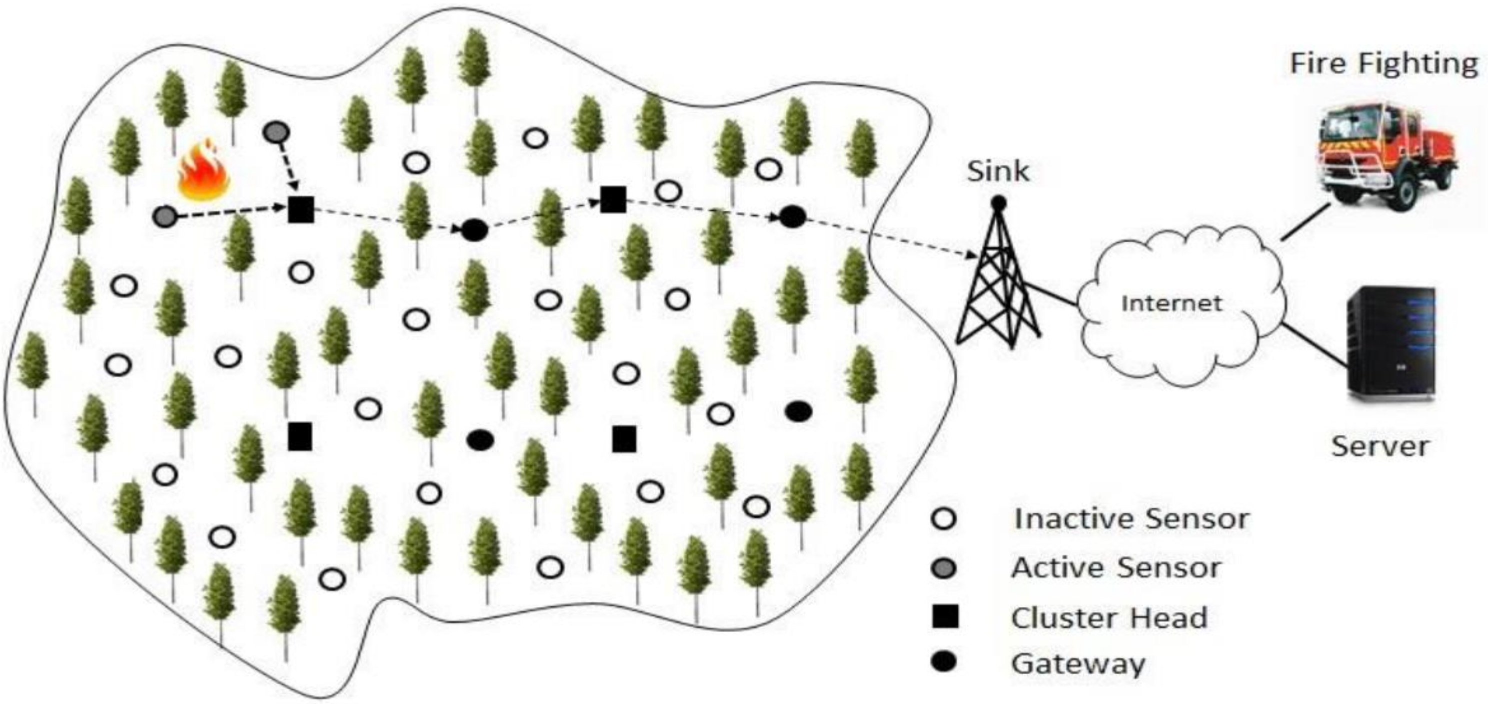
EMERGING METHODS FOR EARLY DETECTION OF FOREST FIRES

SOLUTION ARCHITECTURE:



TECHNICAL ARCHITECTURE





5.3 USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Forest Management Team	Setting up a camera	USN-1	As a user, the forest management team has to survey the forest by adding camera to the fire prone areas.	The live video captured can be monitored	High	Sprint-1
		USN-2	As a user, the forest management team can get video feed which is used for processing	The camera sends video or image to the forest centre	High	Sprint-2
		USN-3	Along with forest team, the NGO can also get access of the video to take some early measurement of forest fires.	They can also get the view of the live monitoring of forest	Low	Sprint-1
Technical Team	Image Classification	USN-4	By using CNN Model, the images captured by the camera is classified accordingly by testing & training the model	The model should be able to identify the difference between fire and a normal smoke	Medium	Sprint-2
	Using Open CV	USN-5	The recorded video is under monitoring continuously to determine the detection of early video	Therefore, by using CNN we can determine the input layer, classify the hidden layers and send warnings through output layer	High	Sprint-2
Alert Team	Dashboard	USN-6	Thus, after successful detection of fire by processing images. This, API sends the alert by buzzing the alarm and sends messages through chatbot	Thus, the immediate response which is required for earlier determination through sending quick responses	High	Sprint-2

Fire Management	Twilio API		They play the most important role to cool the	They take the following measures to	High	Sprint-2
			fire and manage the excess spread of fire further	stop fire from spreading		

User Type	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Acceptance criteria**	**Priority**	**Release**
Media & Nearby Residing People	News, Radio, Alerts,	USN-7	Protecting wildlife, human from the disaster caused	Thus, helping unit should be sent to protect lives	Medium	Sprint-2

CHAPTER 6 PROJECT PLANNING AND SCHEDULING

6.1 SPRINT PLANNING AND ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Import the Required, Collecting the Dataset	USN-1	To analyse the fire prone areas and to set the surveillance camera to collect and observe the region continuously for early detection.	2	High	NISHANTH D NITHYA D
Sprint-2	Training & Testing of model	USN-2	The collected data are categorized on the basis of parameters set to identify. To train the model, CNN is used to test repeatedly by storing the datasets in server.	1	High	VASANTHAKUMAR V NAVEEN KUMAR S
Sprint-3	Model Building, Reviewing the model	USN-3	The main task is to check that the model is efficient to work in real time. Therefore, smallest of error decoded needed to be corrected to avoid future lags	1	Medium	NISHANTH D NITHYA D
Sprint-4	Implementing the model	USN-4	The model after testing all it’s functionalities is been implemented at forest management offices to get quick responses from the model.	2	High	NAVEENKUMAR S VASANTHAKUMAR V
Sprint-4	Connecting it with API	USN-5	The model should connect with API named Twilio, which receives & sends the management with messages.	2	High	NISHANTH D NITHYA D VASANTHAKUMAR V NAVEENKUMAR S

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	15	06 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	10	14 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	5	20 Nov 2022

CHAPTER 7 CODING & SOLUTIONING

```python
import cv2
import numpy as np
from tensorflow.keras.preprocessing import image
from tensorflow.keras.models import load_model
from twilio.rest import Client
from playsound import playsound
from decouple import config


message_sent = False


model = load_model("./model.h5")

video = cv2.VideoCapture("fire.mp4")

name = ["No fire", "Fire Detected"]


def send_message():
    account_sid = config("ACCOUNT_SID")
    auth_token = config("AUTH_TOKEN")

    client = Client(account_sid, auth_token)
    message = client.messages.create(
        body="Forest Fire detected , Stay safe!!!",
        from_=config("FROM"),
        to=config("TO")
    )
    print(message.sid)
    print("Fire Detected")
    print("SMS Sent!")

playsound("./beep.mp3")
```

```python
while True:
    success, frame = video.read()
    cv2.imwrite("image.jpg", frame)
    img = image.load_img("image.jpg", target_size=(128, 128))
    x = image.img_to_array(img)
    x = np.expand_dims(x, axis=0)
    pred = model.predict(x)
    p = int(pred[0][0])
    cv2.putText(frame, str(name[p]), (100, 100), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 0), 1)

    if p == 1:
        if not message_sent:
            send_message()
            message_sent = True
        print("Fire Detected , stay safe!!!")
    else:
        print("No Fire Detected")

    cv2.imshow("Image", frame)

    if cv2.waitKey(1) & 0xFF == ord('x'):
        break

video.release()
cv2.destroyAllWindows()
```

CHAPTER 8

TESTING

8.1 TEST CASES

Test case ID	Feature Type	Component	Test Scenario	Steps To Execute	Test Data	Expected Result	Actual Result	Status	BUG ID	Executed By
OP_RT_001	Functional	Page	Check if user can upload their file	The sensor senses the fire	Sample 1.png	The input image should be uploaded to the application successfully	Working as expected	PASS		NISHANTH D NITHYA D
OP_RT_002	Functional	Page	Check if user cannot upload unsupported files	1) The sensor senses the fire 2)checks with the pre-uploads images	installer.exe	The application should not allow user to select a non image file	User is able to upload any file	FAIL	BUG_HP_002	VASANTHAKUMAR V NAVEEN KUMAR S
OP_RT_003	Functional	Page	Checks whether the page redirects to the result page to the given output	1) The sensor senses the fire 2)checks with the pre-uploaded images 3)checks if there is fire detection	Sample 1.png	The page should redirect to the results page	Working as expected	PASS		VASANTHAKUMAR V NAVEENKUMAR S
MB_RT_001	Functional	Backend	Checks if all the routes are working properly	1) The sensor senses the fire 2)checks with the pre-uploaded images 3)checks if there is fire detection	Sample 1.png	All the routes should properly work	Working as expected	PASS		NISHANTH D VASANTHAKUMAR V
N_DC_001	Functional	Model	Checks whether the model can handle various image sizes	1) Open the page in a specific device 2) Upload the input image 3) Repeat the above steps with different input	Sample 1.png Sample 1 XS.png Sample 1 XL.png	The model should rescale the image and predict the results	Working as expected	PASS		NITHYA D NISHANTH D
N_DC_002	Functional	Model	Check if the model predicts the digit	1) Open the page 2) Select the input images	Sample 1.png	The model should predict the number	Working as expected	PASS		NITHYA D NISHANTH D
N_DC_003	Functional	Model	Check if the model can handle complex input image	1) Open the page 2) Select the input images 3) Check the results	Complex Sample.png	The model should predict the number in the compex image	The model fails to identify the digit since the model is not built to handle such data	FAIL	BUG_M_001	VASANTHAKUMAR V NAVEENKUMAR S
RL_DC_001	Functional	Result Page	Verify the elements	1) Open the page 2) Select the input image 3) Check if all the UI elements are displayed properly	Sample 1.png	The Result page must be displayed properly	Working as expected	PASS		NISHANTH D NAVEENKUMAR S
RL_DC_002	Functional	Result Page	Check if that image is displayed properly	1) Open the page 2) Select the input image 3) Check if the input image are displayed	Sample 1.png	The input image should be displayed properly	The size of the input image exceeds the display container	FAIL	BUG_RP_001	NITHYA D VASANTHAKUMAR V
RL_DC_003	Functional	Result Page	Checks whether the displayed prediction is accurate	1) Open the page 2) Select the input image 3) Check if all the other predictions are displayed	Sample 1.png	The other predictions should be displayed properly	Working as expected	PASS		NITHYA D VASANTHAKUMAR V

8.2 USER ACCEPTANCE TESTING

8.2.1 DEFECT ANALYSIS

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Total
By Design	1	1	1	0	3
Duplicate	0	0	0	0	0
External	0	0	2	0	2
Fixed	3	1	0	1	5
Not Reproduced	0	0	0	1	1
Skipped	1	0	1	0	2
Won't Fix	1	0	0	0	1
Total	6	3	4	3	14

8.2.2 TEST CASE ANALYSIS

Section	Total Cases	Not Tested	Fail	Pass
Client Application	10	0	2	8
Security	3	0	2	2
Performance	2	0	1	1

Exception Reporting	3	0	0	3

CHAPTER 9 RESULTS

9.1 PERFORMANCE METRICS

Locust Test Report

During: 13/12/2022, 7:05:40 AM - 13/12/2022, 7:14:47 AM

Target Host: http://127.0.0.1:5000/

Script: locust.py

Request Statistics

Method	Name	# Requests	# Fails	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	RPS	Failures/s
GET	//	1044	0	14	4	292	1080	2.2	0.0
GET	//predict	1007	0	39649	387	59814	2670	1.8	0.0
	Aggregated	2050	**0**	19464	**4**	**59814**	**1859**	4.0	**0.0**

Response Time Statistics

Method	Name	50%ile (ms)	60%ile (ms)	70%ile (ms)	80%ile (ms)	90%ile (ms)	95%ile (ms)	99%ile (ms)	100%ile (ms)
GET	//	11	12	13	15	20	22	64	290
GET	//predict	44000	46000	47000	48000	50000	52000	55000	60000
	Aggregated	37	**37000**	**43000**	**45000**	49000	**50000**	56000	**60000**


Charts
Total Requests per Second
RPS
Failures/s
0
1
2
3
4
5
6
7:05:43 AM
7:06:38 AM
7:07:33 AM
7:08:28 AM
7:09:23 AM
7:10:18 AM
7:11:13 AM
7:12:08 AM
7:13:03 AM
7:13:58 AM
Response Times (ms)
Median Response Time
95% percentile
0
10,000
20,000
30,000
40,000
50,000
60,000
7:05:43 AM
7:06:38 AM
7:07:33 AM
7:08:28 AM
7:09:23 AM
7:10:18 AM
7:11:13 AM
7:12:08 AM
7:13:03 AM
7:13:58 AM
Number of Users
Users
0
20
40
60
80
100
120
7:05:43 AM
7:06:38 AM
7:07:33 AM
7:08:28 AM
7:09:23 AM
7:10:18 AM
7:11:13 AM
7:12:08 AM
7:13:03 AM
7:13:58 AM

CHAPTER 10 ADVANTAGES & DISADVANTAGES

ADVANTAGES

The proposed system detects the forest fire at a faster rate compared to existing system. It has enhanced data collection feature. The major aspect is that it reduces false alarm and also has accuracy due to various sensors present. It minimizes the human effort as it works automatically. This is very affordable due to which can be easily accessed. The main objective of our project is to receive an alert message through an app to the respective user.

DISADVANTAGES

The electrical interference diminishes the effectiveness of radio receiver. The main drawback is that it has less coverage range areas.

CHAPTER 11 CONCLUSION

This type of system is the first of its kind to ensure no further damage is then to forests when there is fire breakout and immediately a message is sent to the user through the App. Immediate response or early warning to a fire breakout is mostly the only ways to avoid losses and environmental, cultural heritage damages to a great extent. Therefore the most important goals in fire surveillance are quick and reliable detection of fire. It is so much easier to suppress fire while it is in its early stages. Information about progress of fire is highly valuable for managing fire during all its stages. Based on this information the firefighting staff can be guided on target to block fire before it reaches cultural heritage sites and to suppress it quickly by utilizing required firefighting equipment and vehicles. With further research and innovation, this project can be implemented in various forest areas so that we can save our forests and maintain great environment.

CHAPTER 12 FUTURE SCOPE

This project is far from complete and there is a lot of room for improvement. Some of the improvements that can be made to this project are as follows:

Additional pump can be added so that it automatically sends water when there is a fire breakout. Also industrial sensors can be used for better ranging and accuracy.

❾ This project has endless potential and can always be enhanced to become better. Implementing this concept in the real world will benefit several industries and reduce the workload on many workers, enhancing overall work efficiency.

APPENDIX

SOURCE CODE

Import the neccessary libraries

```
import keras
import tensorflow

from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

✓ 1m 18.2s Python

```
import tensorflow

from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

Python

```
train_datagen = ImageDataGenerator(rescale=1./255,
                                   shear_range=0.2,
                                   rotation_range=180,
                                   zoom_range=0.2,
                                   horizontal_flip=True)

test_datagen = ImageDataGenerator(rescale=1./255)
```

✓ 0.1s Python

```
x_train = train_datagen.flow_from_directory(r'./Dataset/train_set/',
                                            target_size=(128, 128),
                                            batch_size=32,
                                            class_mode='binary')
```

✓ 0.6s Python

```
Found 436 images belonging to 2 classes.
```

```
x_test = train_datagen.flow_from_directory(r'./Dataset/test_set/',
                                           target_size=(128, 128),
                                           batch_size=32,
                                           class_mode='binary')
```

✓ 0.1s Python

```
Found 121 images belonging to 2 classes.
```

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Convolution2D, MaxPooling2D, Flatten
```

✓ 0.1s Python

```
model = Sequential()
model.add(Convolution2D(32, (3,3), input_shape=(128, 128, 3), activation="relu"))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())
model.add(Dense(150,activation="relu"))
model.add(Dense(1, activation="sigmoid"))
```

✓ 1.8s Python

```python
model.compile(loss="binary_crossentropy",
              optimizer="adam",
              metrics=["accuracy"])
```

✓ 0.1s

```python
model.fit(x_train, steps_per_epoch=14, epochs=10, validation_data=x_test, validation_steps=4)
```

```
Epoch 1/10
14/14 [==============================] - 64s 4s/step - loss: 3.5440 - accuracy: 0.5665 - val_loss: 0.4052 - val_accuracy: 0.8430
Epoch 2/10
14/14 [==============================] - 23s 2s/step - loss: 0.5222 - accuracy: 0.7431 - val_loss: 0.2283 - val_accuracy: 0.9669
Epoch 3/10
14/14 [==============================] - 23s 2s/step - loss: 0.3097 - accuracy: 0.8647 - val_loss: 0.1622 - val_accuracy: 0.9504
Epoch 4/10
14/14 [==============================] - 22s 2s/step - loss: 0.2392 - accuracy: 0.8945 - val_loss: 0.1137 - val_accuracy: 0.9669
Epoch 5/10
14/14 [==============================] - 23s 2s/step - loss: 0.2125 - accuracy: 0.8968 - val_loss: 0.1337 - val_accuracy: 0.9504
Epoch 6/10
14/14 [==============================] - 23s 2s/step - loss: 0.1922 - accuracy: 0.9243 - val_loss: 0.0887 - val_accuracy: 0.9669
Epoch 7/10
14/14 [==============================] - 23s 2s/step - loss: 0.1773 - accuracy: 0.9266 - val_loss: 0.1454 - val_accuracy: 0.9339
Epoch 8/10
14/14 [==============================] - 21s 2s/step - loss: 0.1678 - accuracy: 0.9427 - val_loss: 0.0835 - val_accuracy: 0.9752
Epoch 9/10
14/14 [==============================] - 24s 2s/step - loss: 0.1733 - accuracy: 0.9243 - val_loss: 0.1079 - val_accuracy: 0.9669
Epoch 10/10
14/14 [==============================] - 25s 2s/step - loss: 0.1647 - accuracy: 0.9335 - val_loss: 0.0716 - val_accuracy: 0.9752

<keras.callbacks.History at 0x1920c974be0>
```

Save the model

```python
model.save("model.h5")
```

Prediction

```python
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import numpy as np
import cv2
```

Python

```python
model = load_model("model.h5")
```

Python

Reviewing the model

```python
img = image.load_img("forest-fire.jpg")
x = image.img_to_array(img)
res = cv2.resize(x, dsize=(128, 128), interpolation=cv2.INTER_CUBIC)
x = np.expand_dims(res, axis=0)
```

Python

```
pred = model.predict(x)
pred = int(pred[0][0])
pred
```

Python

```
1/1 [==============================] - 1s 524ms/step

0
```

Fire.py (Main file)

```python
import cv2
import numpy as np
from tensorflow.keras.preprocessing import image
from tensorflow.keras.models import load_model
from twilio.rest import Client
from playsound import playsound
from decouple import config


message_sent = False

model = load_model("./model.h5")

video = cv2.VideoCapture("fire.mp4")

name = ["No fire", "Fire Detected"]


def send_message():
    account_sid = config("ACCOUNT_SID")
    auth_token = config("AUTH_TOKEN")

    client = Client(account_sid, auth_token)
    message = client.messages.create(
        body="Forest Fire detected , Stay safe!!!",
        from_=config("FROM"),
        to=config("TO")
    )
    print(message.sid)
    print("Fire Detected")
    print("SMS Sent!")

playsound("./beep.mp3")
```

```python
while True:
    success, frame = video.read()
    cv2.imwrite("image.jpg", frame)
    img = image.load_img("image.jpg", target_size=(128, 128))
    x = image.img_to_array(img)
    x = np.expand_dims(x, axis=0)
    pred = model.predict(x)
    p = int(pred[0][0])
    cv2.putText(frame, str(name[p]), (100, 100), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 0), 1)

    if p == 1:
        if not message_sent:
            send_message()
            message_sent = True
        print("Fire Detected , stay safe!!!")
    else:
        print("No Fire Detected")

    cv2.imshow("Image", frame)

    if cv2.waitKey(1) & 0xFF == ord('x'):
        break

video.release()
cv2.destroyAllWindows()
```

PROJECT FILES AND DEMONSTRATION VIDEO

GITHUB LINK:

https://github.com/IBM-EPBL/IBM-Project-32048-1660207665

PROJECT DEMO:

File located in google drive.

https://drive.google.com/file/d/1Qn1UB28TqYp0Yfj2af5nb4dtajh3HK2K/view?usp=share_link