|  |
| --- |
| **EMERGING METHODS FOR EARLY DETECTION OF FOREST FIRES**  **A PROJECT REPORT**  ***Submitted by***  **SRIRAM.P (621419104034)**  **GOKUL.N (621419104009)**  **PRADEEP.R (621419104025) RAMACHANDIRAN.P(621419104026)**  **TEAM I’D:** **PNT2022TMID41576**  **BACHELOR OF ENGINEERING *IN***  **COMPUTER SCIENCE AND**  **ENGINEERING**      **MAHA BARATHI ENGINEERING COLLEGE** |

|  |  |
| --- | --- |
| **S.NO** | **TABLE OF CONTENT** |
| **1** | **INTRODUCTION**  1.1 Project Overview  1.2 Purpose |
| **2** | **LITERATURE SURVEY**  2.1 Existing problem  2.2 Reference  2.3 Problem Statement Definition |
| **3** | **IDEATION & PROPOSED SOLUTION**  3.1 Empathy Map Canvas  3.2 Ideation &Brainstorming  3.3 Proposed Solution  3.4 Problem Solution Fit |
| **4** | **REQUIREMENT ANALYSIS**  4.1 Functional requirements  4.2 Non-Functional requirements |
| **5** | **PROJECT DESIGN**  5.1 Data Flow Diagrams  5.2 Solution & Technical Architecture  5.3 User Stories |
| **6** | **PROJECT PLANNING & SCHEDULING**  6.1 Sprint Planning & Estimation  6.2 Sprint Delivery Schedule  6.3 Reports from JIRA |
| **7** | **CODING & SOLUTIONING (Explain the features**  **added in the project along with code)**  7.1 Feature 1  7.2 Feature 2  7.3 Database Scheme (if applicable) |
| **8** | **TESTING** |

|  |  |
| --- | --- |
|  | 8.1 Test Cases  8.2 User Acceptance Testing |
| **9** | **RESULT**  9.1 Performance Metrics |
| **10** | **ADVANTAGES &DISADVANTAGES** |
| **11** | **CONCLUSION** |
| **12** | **FUTURE SCOPE** |
| **13** | **APPENDIX**  Source Code  GitHub & Project Demo Link |

**1.INTRODUCTION**

**1.1.PROJECT OVERVIEW**

Forests are the protectors of earth’s ecological balance. Unfortunately, the forest fire is usually only observed when it has already spread over a large area, making its control and stoppage arduous and even impossible at times. The result is devastating loss and irreparable damage to the environment and atmosphere (30% of carbon dioxide (CO2) in the atmosphere comes from forest fires), in addition to irreparable damage to the ecology (huge amounts of smoke and carbon dioxide (CO2) in the atmosphere). Fast and effective detection is a key factor in forest fire fighting. To avoid uncontrollable wide spreading of forest fires it is necessary to detect fires in an early state and to prevent the propagation. Nowadays, image processing are critical components of the increasingly object detections . Such systems have a large applicability, and the environmental monitoring field can also benefit from their innovation.

**1.2.PURPOSE:**

. The purpose of the image processing concept is to capture the image from the real world and every day scenario appliances, etc., into intelligent interconnected virtual objects. By keeping the user informed on the state of things and giving the users control of things, a better global humans-devices-humans communication can be achieved.

# 2.LITERATURE SURVEY

**2.1.EXISTING METHOD:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **AUTHOR** | **TITLE** | YEAR |
| 1. | G.V.Hristor  Diyana kyuchukova  Jordan Raychev | Emerging method for early detection of  forest fires using unmanned Aerial vehicles and Lorawan sensor network | 2018 |

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **AUTHOR** | **TITLE** | YEAR |
| 2. | Panagiotis barmpoutis Konsmas dimitropoulas Nikos grammalidis. | A review on early forest fire detection systems using optical remote sensing | 2020 |

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **AUTHOR** | **TITLE** | YEAR |
| 3. | Hamdy soliman | S-mart forest fires early detection sensory system: Another approach of utilizing wireless sensor and neutral networks | 2010 |

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **AUTHOR** | **TITLE** | YEAR |
| 4. | Lilu cui  Chaolong yao  Zhengbo zou | The influence of climate change on  forest fires in  Yunnan province, Southwest china detected by GRACE satellites | 2022 |

.

**2.2.REFERENCES:**

[1]Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems.

2012.

1. A. Grivei, A. Rdoi, C. Vduva and M. Datcu, "An Active-Learning approach to the query by example retrieval in remote sensing images,"

International Conference on Communications (COMM), pp. 377-380, 2016.

1. G. Suciu, et al. “Remote Sensing for Forest Environment Preservation”

WorldCIST, Recent Advances in Information Systems and Technologies, pp. 211-220, 2017.

1. E. Olteanu, et al. "Forest Monitoring System Through Sound

Recognition." In 2018 International Conference on Communications (COMM), pp. 75-80. IEEE, 2018.

[4] Arasvathi, Nahalingham and Chelsea, Ferdyanti Kosasih “Study and

Implementation of Internet of Things (IoT) Based Forest Fire

Automation System to Detect and Prevent Wildfire”. INTI Journal, 1

(15) , pp. 1-5, 2018.

[5]J. Papán, M. Jurecka, J. Púchyová, “WSN for Forest Monitoring to Prevent Illegal Logging”, Proceedings of the Federated Cinference on Computer Science and Information Systems, pp. 809-812, 2012.

[6]Krivtsova et al. “Implementing a broadcast storm attack on a mission- critical wireless sensor network” In: International Conference on Wired/Wireless Internet Communication, 2016.

[7]Chen, Thou-Ho, Cheng-Liang Kao, and Sju-Mo Chang. "An intelligent real-time fire-detection method based on video processing." Security Technology, 2003. Proceedings. IEEE 37th

Annual 2003 International Carnahan Conference on. IEEE, 2003.

[8] Chen, Thou-Ho, et al. "The smoke detection for early fire- alarming system base on video processing." Intelligent

Information Hiding and Multimedia Signal Processing, 2006. IIH-

MSP'06. International Conference on. IEEE, 2006

1. **3.PROBLEM STATEMENT DEFINITION:**

The most common hazard in forests is forests fire. They pose a treat not only to the forest wealth but also to the entire regime to fauna and flora seriously disturbing the bio – diversity and the ecology and environment of region.During summer , when there is no rain for months, the forests become littered with dry senescent leaves and twinges, which could burst into flames ignited by the slightest spark.Forest fire causes imbalances in nature and endangers biodiversity by reducing faunal and floral wealth.Traditional methods of fire prevention are not proving effective and it is now essential to raise public awareness on the matter , particularly among those people who live close to or in forested areas.

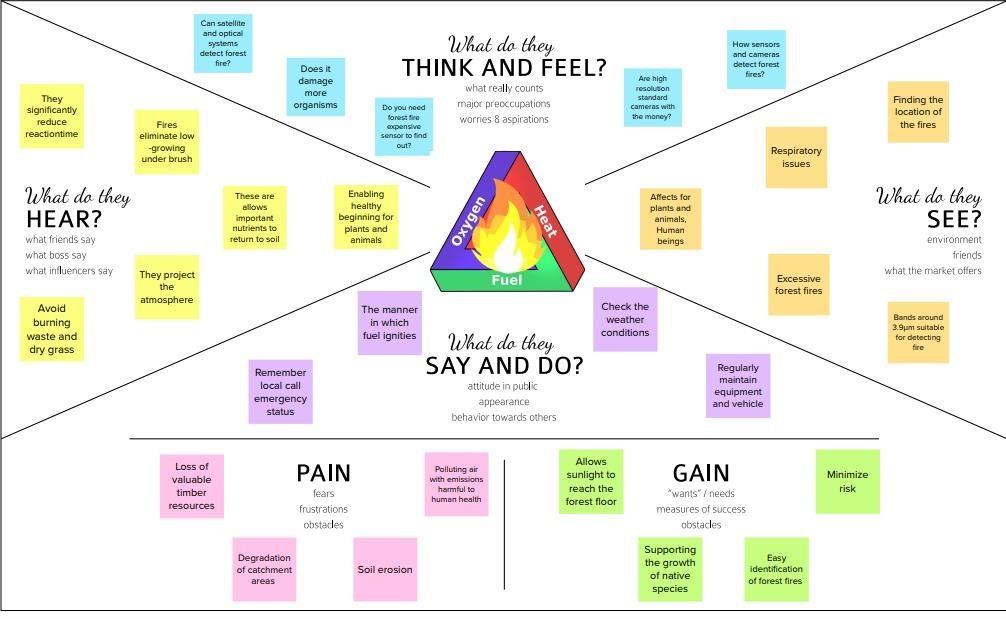
|  |  |
| --- | --- |
| I am | Humans are responsible for 75% of all forest fire. Naturally occurring forest fires can be caused by lightning, volcanic activity and coal seam fires , though these are relatively rare. |
| I’m trying to | Using the recent technologies to avoid forest fires in Deep learning based on pre-trained satellite image processing and forest officer can view the recommendable forest fires through Gmail sms so avoid overexposure. |
| But, | I don’t know much about the recent technology that helps me predict forest fires, and I haven’t found the right solutions for forest fires. |
| Becaus | I don’t want to cause devasting damage to both nature and humans, air pollution,every fire huge amounts of gases released in the atmosphere . |
| Which makes me feel | I’m not capable of early detect the fires and maintaining the area clean of forest but I trying solution for this problem. |

**3**

**.IDEATION & PROPOSED SOLUTION**

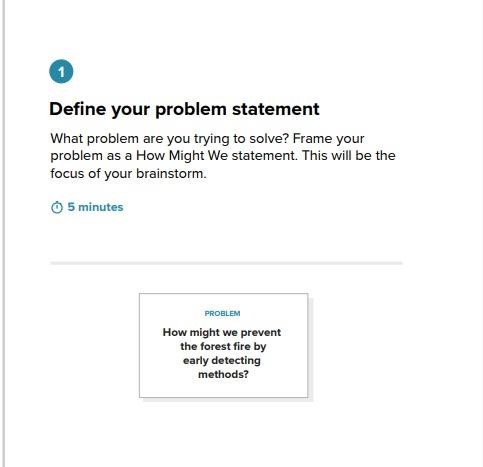
**3.1.**

EMPATHY MAP CANVAS

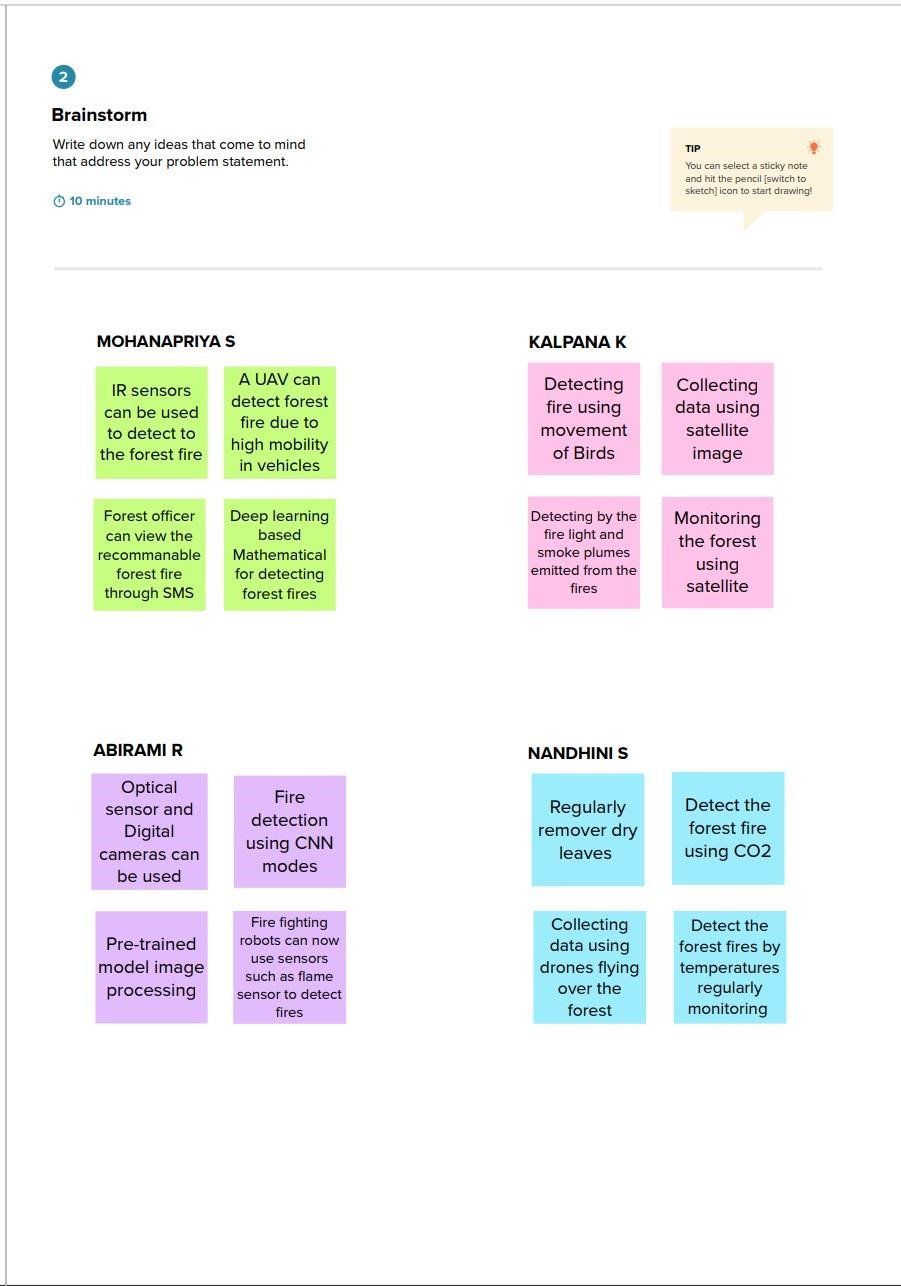


1. **2.BRAINSTORMING :**

PROBLEM :



**Brainstorm:**



**Group ideas**:



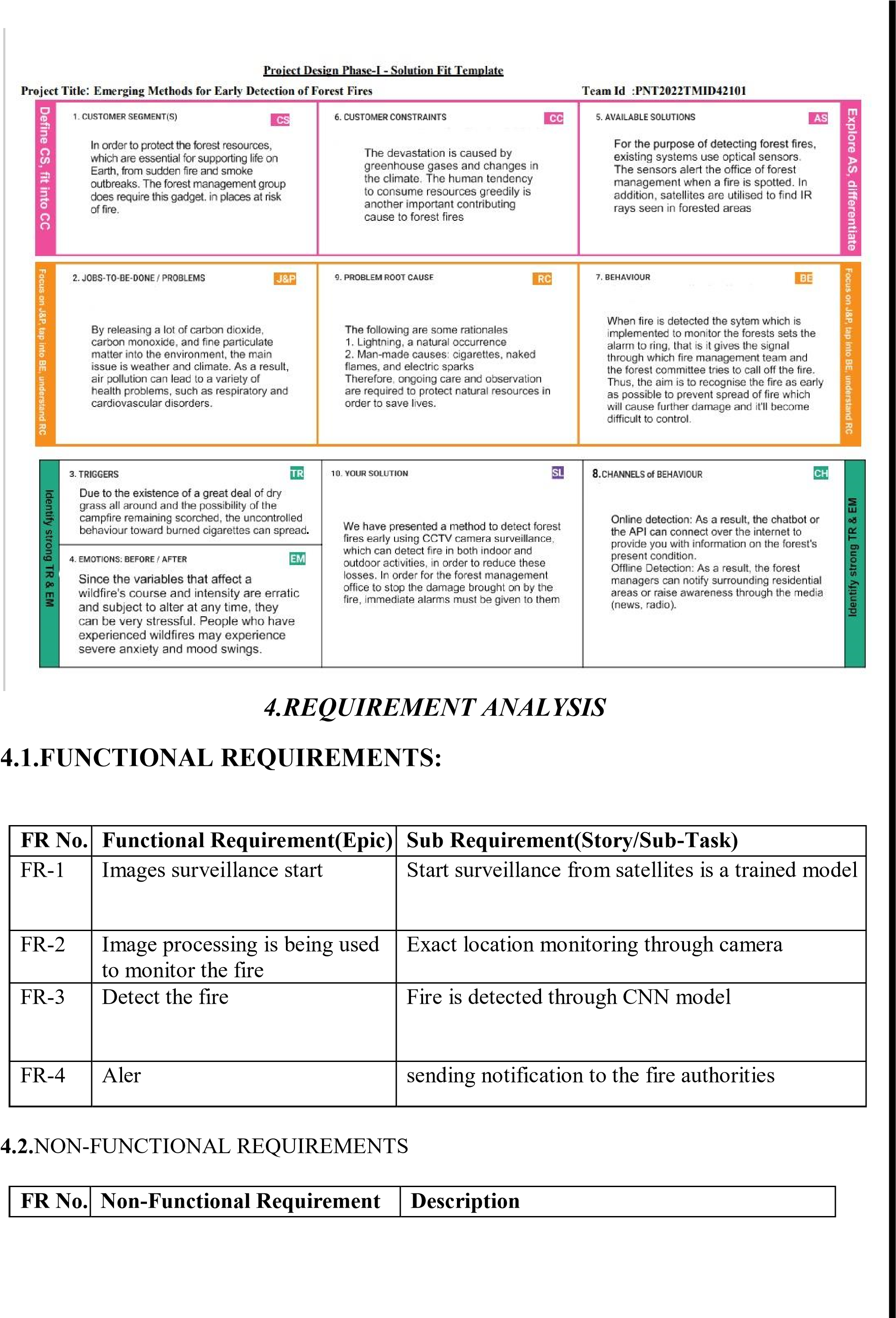
**Priortize:**



**3.3.PROPOSED SOLUTION:**

|  |  |  |
| --- | --- | --- |
| **S. No.** | **Parameter** | **Description** |
| 1. | Problem Statement (Problem to be solved) | Forests are one of the main factors in balancing the ecology.Forest fires are one of the most worrisome natural  disasters,destroying thousands of acres of forests and nearby urban zones,affecting plant,animals and human life.So, the fire detection is important in this scenario. Finding of the exact location of the fire and sending notification to the fire authorities soon after the occurrence of fire can make a positive impact |
| 2. | Idea / Solution description | Our solution aims at collecting the dataset to test and train the model . The damage and the cost for distinguish fire because of forest fire can be reduced when the fire detected early as possible. So, the fire detection is important in this scenario. Finding of the exact location of the fire and sending notification to the fire authorities soon after the occurrence of fire can make a positive impact. We have implemented a fire detection system to detect fire by capturing images. The system uses CNN(convolutional neural network), and image processing techniques. |
| 3. | Novelty / Uniqueness | Real time computer program detect forest firein earliest before it spread to larger area. Our proposed system depends on using AI to make it cheaper and easier for the forest management. Accuracy and timely prediction using AI, CNN and API made it possible. |
| 4. | Social Impact / Customer Satisfaction | The destroying homes, wildlife habitat and timber, and polluting the air with emissions harmful to human health. The proposed solution fulfills the satisfaction requirements of the customer as it provides instant alerts on fire detection which helps the forest officer to take action as soon as possible. |
| 5. | Business Model (Revenue Model) | A working model in which mini cameras continuously monitor the forest area and capture live images from satellites is a trained model that automatically detects fire or smoke. This proposed model can detect the exact location of the fire and can be activated by SMS.The fire officer can implement quick responses and preventive measures. |
| 6. | Scalability of the Solution | The device should be compatible with a minimum of 4GB RAM and WINDOWS  10 (x64 bit) and 100 GB ROM to support usage of various software like PYTHON 3.6.5. Testing and training undergo using  latest technology like KERAS  ,TENSORFLOW ,NUMPY and PILLOW |

**3.4.PROPOSED SOLUTION FIT:**

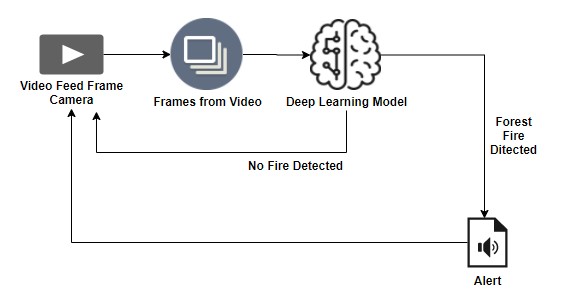


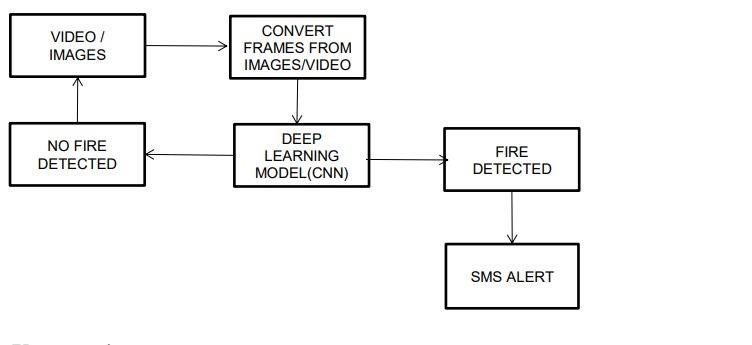
|  |  |  |
| --- | --- | --- |
| NFR-1 | **Usability** | Usability is a unique and significant perspective to analyze user requirements, which can further improve the design quality, according to AI devices with machine learning. |
| NFR-2 | **Security** | * HD and powerful CCTV cameras are used. * The fire is found using image processing and 24-hour monitoring. |
| NFR-3 | **Reliability** | A real-time and dependable fire detection method for an early warning system is required to ensure an effective response to an incident. |
| NFR-4 | **Performance** | * The system is intended to monitor forest fires through image processing via a camera. * CCTV cameras are used to process images and detect forest fires. * The twilio module is used to send the forest officer an alert message |
| NFR-5 | **Availability** | * By progressing to a more advanced system that uses real-time CCTV cameras to detect and alert on fires. * The convolutional neural network algorithm is extremely useful for detecting fire in captured images |
| NFR-6 | **Scalability** | By detecting forest fires early, we can prevent loss of life as well as resource damage while decreasing air pollution, landslides, soil erosion, and Emission emissions into the environment |

**5.PROJECT DESIGN**

**5.1.DATA FLOW DIAGRAMS:**

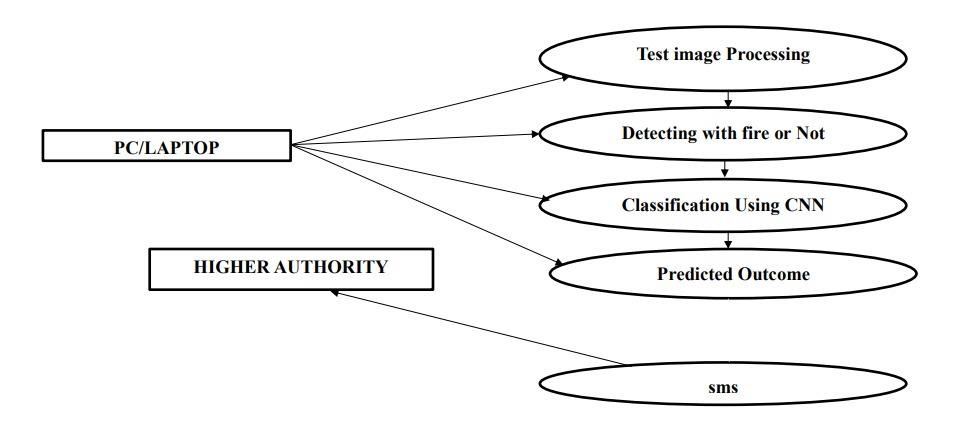
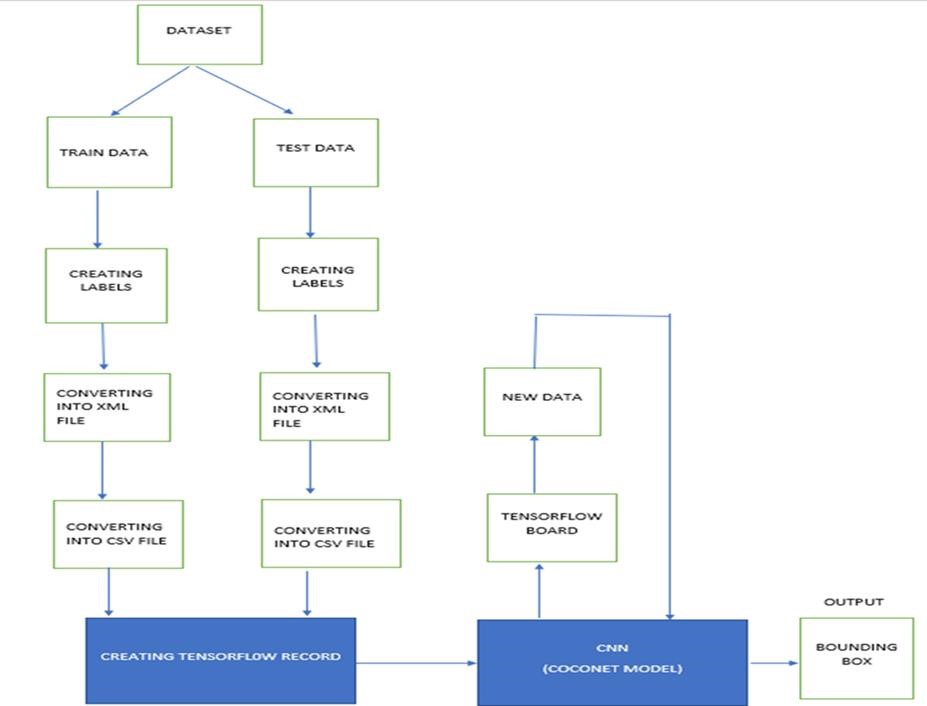
The traditional visual representation of how information moves through a system is a data flow diagram (DFD). A tidy and understandable DFD can graphically represent the appropriate amount of the system requirement. It demonstrates how information enters and exits the system, what modifies the data, and where information is kept.





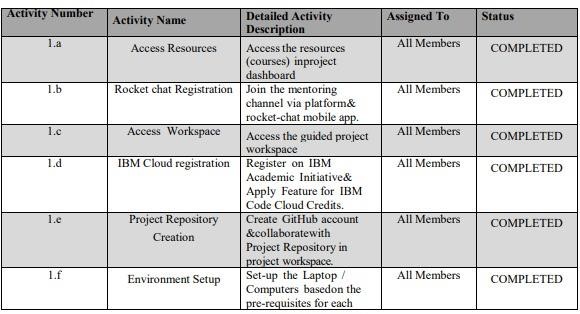
**?5.2.SOLUTION AND TECHNICAL ARCHIETECTURE:**

**Solution Architecture**:

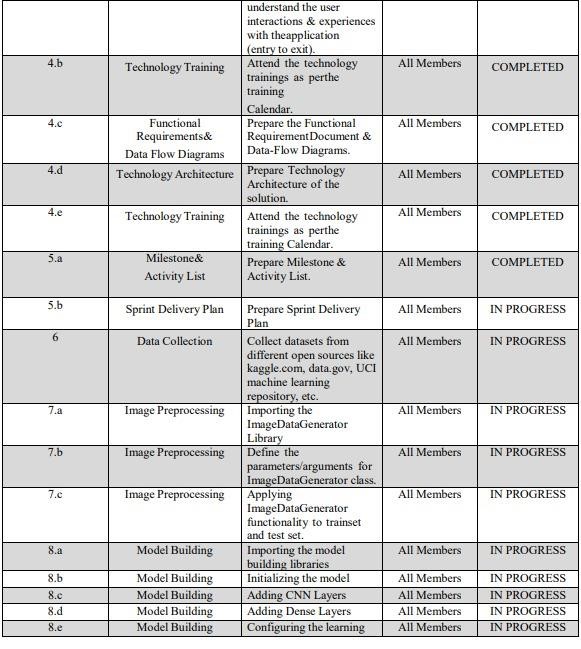


|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **5.3.USER STORIES:**  Use the below template to list all the user stories for the product. | | | | | | | | | | | | |
| **User story** | | **Function**  **al**  **Require ment(Epi**  **c)** | | **User**  **Story**  **Numbe**  **r** | **User Story/Task** | **Acceptance criteria** | | | **Priority** | | **Release** | |
| Environ  mentalis  t | | Collect the data | | USN-1 | It is necessary for an animal rights activist to gather information about forest fires. | We must  collect the correct data.because of prediction. | | | High | | Sprint-1 | |
|  | |  | | USN-2 | Determine which algorithms can be used for prediction. | To gather the algorithms and determine each algorithm's accuracy . | | | Medium | | Sprint-2 | |
|  | | Implement Algorithm | | USN-3 | Determine each algorithm's accuracy. | Accuracy of the algorithm is must to be calculated . | | | High | | Sprint-2 | |
|  | |  | | USN-4 | assess the data set | Data is preprocessing before the training. | | | High | | Sprint-1 | |
|  | | Evaluate Accuracy  of  Algorithm | | USN-5 | Decide the precision, accuracy, as well as recall of each algorithm. | Accuracy is important to detect the seviearity of fire | | | High | | Sprint-3 | |
| **6.PROJECT PLANNING & SCHEDULING:**  **6.1.SPRINT PLANNING & ESTIMATION:** | | | | | | | | | | | | |
| **Sprint** | **Functional**  **Requirement**  **(Epic)** | | **User**  **Story**  **Number** | | **User Story / Task** | | **Story Points** | **Priority** | | **Team**  **Members** | |  |
|  | | | | | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Sprint-1 | Download data set | USN-1 | The data is downloaded from the Kaggle website and then the data set is classified into training and testing images. |  | 5 | High | S.Mohanapriy  a  R.Abirami  K.Kalpana  S.Nanthini |  |
| Sprint-2 | Image preprocessing | USN-2 | In Image processing technique the first step is usually importing the libraries that will be  needed    In the program. Import Keras library from that library and import the  ImageDataGenerator Library to your Python script.    The next step is defining the arguments for the ImageDataGenerator  And next step is applying the ImageDataGenerator arguments to the train and test dataset. |  | 10 | Medium | S.Mohanapriy  a  R.Abirami  K.Kalpana  S.Nanthini |
| Sprint-3 | Training image | USN-3 | In this training phase the ImageDataGenerator arguments is applied to the training images and the model is tested with several images and the model is saved. | 5 |  | High | S.Mohanapriy  a  R.Abirami  K.Kalpana  S.Nanthini |
| Sprint-4 | Testing  Image, Evaluation metrics and accuracy | USN-4 | In this testing phase the Image processing techniques is applied to the testing images and executed for prediction. In this phase the result, prediction, accuracy, and performance of the project are tested. | 5 |  | High | S.Mohanapriy a  R.Abirami  K.Kalpana  S.Nanthini |
| MILESTONE & ACTIVITY LIST: | | | | |  |  | |







|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **6.2.SPRINT DELIVERY SCHEDULE:** | | | | | | | |
| **Sprint** | **Total**  **Story**  **Points** | **Duratio n** | **Sprint**  **Start**  **Date** | **Sprint End**  **Date**  **(Planned)** | **Story Points**  **Completed**  **(as on**  **Planned End**  **Date)** | **Sprint**  **Release Date**  **(Actual)** |  |
| Sprint-1 | 5 | 6 Days | 24 Oct 2022 | 29 Oct 2022 | 5 | 29 Oct 2022 |
| Sprint-2 | 10 | 6 Days | 31 Oct 2022 | 05 Nov 2022 | 10 | 05 Nov 2022 |
| Sprint-3 | 5 | 6 Days | 07 Nov 2022 | 12 Nov 2022 | 5 | 12 Nov 2022 |
| 5 6 Days 14 Nov 19 Nov 2022  Sprint-4 2022        **6.3.REPORTS FROM JIRA:** | | | | | 5 | 19 Nov 2022 |

**7**

**.CODING & SOLUTIONING**

**7.1**

**.FEATURE**

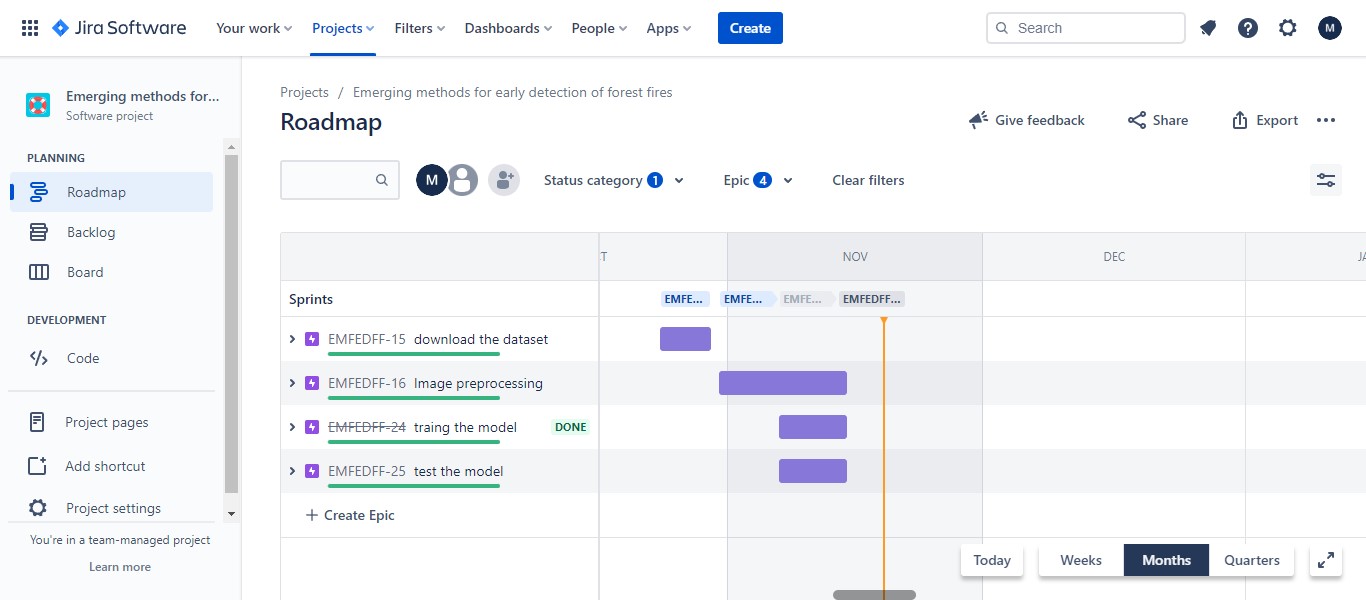
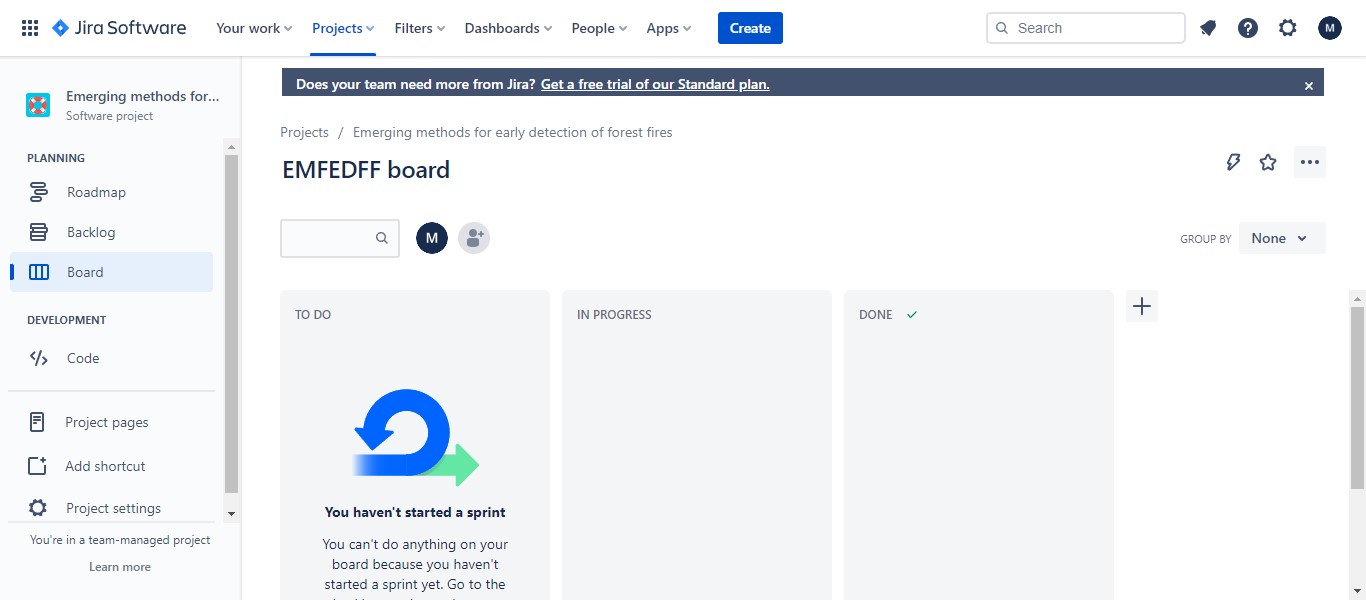
**1:**

Keras leverages various optimization techniques to make high level neural network API easier

and more performant. It supports the following features:



Consistent, simple and extensible API.



* Minimal structure - easy to achieve the result without any frills.
* It supports multiple platforms and backends.
* It is user friendly framework which runs on both CPU and GPU.
* Highly scalability of computation.

**1.IMAGE DATA GENERATOR:**

Keras ImageDataGenerator is used for getting the input of the original data and further, it makes the transformation of this data on a random basis and gives the output resultant containing only the data that is newly transformed. It does not add the data.

From keras.preprocessing.image import ImageDataGenerator

**2.PARAMETRES**

**2.1.Rescale:**

The ImageDataGenerator class can be used to rescale pixel values from the range of 0-255 to the range 0-1 preferred for neural network models. Scaling data to the range of 0-1 is traditionally referred to as normalization.

**2.2.Shear Range:**

Shear range means that the image will be distorted along an axis, mostly to create or rectify the perception angles. It's usually used to augment images so that computers can see how humans see things from different angles.

**2.3.Rotation range:**

ImageDataGenerator class allows you to randomly rotate images through any degree between 0 and 360 by providing an integer value in the rotation\_range argument. When the image is rotated, some pixels will move outside the image and leave an empty area that needs to be filled in.

**2.4.Zoom Range:**

The zoom augmentation method is used to zooming the image. This method randomly zooms the image either by zooming in or it adds some pixels aroundthe image to enlarge the image. This method uses the zoom\_range argument of the ImageDataGenerator class. We can specify the percentage value of the zooms either in a float**,** range in the form of an array.

1. **5.Horizontal Flip:**

Horizontal flip basically flips both rows and columns horizontally. So for this, we have to pass the horizontal\_flip=True argument in the ImageDataGenerator constructor.

**3.CONVOLUTION NEURAL NETWORK:**

Convolutional neural network is one of the most popular ANN. It is widely used in the fields of image and video recognition. It is based on the concept of convolution, a mathematical concept. It is almost similar to multi-layer perceptron except it contains series of convolution layer and pooling layer before the fully connected hidden neuron layer.

1. **1.Convolutional Layer:**

Convolution layer: It is the primary building block and perform computational tasks based on convolution function.

**3.2.Pooling Layer:**

Pooling layer: It is arranged next to convolution layer and is used to reduce the size of inputs by removing unnecessary information so computation can be performed faster.

**3.3.Fully Connected Layer:**

Fully connected layer: It is arranged to next to series of convolution and pooling layer and classify input into various categories

**7.2.FEATURE 2(CODE): Image Pre-Processing:**

from google.colab import drive drive.mount('/content/drive')

## Importing ImageDataGenerator from Keras

from keras.preprocessing.image import ImageDataGenerator

**Defining the Parameters**

train\_datagen = ImageDataGenerator(rescale = 1./255, shear\_range = 0.2, zoom\_range = 0.2, horizontal\_flip = True)

val\_datagen = ImageDataGenerator(rescale = 1./255)

## Applying ImageDataGenerator functionality to train dataset

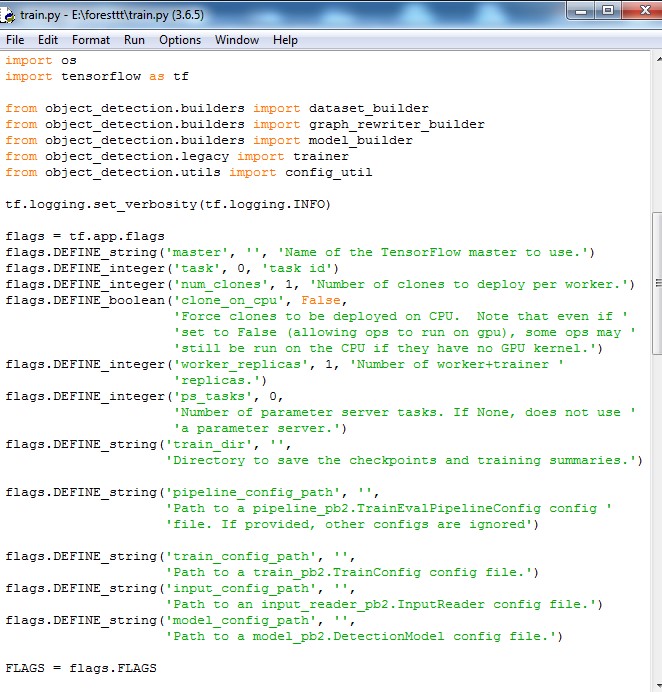
training\_set = train\_datagen.flow\_from\_directory('/content/drive

/MyDrive/Data set Collection', target\_size = (512,512), batch\_size = 8, class\_mode = 'binary') val\_set = val\_datagen.flow\_from\_directory('/content/drive/MyDriv e/Data set Collection', target\_size = (512,512), batch\_size = 8, class\_mode = 'binary') x\_train = train\_datagen.flow\_from\_directory(r'/content/drive/MyD rive/Data set Collection') x\_test = val\_datagen.flow\_from\_directory('/content/drive/MyDrive

/Data set Collection')

**Training code:**

To train the model we will use the train.py file, which is located in the object\_ detection/legacy folder. We will copy it into the object\_detection folder and then we will open a command line and type default.



**TESTING OBJECT DETECTOR**

In order to test our newly created object detector, we can use the code which we already created.

**8**

**.TESTING**

**8.1**

**.Test Cases**

**:**

**8.2.**

**User Acceptance Testing:**

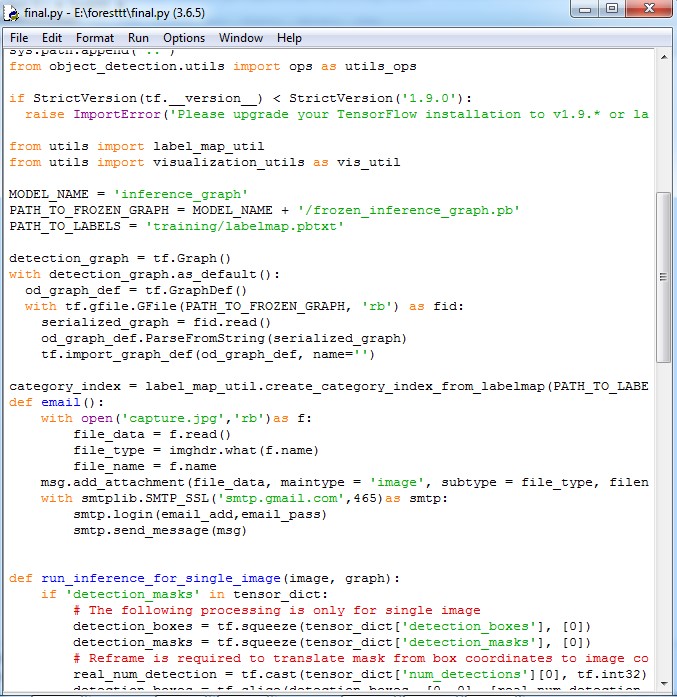
**1.**

**Purpose**

**of**

**Document**

**:**



The purpose of this document is to briefly explain the test coverage and open issues of the Emerging Methods for Early Forest Fire Detection Project at the time of the release to User Acceptance Testing (UAT).

1. **Defect Analysis:**

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Resolution** | **Severity 1** | **Severity 2** | **Severity 3** | **Severity 4** | **Subt otal** |
| By Design | 1 | 0 | 0 | 0 | 1 |
| Duplicate | 0 | 0 | 0 | 0 | 0 |
| External | 0 | 0 | 0 | 0 | 0 |
| Fixed | 0 | 0 | 0 | 0 | 0 |
| Not Reproduced | 0 | 2 | 0 | 0 | 2 |
| Skipped | 0 | 0 | 0 | 0 | 0 |
| Won't Fix | 0 | 0 | 0 | 0 | 0 |
| Totals | 1 | 2 | 0 | 0 | 3 |

1. **Test Case Analysis:**

This report shows the number of test cases that have passed, failed, and untested.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Section** | **Total Cases** | **Not Tested** | **Fail** | **P a s**  **s** |
| Performance | 5 | 0 | 0 | 5 |
| UI | 1 | 0 | 0 | 1 |
| Security | 3 | 0 | 0 | 3 |

|  |  |  |  |
| --- | --- | --- | --- |
| **9.RESULTS**  **9.1.PERFORMANCSE METRICS:** | | | |
| **S.No.** | **Parameter** | **Values** |  |
| 1. | Model  Summary |  |
| 2. | Accuracy | Training Accuracy - 94.50%    Validation Accuracy - 98.35% |
|  | | |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| 10.**ADVANTAGES & DISADVANTAGES:**    **ADVANTAGES:**  .   * CNN gives best accuracy when compared to machine learning techniques. * Can be used for classification with above 96% accuracy.         **DISADVANTAGES:**   * The temperature sensor which has a particular range to detect fire in a forest. * We can’t cover large area to detect fire in a whole forest. * There is some specific range to sense the smoke using smoke sensor as same as temperature sensor which only covers limited area.   . | | |

## 11. CONCLUSION

The recent improved processing capabilities of smart devices have shown promising results in surveillance systems for identification of different abnormal events i.e., fire, accidents, and other emergencies. Fire is one of the dangerous events which can result in great losses if it is not controlled on time. This necessitates the importance of developing early fire detection systems. Therefore, in this research article, we propose a cost-effective fire detection CNN architecture for forest architecture. Translations and content mining are permitted for academic research only. Although, this work improved the flame detection accuracy, yet the number of false alarms is still high and further research is required in this direction. In addition, the current flame detection frameworks can be intelligently tuned for detection of fire. This will enable the video surveillance systems on forest to handle more complex situations in real-world.

## 12.FUTURE SCOPE

* Supporting research to improve the understanding of forest fires and their ecology, ecological and social costs and benefits, causes and management options.
* Building awareness among st policy-makers, the public and the media of the underlying causes of catastrophic forest fires.
* Mandating and equipping managers to implement integrated fire management programs.
* Involving local communities and land managers in management planning and implementation, assisting them to participate effectively.
* Developing and enforcing compatible and mutually reinforcing land-use laws that provide a legal basis for the ecologically appropriate use of fire.
* Discouraging land management practices that predispose forests to harmful fires.
* Promoting management strategies to mimic natural fire regimes, including techniques such as prescribed burns and managed wildfires.
* Avoiding manipulating natural or well-established fire regimes.
* Establishing reliable fire monitoring systems that provide early warning of high fire risk and fire occurrence, and include evaluation of ecological and human impacts of fire.
* Preventing further forest loss and degradation from recurrent catastrophic fires, and reduce fire risk in forested landscapes, through ecologically appropriate restoration.

**13.APPENDIX:**

**SOURCE CODE:**

**TRANING CODE**:

import functools import json import os import tensorflow as tf

from object\_detection.builders import dataset\_builder from object\_detection.builders import graph\_rewriter\_builder from object\_detection.builders import model\_builder from object\_detection.legacy import trainer from object\_detection.utils import config\_util

tf.logging.set\_verbosity(tf.logging.INFO)

flags = tf.app.flags

flags.DEFINE\_string('master', '', 'Name of the TensorFlow master to use.') flags.DEFINE\_integer('task', 0, 'task id') flags.DEFINE\_integer('num\_clones', 1, 'Number of clones to deploy per worker.') flags.DEFINE\_boolean('clone\_on\_cpu', False,

'Force clones to be deployed on CPU. Note that even if '

'set to False (allowing ops to run on gpu), some ops may ' 'still be run on the CPU if they have no GPU kernel.') flags.DEFINE\_integer('worker\_replicas', 1, 'Number of worker+trainer '

'replicas.')

flags.DEFINE\_integer('ps\_tasks', 0,

'Number of parameter server tasks. If None, does not use '

'a parameter server.')

flags.DEFINE\_string('train\_dir', '',

'Directory to save the checkpoints and training summaries.') flags.DEFINE\_string('pipeline\_config\_path', '',

'Path to a pipeline\_pb2.TrainEvalPipelineConfig config '

'file. If provided, other configs are ignored')

flags.DEFINE\_string('train\_config\_path', '',

'Path to a train\_pb2.TrainConfig config file.') flags.DEFINE\_string('input\_config\_path', '',

'Path to an input\_reader\_pb2.InputReader config file.') flags.DEFINE\_string('model\_config\_path', '',

'Path to a model\_pb2.DetectionModel config file.')

FLAGS = flags.FLAGS

@tf.contrib.framework.deprecated(None, 'Use object\_detection/model\_main.py.') def main(\_):

assert FLAGS.train\_dir, '`train\_dir` is missing.' if FLAGS.task == 0: tf.gfile.MakeDirs(FLAGS.train\_dir) if FLAGS.pipeline\_config\_path: configs = config\_util.get\_configs\_from\_pipeline\_file( FLAGS.pipeline\_config\_path) if FLAGS.task == 0:

tf.gfile.Copy(FLAGS.pipeline\_config\_path, os.path.join(FLAGS.train\_dir, 'pipeline.config'), overwrite=True) else:

configs = config\_util.get\_configs\_from\_multiple\_files( model\_config\_path=FLAGS.model\_config\_path, train\_config\_path=FLAGS.train\_config\_path, train\_input\_config\_path=FLAGS.input\_config\_path) if FLAGS.task == 0:

for name, config in [('model.config', FLAGS.model\_config\_path),

('train.config', FLAGS.train\_config\_path), ('input.config', FLAGS.input\_config\_path)]: tf.gfile.Copy(config, os.path.join(FLAGS.train\_dir, name), overwrite=True)

model\_config = configs['model'] train\_config = configs['train\_config'] input\_config = configs['train\_input\_config']

model\_fn = functools.partial( model\_builder.build, model\_config=model\_config, is\_training=True)

def get\_next(config):

return dataset\_builder.make\_initializable\_iterator( dataset\_builder.build(config)).get\_next()

create\_input\_dict\_fn = functools.partial(get\_next, input\_config)

env = json.loads(os.environ.get('TF\_CONFIG', '{}')) cluster\_data = env.get('cluster', None)

cluster = tf.train.ClusterSpec(cluster\_data) if cluster\_data else None task\_data = env.get('task', None) or {'type': 'master', 'index': 0} task\_info = type('TaskSpec', (object,), task\_data)

# Parameters for a single worker.

ps\_tasks = 0 worker\_replicas = 1 worker\_job\_name = 'lonely\_worker' task = 0 is\_chief = True master = ''

if cluster\_data and 'worker' in cluster\_data:

# Number of total worker replicas include "worker"s and the "master". worker\_replicas = len(cluster\_data['worker']) + 1 if cluster\_data and 'ps' in cluster\_data: ps\_tasks = len(cluster\_data['ps'])

if worker\_replicas > 1 and ps\_tasks < 1:

raise ValueError('At least 1 ps task is needed for distributed training.')

if worker\_replicas >= 1 and ps\_tasks > 0:

# Set up distributed training.

server = tf.train.Server(tf.train.ClusterSpec(cluster), protocol='grpc', job\_name=task\_info.type, task\_index=task\_info.index) if task\_info.type == 'ps':

server.join() return

worker\_job\_name = '%s/task:%d' % (task\_info.type, task\_info.index) task = task\_info.index is\_chief = (task\_info.type == 'master') master = server.target

graph\_rewriter\_fn = None if 'graph\_rewriter\_config' in configs:

graph\_rewriter\_fn = graph\_rewriter\_builder.build( configs['graph\_rewriter\_config'], is\_training=True)

trainer.train( create\_input\_dict\_fn, model\_fn, train\_config, master, task,

FLAGS.num\_clones,

worker\_replicas,

FLAGS.clone\_on\_cpu,

ps\_tasks, worker\_job\_name,

is\_chief,

FLAGS.train\_dir, graph\_hook\_fn=graph\_rewriter\_fn) if \_\_name\_\_ == '\_\_main\_\_':

tf.app.run()

**TEST 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 os from twilio.rest import Client

account\_sid = 'AC17937b910b4774c95b3b07358838a842' auth\_token = '4b511b022adaac785f6b79b32cac4f28' client1 = Client(account\_sid, auth\_token) def sms(): message = client1.messages \

.create(

body='fire detected successfully', from\_='+13465214387', to='+9199434 63572'

)

print(message.sid)

# 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') sms()

if output\_dict['detection\_classes'][0] == 2 and output\_dict['detection\_scores'][0] > 0.70:

print('FIRE') sms()

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)

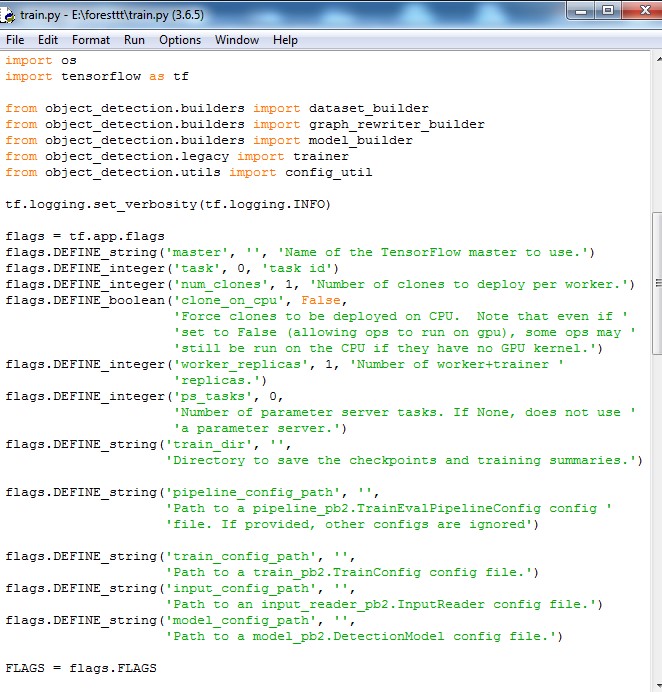
cv2.imshow('Frame', cv2.resize(image\_np,(800,600))) if cv2.waitKey(1) == ord('q'): cap.release()

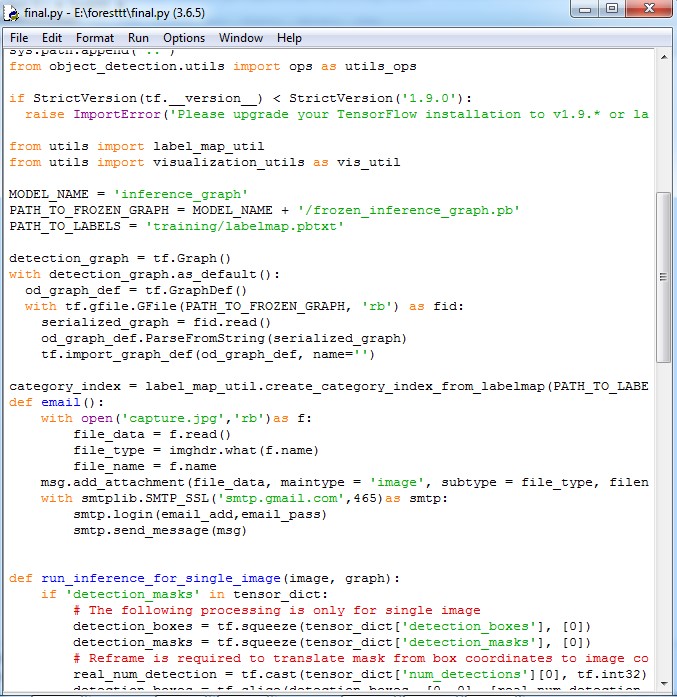
cv2.destroyAllWindows()

break

except Exception as e:

print(e) cap.release()





Our Github Link:

https://github.com/IBM

-

EPBL/IBM

-

Project

-

52135

-

1664613797

