 EMERGING METHODS FOR

EARLY DETECTION OF

FOREST FIRES

IBM NALAIYA THIRAN

## PROJECT REPORT

***Submitted By***

**NANDHA KUMAR G (611219104063)**

**SANDEEP KUMAR VARMA M (611219104081)**

**SANJAY GJ (611219104084)**

**SUDHIR KUMAAR SJ (611219104109)**

***in partial fulfillment for the award of the degree***

***of***

**BACHELOR OF ENGINEERING**

***in***

## COMPUTER SCIENCE AND ENGINEERING

**KNOWLEDGE INSTITUTE OF TECHNOLOGY,**

## SALEM-637504

**ANNA UNIVERSITY :: CHENNAI 600 025**

**NOVEMBER 2022**



 EMERGING METHODS FOR

EARLY DETECTION OF

FOREST FIRES

IBM NALAIYA THIRAN

## PROJECT REPORT

***Submitted By***

**NANDHA KUMAR G (611219104063)**

**SANDEEP KUMAR VARMA M (611219104081)**

**SANJAY GJ (611219104084)**

**SUDHIR KUMAAR SJ (611219104109)**

***in partial fulfillment for the award of the degree***

***of***

**BACHELOR OF ENGINEERING**

***in***

## COMPUTER SCIENCE AND ENGINEERING

**KNOWLEDGE INSTITUTE OF TECHNOLOGY,**

## SALEM-637504

**ANNA UNIVERSITY :: CHENNAI 600 025**

**NOVEMBER 2022**



**BONAFIDE CERTIFICATE**

Certified that this project report titled **“EMERGING METHODS FOR EARLY DETECTION OF FOREST FIRE”** is the bonafide work of **“NANDHAKUMAR G (611219104063), SANDEEP KUMAR VARMA M (611219104081), SANJAY GJ (611219104084), SUDHIR KUMAAR SJ (611219104109)”** who carried out the project work under my supervision.

|  |  |
| --- | --- |
| **SIGNATURE** | **SIGNATURE** |
| Dr. V. KUMAR M.E., Ph.D., | Mr. A. GOPALAKRISHNAN M.E., |
| **HEAD OF THE DEPARTMENT** | **SUPERVISOR** |
| **PROFESSOR** | **ASSISTANT PROFESSOR** |
| Department of Computer Science | Department of Computer Science |
| and Engineering, | and Engineering, |
| Knowledge Institute of Technology, | Knowledge Institute of Technology, |
| Kakapalayam, | Kakapalayam, |
| Salem- 637 504. | Salem- 637 504. |

## ---------------------------- -----------------------------------------------------

## SPOC HEAD OF THE DEPARTMENT

## ACKNOWLEDGEMENT

At the outset, we express our heartfelt gratitude to **GOD,** who has been our strength to bring this project to light.

At this pleasing moment of having successfully completed our project, we wish to convey our sincere thanks and gratitude to our beloved president **Mr. C. Balakrishnan,** who has provided all the facilities to us.

We would like to convey our sincere thanks to our beloved Principal **Dr. PSS. Srinivasan,** for forwarding us to do our project and offering adequate duration in completing our project.

We express our sincere thanks to our Head of the Department **Dr. V. Kumar,** Department of Computer Science and Engineering for fostering the excellent academic climate in the Department.

We express our pronounced sense of thanks with deepest respect and gratitude to our internal guide **Mr. A. Gopalakrishnan,** Assistant Professor Computer Science and Engineering Department and Faculty Evaluator **Mr. P. Sasikumar,** Assistant Professor, Department of Computer Science and Engineering for their valuable and precious guidance and for having amicable relation.

With deep sense of gratitude, we extend our earnest and sincere thanks to our SPOC **Mr. T. Karthikeyan,** Assistant Professor, Department of Computer Science and Engineering for his guidance and encouragement during this project.

We would also like express our thanks to all the faculty members of our department, friends and students who helped us directly and indirectly in all aspects of the project work to get completed successfully.

**TABLE OF CONTENTS**

|  |  |  |  |
| --- | --- | --- | --- |
| **CHAPTER**  **NO.** | **TITLE** | | **PAGE**  **NO.** |
| **1** | **INTRODUCTION** | |  |
|  | 1.1 | Project Overview |  |
|  | 1.2 | Purpose |  |
| **2** | **LITERATURE SURVEY** | |  |
|  | 2.1 | Existing problem |  |
|  | 2.2 | References |  |
|  | 2.3 | Problem Statement Definition |  |
| **3** | **IDEATION & PROPOSED SOLUTION** | |  |
|  | 3.1 | Empathy Map Canvas |  |
|  | 3.2 | Ideation and Brainstorming |  |
|  | 3.3 | Proposed Solution |  |
|  | 3.4 | Problem Solution Fit |  |
| **4** | **REQUIREMENT ANALYSIS** | |  |
|  | 4.1 | Functional requirements |  |
|  | 4.2 | Non – Functional Requirement |  |
| **5** | **PROJECT DESIGN** | |  |
|  | 5.1 | Data Flow Diagrams |  |
|  | 5.2 | Solution and Technical Architecture |  |
|  | 5.3 | User Stories |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **6** | **PROJECT PLANNING & SCHEDULING** | |  |
|  | 6.1 | Sprint Planning & Estimation |  |
|  | 6.2 | Sprint Delivery Schedule |  |
|  | 6.3 | Report from JIRA |  |
| **7** | **CODING & SOLUTIONING** | |  |
|  | 7.1 | Feature 1 |  |
|  | 7.2 | Feature 2 |  |
| **8** | **SYSTEM TESTING** | |  |
|  | 8.1 | Test Cases |  |
|  | 8.2 | User Acceptance Testing |  |
| **9** | **RESULTS** | |  |
|  | 9.1 | Performance Metrics |  |
| **10** | **ADVANTAGES & DISADVANTAGES** | |  |
| **11** | **CONCLUSION** | |  |
| **12** | **FUTURE SCOPE** | |  |
| **13** | **APPENDIX** | |  |
|  | 13.1 | Source Code |  |
|  | 13.2 | Screenshots |  |

**INTRODUCTION**

**CHAPTER – 1**

**INTRODUCTION**

* 1. **PROJECT OVERVIEW:**

Forest fires is an big environmental issues, for an year destroys around 1L trees, 90% of the pollution is from humans activities which can't be stopped, but we can reduce the rest of the 10% of the pollution and stop it as soon as possible at early stage. Forest fires are not an easy talks to taken into fact its a big tedious process, so we have brought up an idea of an camero. Or a video frame approach. Where we feed the video / frame into the project to find the forest fire is present or not in early stages. If we are to observe an area we place cameras covers each corners and areas of that area covering different grounds at different lengths. We use deep learning model, that helps in prediction of the fire which we train it with datasets using previous examples of fires and non fires to make the model understand the difference between them and helps in predicting the fire. This camera takes footage and feeds the video frames into the model where the model predicts the forest fires with the previously trained knowledge of dataset and compares them, this model can easily classify forest with fire and without, with accuracy increasing by each prediction done. When a video is split into frames multiple frames are broken down which then frame by frames are implemented in the model. Once the fire is detected the model gives us an alert, where the twilio account is used for the alert messages where all the alerts are shown to the register phone number. So this alert messages are sent to the authorities to alert them about the forest fire. This is how our project and model works.

* 1. **PURPOSE:**

To fight forest fires, different solutions were employed throughout the years. They ware primary aimed at the early detection of the fires. The simplest of these solutions is the establishment of a network of observation posts - both cheap and easy to accomplish, but also time-consuming for the involved people.

**LITERATURE SURVEY**

**CHAPTER – 2**

**LITERATURE SURVEY**

**2.1 EXISTING PROBLEM:**

Forest fires are a major environmental issue, creating economic and ecological damage while endangering human lives. There are typically about 100,000 wildfires in the United States every year. Over 9 million acres of land have been destroyed due to treacherous wildfires. It is difficult to predict and detect Forest Fire in a sparsely populated forest area and 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.

**2.2 REFERENCES:**

**1.OPTIMUM SENSORS ALLOCATION FOR A FOREST FIRES MONITORING SYSTEM [Beatriz Flamia Azevedo, Thadeu Brito and Jose Lima 2021]**

Forest fire is a huge issue to tackle this issue we’ve developed a project that can monitor forest fire. Modules of wireless sensor will be spread in the forest to collect data about humidity, temperature through sensors. optimum position is allocated to each sensor. Wireless sensor network capable of monitoring large areas. The sensor measures physical parameters, such as changes in barometric pressure, solar radiation, and chemical parameters, such as carbon dioxide. Two data mining methods Random forest and SVM, were used to produce forest fire maps. The data is mostly collected by sensors. The position of these sensors can determine the quality of the data and the accuracy of the system. Thus, defining the optimal position of each sensor by optimisation tools can provide substantial improvements to the system. The sensors do face drawbacks like available sensors, fire hazard, and the forest density. The fire is categorized by a scale 0 to 5. The forest density is scaled from 0 to 100. Values are obtained from european satellite. If some region is important and valued it’ll have a higher fire hazardous value. A sensor is capable of retrieving information from 360 degree. Sometimes the sensor fail to detect the forest fire due to high density of the forest adding few more sensors can work well. The fire is distinguished by red and blue colors. 4 cases are used to detect a forest fire these cases consist of parameters from coordinates and fire range by using mathematical formulas we derive if the forest on observation is in fact on fire. We use LoraWan technology since it doesn’t have monthly costs and they have batteries that can last upto 6 months. Stochastic method is executed more than once which then give the value. A general analysis is done and a graph is produced which determines the rate of forest fire acceleration. A alert is then issued when fire is detected . The goal is making sure that forest fire crisis is reset to 0 soon. Since this model focuses on sensors which work on range from 0 to 100m.We won’t be able to allocate that number of sensors in such forest regions.

**2.EARLY DETECTION OF FOREST FIRE USING MIXED LEARNING TECHNIQUES AND UAV [Varanasi LVSKB Kasyap, D. Sumathi and Pradeep Reddy CH 2022]**

This model doesn't need any human observation and monitoring, It uses satellite images to monitor to observe forest locations. Wireless sensor networks are implemented to observe fire events WSN are deployed this makes sovereignty which cost less to deploy. They are autonomous and they use UAVs to detect fire and manage them , it also observes fire in the lower region Detections are done through remote sensing methods like satellite, high resolution camera on ground and aerial observation, the images received are not often clear sometimes the forest is put through heavy weather conditions. Optical and thermal cameras are deployed in addition with sensors that can detect smoke, temperature and humidity. Aerial observation is done using UAV, the recognition rate is up to 83 %, it uses datasets that classifies images as forests that are on fire and that are not. In this model is trained using ML, it uses CNN. This model also analyses temperature and wind speed of the area in observation. Detection of forest fire using UAVs are much better compared to any other methods, it can detect fires from above and the range of fire. We can easily develop a 3D model for the scene in capture, if the UAV fails to detect fire because of fog / smoke. Ground surveillance cameras and sensors are put to use to detect fire and analyses is made. In future we will explore more possibilities that doesn’t stop or interfere the detection system.

**3. FOREST 4.0: DIGITALIZATION OF FOREST USING THE INTERNET OF THINGS [Rajesh Singh, Anita Gehlot and Shaik Vaseem Akram 2021]**

Forest is very important for the thriving of the earth. Forest helps with carbon emission and climate change. A survey of wildlife and the forest is presented. The significance of the internet for wildlife monitoring is addressed with recent trends. A detailed suggestion was presented for enhancement of connectivity, implementation of realtime sensing. With the help of IOT and components we established multiple communications with devices through IP. With the use of classical sensors, we measure environmental variables. We use sensors to gather signals and transform them into digital information’s and further work on them. There are 4 cases in digitizing, human to forest interaction, big data and cloud computing. To handle the huge data receiving, and cloud computing platform is used for affordable services, the data and then analysed then we use AI to observe and learn and make autonomous choices. We use internet based modules to communicate, monitor and track with the help of IP. We use Lora and edge gateway for monitoring, Lora has low power consumption, it’s capable of performing analysis. Forest is spatially mapped using GIS and remote sensing tech. It uses RF long range for communication. It uses IR sensor to detect temperature humidity. It uses parameters from Environmental sensors and communicates with nearby IOT devices, It uses RF ID reader and Lora communication and internet connectivity. The information is then digitally stored in cloud server’s. Cloud server enables data sharing and real-time remote location sharing, GSM is implemented for transmission of sensory data. We can also use instant detect monitoring system for observation of fauna. The integration of IOT in wildlife & forest, to protect and conserve them by real-time sensing CV node, DL, ML and sustainable deployment of sensors.

**4. IOT ENABLED ADVANCED FOREST FIRE DETECTING AND MONITORING ON UBIDOTS PLATFORM [P. Kanakaraja, P. Syam Sundar and S. Gopal Krishna Reddy 2021]**

The use of IoT devices to ensure the forest doesn’t burn to nothing. We integrate the Smoke detection is the most important aspect of forest fire detection the smoke sensor will detect and the day-to-day value will be updated in IOT dashboard we are setting a threshold level for the smoke detection the alert will be on only when the level of smoke exceeds the given threshold value. PIR sensor for measuring the fall in humidity and temperature we can also detect movements with this sensor. Wireless Sensor Networks for Forest Fire Detection. We use different types of sensors like rain sensor which detects rainfall in the forest, sound sensor which detects any type of noise, Dht11 sensor which detects the temperature and humidity of the surroundings in the forest and we use PIR sensor which detects the motion of the bodies in the forest. Along with these we also use which is used to monitor the situation or to take images of the surroundings. By using UBI dotsIoT (platform) it reads the data from sensors and filters the message and then it converts the data and validate it. If the data is valid, it uploads the data to UBI dots IoT dashboard and alerts it is repeated process. The data is made readable and the result is displayed on the screen real time on an LCD Screen for better analysis of the data and interpreting the information. If the sensor detects a hike in temperature and PIR sensor detects lot of movements from the surroundings a fire alert is issued immediately. This also doesn’t just stop at detection of forest fires this works on preventing and monitoring wildlife as well.

## 2.3 PROBLEM STATEMENT DEFINITION:

## Forest fire is a major issue across the world and it destroys many rare flora and faunas. Lots of rare and most important species is destroyed due to Forest fire. To stop this we developed a model called forest fire detection, This proposed model is able to detect forest fires before it begins or at the early stages of forest fire. So it can be stopped easily without any trouble. The prediction process is based on Training the model so it can distinguish between Forests that are on fire and forests that aren’t using machine learning algorithm to train the model. It uses multiple layers to learn from the image that can help distinguish fires. Uses layer like Convolutional Neural Network (CNN). If the model detects a increase in the safety limits set by the user from the dataset images. The prediction will alert the user a fire is detected.

## IDEATION AND PROPOSED SOLUTION

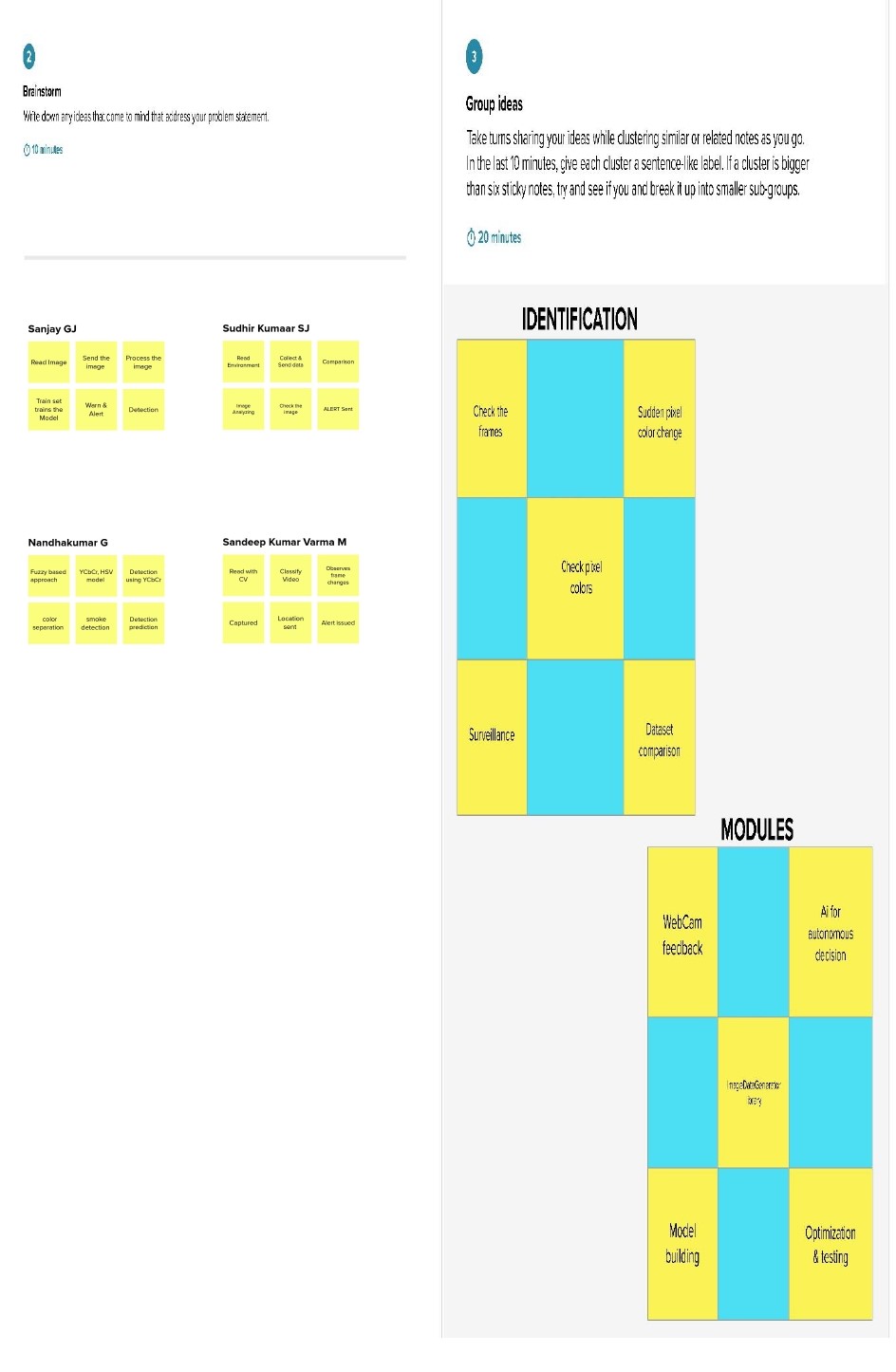
**CHAPTER – 3**

**IDEATION AND PROPOSED SOLUTION**

## EMPATHY MAP CANVAS:

## 

**3.2 IDEATION AND BRAINSTORMING:**

****

****

**3.3 PROPOSED SOLUTION:**

|  |  |  |
| --- | --- | --- |
| **S. No** | **Parameter** | **Description** |
| 1. | Problem Statement (Problem to be solved) | To Predict the Forest fire at early stage and send an alert. |
| 2. | Idea / Solution description | Forest fires are a big issue, every year millions of trees burn down due to forest fires. To tackle this, A model which can detect forest fires by using the user's video feed. It converts the user's video into frames and sends it through the model. This model is already trained enough to detect and distinguish between fire and no fire the model's learning process is configured by Machine learning. After the frames are sent to the model, it goes through multiple layers that help identify the model if there is a fire in the frame. If detected an alert is issued  immediately. |
| 3. | Novelty / Uniqueness | Prediction is completely autonomous and rate of prediction and analyzation is much efficient. It is completely economical. |
| 4. | Social Impact / Customer Satisfaction | Using this model can help prevent forest fires and it can help keep the forest under surveillance by also detecting potential forest fires. An alert can be issued and everyone can be evacuated safely. |
| 5. | Business Model (Revenue Model) | * Key Partners-Forest Authorities and forest fire department. * Activities- Consistent monitoring of the forest through cameras. * Key resources - Maintenance of the technology. * Value preposition-low price. * Cost structure- Technology platform. |
| 6. | Scalability of the Solution | This model can be accessed by all authorities in need of information about the forest fires. |

## 3.4 PROBLEM SOLUTION FIT:

## 

**REQUIREMENT ANALYSIS**

**CHAPTER – 4**

**REQUIREMENT ANALYSIS**

## FUNCTIONAL REQUIREMENT:

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Functional Requirement (Epic)** | **Sub Requirement (Story / Sub-Task)** |
| FR-1 | User Registration | Registration through Gmail |
| FR-2 | User Confirmation | Confirmation via Email  Confirmation via OTP |
| FR-3 | User Login | Login using credentials |
| FR-4 | User Search | Search for Info on forest fire occurrence |
| FR-5 | User Profile | User shall be given a live feed of the forest |
| FR-6 | User Application | User is alerted if there is an forest fire occurrence in  their surroundings |

* 1. **NON - FUNCTIONAL REQUIREMENT:**

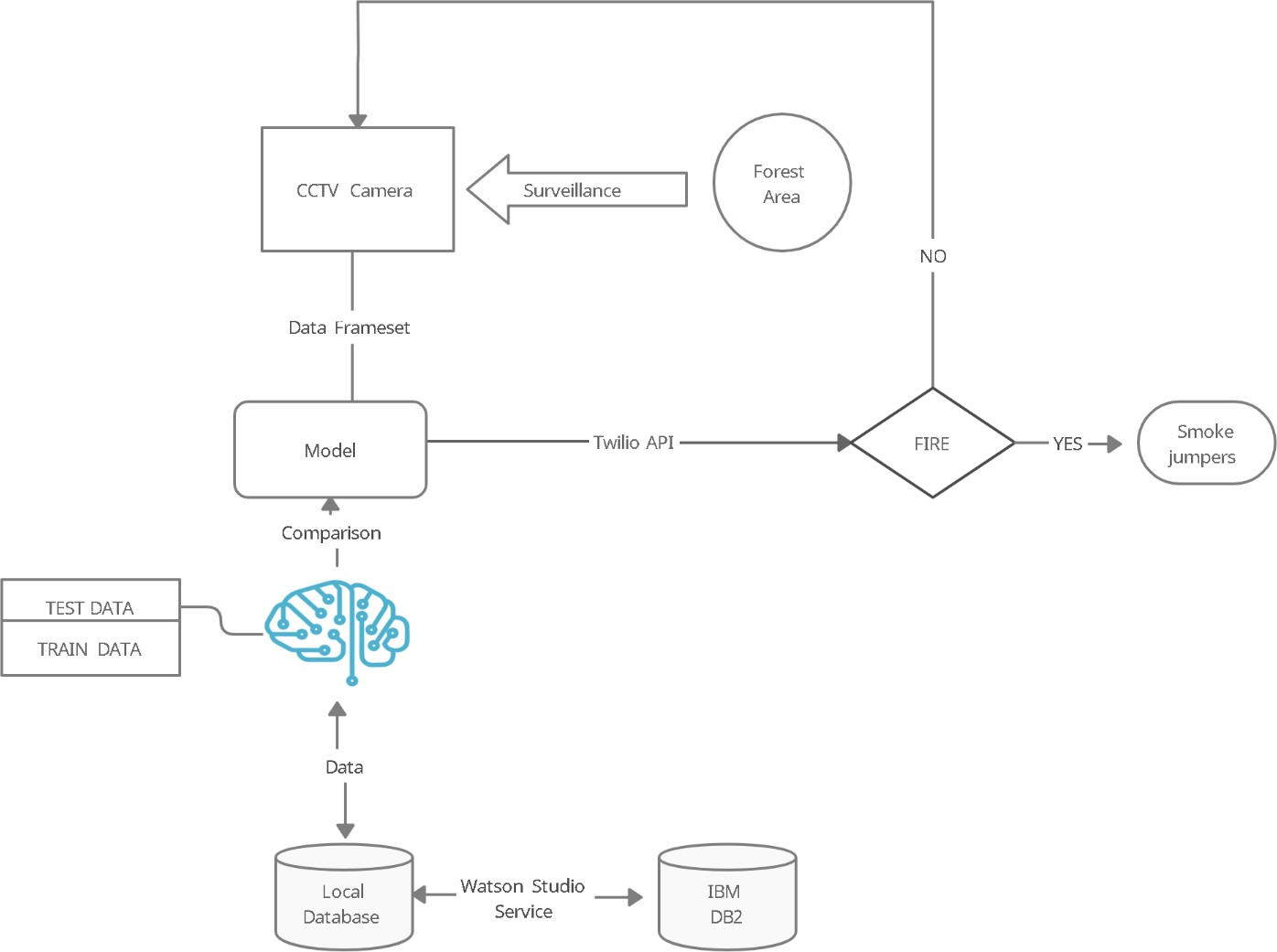
|  |  |  |
| --- | --- | --- |
| **FR No.** | **Non-Functional Requirement** | **Description** |
| NFR-1 | **Usability** | Alerts according to the user location |
| NFR-2 | **Security** | Instant live feed with alert of the situation |
| NFR-3 | **Reliability** | The prediction of the forest fire is 87% accurate |
| NFR-4 | **Performance** | The feed and the alert message is an immediate  action without a lag |
| NFR-5 | **Availability** | The application gives alerts and live feeds 24/7 |
| NFR-6 | **Scalability** | Early detection and alerting users are done  efficiently and in a faster means |

**PROJECT DESIGN**

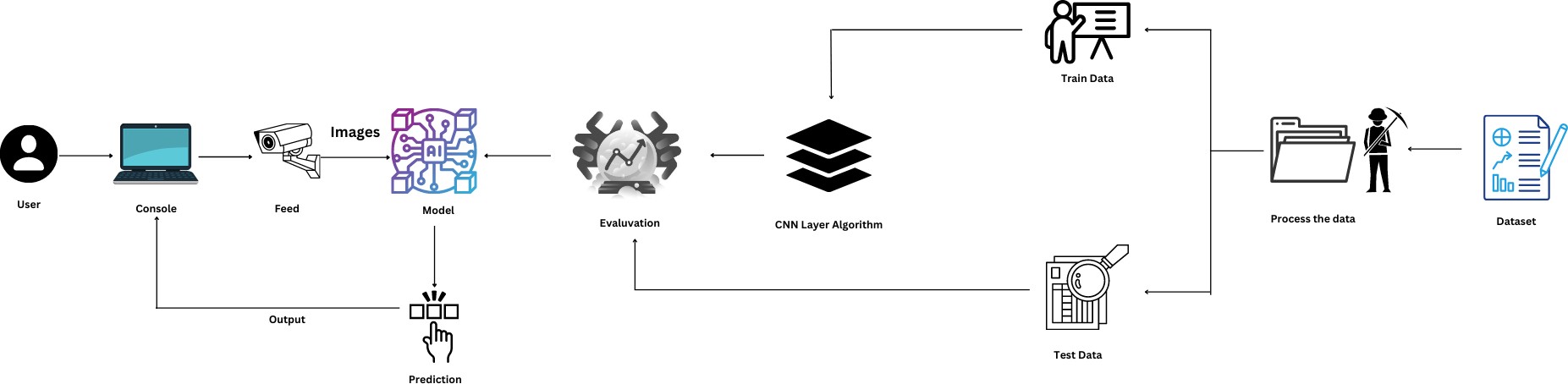
**CHAPTER – 5**

**PROJECT DESIGN**

**5.1 DATA FLOW DIAGRAMS:**



**5.2 SOLUTION AND TECHNICAL ARCHITECTURE:**



**5.3 USER STORIES:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **User Type** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Acceptance Criteria** | **Priority** | **Release** |
| CCTV | Capturing Images | USN-1 | It will proctor the forest area, 24/7, send information contiguously and it is automated by model | I can capture the image of the forest | low | Sprint-1 |
| USN-2 | The proctored image is converted into frameset simultaneously | Image converted using API | High | Sprint-1 |
| USN-3 | The frameset is sent to the model for comparison | Frameset is accepted in pixels | Low | Sprint-2 |
| MODEL | Deep Learning | USN-4 | The model compares the received frameset with the trained dataset | It compares with trained image | Medium | Sprint-1 |
| USN-5 | It is trained with enough data with the use of different images of fire | It can access the Model | High | Sprint-1 |
| USN-6 | The compared image is checked for the intensity of the fire, if is it in smoke the Model | Intensity is denoted in YBR | High | Sprint-2 |
| Display Alert | USN-7 | If the intensity of the fire is high, it alerts with high pitched sound | Application notice fire | High | Sprint-2 |
| Twilio API | USN-8 | The alert message is displayed with the use of Twilio API | Warns on the monitor | Medium | Sprint-2 |
| ADMIN | Surveillance | USN-9 | As an admin, I should be keen at proctoring for alert | The API sends the alert | High | Sprint-1 |
| Send Confirmation | USN-10 | Confirmation of Fire alert is sent to the smokejumpers | Confirmation is received by the Smokejumper | High | Sprint-2 |
| Manage Database | USN-11 | As an admin, I must make data base maintained | IBM Watson studio service is used | Low | Sprint-2 |

**PROJECT PLANNING AND SCHEDULING**

**CHAPTER – 6**

**PROJECT PLANNING AND SCHEDULING**

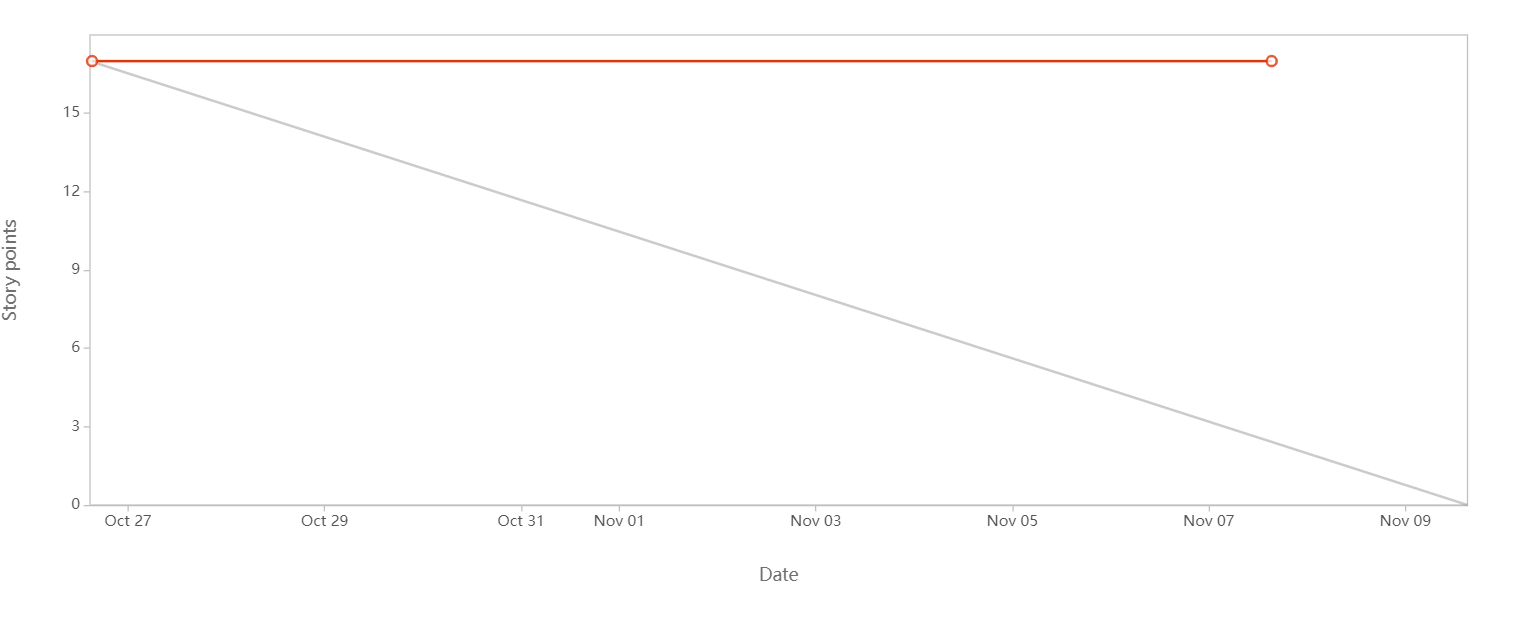
## SPRINT PLANNING AND ESTIMTION:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | **Team Members** |
| Sprint-1 | Registration | USN-1 | As a user, I can register for the application by entering my email, password, and confirming my password. | 5 | High | Sanjay GJ |
| Sprint-1 |  | USN-2 | As a user, I can register for the application through Mobile number. | 3 | Low | Sudhir Kumaar SJ |
| Sprint-1 |  | USN-3 | As a user, I can register for the application through Gmail | 4 | Medium | Sanjay GJ |
| Sprint-1 | Login | USN-4 | As a user, I can log into the application by entering email & password | 5 | High | Sandeep Kumar Varma M |
|  |
| Sprint-2 | Dashboard | USN-5 | As a user ,I can check my account details on the dashboard | 1 | Low | Nandhakumar G |
| Sprint-2 | Providing Images | USN-6 | As a user, I should check for forest fires by either providing images or Camera. | 5 | High | Sudhir kumaar SJ |
| Sprint-4 | Prediction | USN-7 | As a user, I can view whether the forest is on fire or not. | 5 | High | Nandhakumar G  Sandeep Kumar Varma M |
|  |
| Sprint-3 | Data Pre-Processing | USN-8 | As a admin, I should clean and pre-process data using pandas. | 5 | High | Sanjay GJ Nandhakuma r G Sudhir Kumaar SJ |
| Sprint-3 |  | USN-9 | As a admin, I should train and test the dataset using sklearn. | 5 | High | Sanjay GJ  Sandeep Kumar Varma M |
|  |
| Sprint-3 | Model Building | USN-10 | As an admin, I should predict the accuracy of data using supervised machine learning. | 12 | High | Sanjay GJ Sandeep Kumar  Varma M |
|  |
| Sprint-4 | API | USN-11 | As an admin, I should connect the results and display it to the console or display using py flask | 10 | High | Sanjay GJ Nandhakumar G  Sandeep Kumar Varma M  Sudhir Kumaar SJ |
|  |
|  |
|  |
| Sprint-4 | Notification | USN-12 | As an admin, I should send the prediction Information | 5 | Medium | Sanjay GJ Sandeep Kumar Varma M  Sudhir Kumaar SJ |

* 1. **SPRINT DELIVERY SCHEDULE:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Total Story Points** | **Duration** | **Sprint Start Date** | **Sprint End Date (Planned)** | **Story Points** | **Sprint Release Date (Actual)** |
| **Completed (as on Planned End Date)** |
| Sprint-1 | 17 | 8 Days | 26-Oct-22 | 03-Nov-22 |  |  |
| Sprint-2 | 11 | 4 Days | 05-Nov-22 | 08-Nov-22 |  |  |
| Sprint-3 | 22 | 5 Days | 09-Nov-22 | 14-Nov-22 |  |  |
| Sprint-4 | 15 | 4 Days | 16-Nov-22 | 20-Nov-22 |  |  |

* 1. **REPORT FROM JIRA:**

**SPRINT:1**

## SPRINT:2

## 

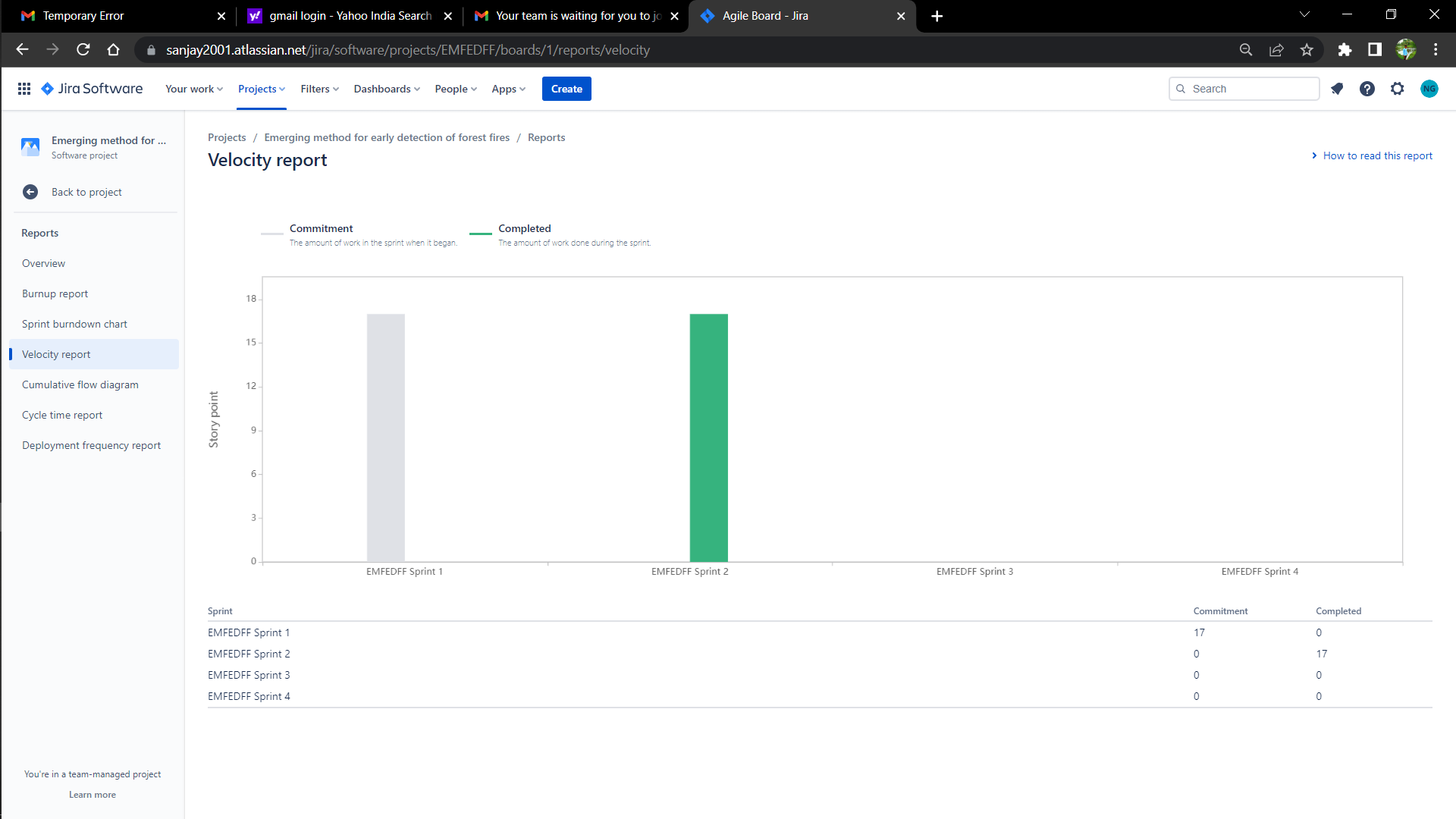
## SPRINT:3

## 

## SPRINT:4

## 

# **VELOCITY:**



## CODING & SOLUTIONING

## CHAPTER – 7

## CODING & SOLUTIONING

## 7.1 FEATURE 1:

## HTML is the basic markup language used to build the structure of the website. CSS stands for Cascading Style Sheets. CSS describes how HTML elements are displayed on screen, paper, or in other media. CSS saves a lot of work. It can control the layout of multiple web pages all at once. By using HTML and CSS we can able to show the dashboard, story and report of the project in a very effective manner as a web application.

## 

## 7.1 FEATURE 2:

## Python is a general-purpose high level programming language that is widely used in data analytics. It is a valuable part of the data analyst’s toolbox, as it’s tailor-made for carrying out repetitive tasks and data manipulation, and anyone who has worked with large amounts of data knows just how often repetition enters it. By having a tool that handles the grunt work, the data analysts are free to handle the more interesting and rewarding parts of the job. Data analysts should also keep in mind the wide variety of other Python libraries available out there. These libraries, such as NumPy, Pandas, and Matplotlib, help the data analyst carry out their functions.

## 

**TESTING**

**CHAPTER – 8**

**TESTING**

* 1. **TEST CASES:**

**Test Cases for Registration Page**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Cases** | **Feature** | **Description** | **Steps to Execute** | **Expected Results** |
| TC-001 | User Interface | Check all textboxes, checkboxes and buttons | 1.Click textboxes, checkboxes and buttons | UI should work properly |
| TC-002 | Required fields | Check the required fields by not filling any data | 1. Do not enter any value in the field.  2. Click on the Register button. | A required field message should be displayed |
| TC-003 | Required fields | Check if the user is registered by filling all the required fields | 1. Enter valid values in the required fields.  2. Click the Register button. | 1. Users should be registered successfully |
| TC-004 | Required fields | Check if password and confirm password are same | 1.Enter different passwords for Password and Confirm Password fields | It should display a message saying that the passwords don’t match |
| TC-005 | Email validation | Check if the email is valid | 1. Enter Invalid Emails  2. Click on the Register Button. | It should show an invalid email message |
| TC-006 | Email validation | Check all the valid emails | 1.Enter Valid Email  2.Click on the Register Button | It should not show any message |
| TC-007 | Email validation | Check if Email already exists in the database. | 1.Enter an already registered email.  2.Click Register button | It should say that email already exists |

**Test Cases for Login Page**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Cases** | **Feature** | **Description** | **Steps to Execute** | **Expected Results** |
| TC-001 | User Interface | Check all textboxes, checkboxes and buttons | 1.Click textboxes, checkboxes and buttons | UI should work properly |
| TC-002 | Required fields | Check the required fields by not filling any data | 1. Do not enter any value in the field.  2. Click on the Login button. | A required field message should be displayed |
| TC-003 | Required fields | Check user should by filling all the required fields | 1. Enter valid values in the required fields.  2. Click the Login button. | 1. Users should be logged in successfully  2. User should be redirected to home page |
| TC-004 | Email validation | Check if the email is valid | 1. Enter Invalid Emails  2. Click on the Login Button. | It should show an invalid email message |
| TC-005 | Required fields | Check if Password is valid | 1.Enter Invalid password  2.Click on the Login button | It should show invalid password message |

**Test Cases for Home Page**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Cases** | **Feature** | **Description** | **Steps to Execute** | **Expected Results** |
| TC-001 | User Interface | Check all textboxes, checkboxes and buttons | 1.Click textboxes, checkboxes and buttons | UI should work properly |
| TC-002 | User Interface | Check the About us button | 1. Check any contacts available for Help | A required contacts will be hovered |
| TC-003 | FAQ | Check user should be able to chat in Whatsapp | 1. Enter a query and send it  To the admin | 1. Users successfully sent the Query  2. User should be redirected to Whatsapp page |
| TC-004 |  | Check user should be able to use Email | 1. Click on the Email button | 1. It should show an Email interface  2. It should send the mail |

**Test Cases for Upload Page**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Cases** | **Feature** | **Description** | **Steps to Execute** | **Expected Results** |
| TC-001 | User Interface | Check all textboxes, checkboxes and buttons | 1.Click textboxes, checkboxes and buttons | UI should work properly |
| TC-002 | Required fields | Check the required fields by not filling any data | 1. Do not enter any value in the field.  2. Click on the Apply button. | A required field message should be displayed |
| TC-003 | Required fields | Check user should by filling all the required fields | 1. Upload a valid file  2. Click the Apply button. | 1. User successfully upload the file and the fire should be predicted  2. User should be redirected to prediction page |
| TC-004 |  | It will find the local storage to upload file | 1. Select the file and upload the file | Check whether the user selected files is uploaded |

## USER ACCEPTANCE TESTING:

## 1. Defect Analysis:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Resolution | Severity 1 | Severity2 | Severity3 | Severity4 | Sub total |
| By display | 6 | 4 | 2 | 7 | 19 |
| Duplicate | 1 | 0 | 3 | 0 | 4 |
| External | 2 | 3 | 0 | 1 | 6 |
| fixed | 20 | 10 | 5 | 26 | 61 |
| Not Reproduced | 0 | 0 | 1 | 0 | 1 |
| Skipped | 0 | 0 | 1 | 1 | 2 |
| Won’t fix | 0 | 0 | 0 | 0 | 0 |
| Totals | 29 | 17 | 12 | 35 | 93 |

## 2. Test case Analysis:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Section | Total cases | Not Tested | Fail | Pass |
| Fire Detection | 8 | 0 | 2 | 6 |
| Security Alert | 52 | 0 | 0 | 52 |
| Client application | 2 | 0 | 0 | 2 |
| Final Report Output | 5 | 0 | 2 | 3 |
| Version control | 9 | 0 | 0 | 9 |

## RESULTS

## CHAPTER – 9

## RESULTS

## 9.1 PERFORMANCE METRICS:

## 

|  |  |  |  |
| --- | --- | --- | --- |
| S.No | **Parameters** | **Values** | **ScreenShots** |
| 1. | ModelSummary | 92 |  |
| 2. | Accuracy | Training\_Accuracy=90Validation\_Accuracy=93 |  |

**ADVANTAGES & DISADVANTAGES**

**CHAPTER – 10**

**ADVANTAGES & DISADVANTAGES**

**ADVANTAGES:**

* Everything is automanted and doesn't need any human invloved in it.
* Instant detection of the forest fire with quick alerts.
* Prediction of forest fire rate is upto 93% and is increasing each time of newer predictions.
* Can prevent the forest fire ag early stages, giving us an upper hand on controling it earlier and faster on a large scale.
* Alert are sent st an instant making it an more reliable application.

**DISADVANTAGES:**

* Sometimes a technical service checkup often is often needed to see if it runs properly.
* If predicted wrong might burn with fires eventually getting destroyed and rebuild cost is much expensive.
* Due to smoke by fires right beneath the cameras or fog inside cameras can affect the vision of the camera hence dropping every action to zero.

**CONCLUSION**

**CHAPTER – 11**

**CONCLUSION**

The model for early forest fire detection is still in its improvement stage. The model can be upgraded with sensors smoke detectors gas emission equipment, The economical price of the sensor and camera is a challenge so discussed the actual implementation. We have performed a thorough research and some simulation experiments and we believe that we can make improvements in the future that can make this model much efficient and reliable. To keep the model updated we apply adequate and necessary steps and approach that is also up-to-date with fast prediction and quick alerting. For the benefit of society and well-being of the ecosystem this model could really work and predict fire without any such delay. Further more improving this model is the efficient strategy comparing other prediction models. The Deep learning prediction model was way more faster when tested.

**FUTURE SCOPE**

**CHAPTER – 12**

**FUTURE SCOPE**

The Model that is developed to predict fire can be improved in such a way it can make accurate predictions the model can be trained to make more accurate predictions. Sensors can be added in addition to the video camera. Sensors which are able to detect smoke, intensity of fire and rate of gas emission. Since forests are being depleted day by day these features can be added in with the model to ensure the forests thrive and don't burn down.

**APPENDIX**

**CHAPTER – 12**

**APPENDIX**

**13.1 SOURCE CODE:**

**HTML:**

**HOME**

<!DOCTYPE html>

<html>

<head>

<title>Emerging methods of early forest fire detection</title>

<link rel="stylesheet" href="{{url\_for('static',filename='home.css')}}">

<script src="https://kit.fontaweome.com/a9365d9272.js" crossorigin="anonymous"></script>

</head>

<body>

<header><h2> Emerging Methods for Early Detection of Forest Fires</h2></header>

<nav>

<div class="container">

<ul class="dropdown">

<li><a href="#">Home</a></li>

<li><a href="#">About Us &dtrif;</a>

<ul class="Aboutus">

<li><a href="//wa.me/918056867362">Whatsapp</a></li>

<li><a href="mailto:2k19cse081@kiot.ac.in">E-mail</a></li>

</li> </ul>

<li><a href="/login.html">Login</a></li>

<li><a href="/sign-up.html">sign-up</a></li>

</ul></div></nav>

<div>

<h1> Forest fire</h1>

<p>Forest fires area major environmental issue, creating economic and ecological damage while endangering human lives. There are typically about 100,000 wildfires in the United States every year. Over 9 million acres of land have been destroyed due to treacherous wildfires. It is difficult to predict and detect Forest Fire in a sparsely populated forest area and 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.</p>

</div>

<div>

<a href="{{url\_for('upload')}}" class="button">predict fire</a>

</div>

</body>

</html>

**LOGIN:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Login Register Form</title>

<link rel="stylesheet" href="{{url\_for('static',filename='login.css')}}">

</head>

<body>

<div class="container">

<div class="login-register">

<div class="nav-buttons">

<button id="loginBtn" class='active' >Login </button>

<button id="registerBtn">Register</button>

</div>

<div class="form-group">

{{msg}}

<form action="/signin" id="login", method="POST">

<label for="username">username</label>

<input type="text" id="username" name="fullname">

<label for="password">password</label>

<input type="text" id="password" name="password">

<input type="submit" value="Submit" class="submit">

</form>

<form action="/register" id="register", method="POST">

<label for="fullname" name="fullname">fullname</label>

<input type="text" id="fullname" name="fullname">

<label for="email">email</label>

<input type="email" id="email" name="email">

<label for="passwword">password</label>

<input type="text" id="password" name="password">

<label for="confirmpassword">confirm password</label>

<input type="text" id="confirmpassword" name="confirmpassword">

<input type="submit" value="Submit" class="submit">

</form>

</div>

<div id="forgot">

<a href="">forgot password?</a>

</div>

</div>

</div>

<script>

var loginBtn = document.getElementById("loginBtn");

var registerBtn = document.getElementById("registerBtn");

var loginform = document.getElementById("login");

var registerform = document.getElementById("register");

var forgot = document.getElementById("forgot");

registerBtn.onclick= function() {

registerform.style.left='0px';

registerform.style.opacity='1';

loginform.style.left='-500px';

loginform.style.opacity='0';

forgot.style.left='-500px';

forgot.style.opacity='0';

registerBtn.classList.add('active');

loginBtn.classList.remove('active');

}

loginBtn.onclick= function() {

loginform.style.left='0px';

loginform.style.opacity='1';

forgot.style.left='0px';

forgot.style.opacity='1';

registerform.style='500px';

loginBtn.classList.add('active');

registerBtn.classList.remove('active');

registerform.style.opacity='0';

}

</script>

</body>

</html>

**UPLOAD:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Login Register Form</title>

<link rel="stylesheet" href="{{url\_for('static',filename='upload.css')}}">

</head>

<body>

<div class="drag-area">

<div class="icon"><i class="fas fa-cloud-upload-alt"></i></div>

<header>Drag & Drop to Upload File</header>

<span>OR</span>

<button>Browse File</button>

<input type="file" hidden>

</div>

<script>

//selecting all required elements

const dropArea = document.querySelector(".drag-area"),

dragText = dropArea.querySelector("header"),

button = dropArea.querySelector("button"),

input = dropArea.querySelector("input");

let file; //this is a global variable and we'll use it inside multiple functions

button.onclick = ()=>{

input.click(); //if user click on the button then the input also clicked

}

input.addEventListener("change", function(){

//getting user select file and [0] this means if user select multiple files then we'll select only the first one

file = this.files[0];

dropArea.classList.add("active");

showFile(); //calling function

});

//If user Drag File Over DropArea

dropArea.addEventListener("dragover", (event)=>{

event.preventDefault(); //preventing from default behaviour

dropArea.classList.add("active");

dragText.textContent = "Release to Upload File";

});

//If user leave dragged File from DropArea

dropArea.addEventListener("dragleave", ()=>{

dropArea.classList.remove("active");

dragText.textContent = "Drag & Drop to Upload File";

});

//If user drop File on DropArea

dropArea.addEventListener("drop", (event)=>{

event.preventDefault(); //preventing from default behaviour

//getting user select file and [0] this means if user select multiple files then we'll select only the first one

file = event.dataTransfer.files[0];

showFile(); //calling function

});

function showFile(){

let fileType = file.type; //getting selected file type

let validExtensions = ["image/jpeg", "image/jpg", "image/mp4"]; //adding some valid image extensions in array

if(validExtensions.includes(fileType)){ //if user selected file is an image file

let fileReader = new FileReader(); //creating new FileReader object

fileReader.onload = ()=>{

let fileURL = fileReader.result; //passing user file source in fileURL variable

let imgTag = `<img src="${fileURL}" alt="image">`; //creating an img tag and passing user selected file source inside src

attribute

dropArea.innerHTML = imgTag; //adding that created img tag inside dropArea container

}

fileReader.readAsDataURL(file);

}else{

alert("This is not an Image File!");

dropArea.classList.remove("active");

dragText.textContent = "Drag & Drop to Upload File";

}

}

</script>

</body>

</html>

**DATABASE:**

from turtle import st

from flask import Flask, render\_template, request, redirect, url\_for, session

from markupsafe import escape

import ibm\_db

try:

conn = ibm\_db.connect("DATABASE=bludb;HOSTNAME=824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=30119;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=pxj01049;PWD=gHDRlsYaudgJIF2y",'','')

print("Successfully connected with db2")

except:

print("Sorry.. Unable to connect with Database: ", ibm\_db.conn\_errormsg())

app = Flask(\_\_name\_\_)

# Home page open aagum

@app.route('/')

def home():

return render\_template('login.html')

@app.route("/index")

def index():

return render\_template('home.html')

@app.route("/upload")

def upload():

return render\_template('upload.html')

# @app.route('/login')

# def login():

# return render\_template('login.html')

# @app.route('/register')

# def register():

# return render\_template('login.html')

# register oda submit action

@app.route('/register',methods = ['POST','GET'])

def register():

if request.method == 'POST':

print("--------------------------------")

name = request.form['fullname']

email = request.form['email']

password = request.form['password']

cpassword = request.form['confirmpassword']

sql = "SELECT \* FROM user WHERE email =?"

stmt = ibm\_db.prepare(conn, sql)

ibm\_db.bind\_param(stmt,1,email)

ibm\_db.execute(stmt)

account = ibm\_db.fetch\_assoc(stmt)

print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

print(account)

if account:

return render\_template('login.html', msg="You are already a member, please login using your details")

else:

insert\_sql = "INSERT INTO user VALUES (?,?,?)"

prep\_stmt = ibm\_db.prepare(conn, insert\_sql)

ibm\_db.bind\_param(prep\_stmt, 1, name)

ibm\_db.bind\_param(prep\_stmt, 2, email)

ibm\_db.bind\_param(prep\_stmt, 3, password)

ibm\_db.execute(prep\_stmt)

print("Data Stored")

sql = "SELECT \* FROM user WHERE name = ?"

stmt = ibm\_db.prepare(conn, sql)

ibm\_db.bind\_param(stmt, 1, name)

ibm\_db.execute(stmt)

account = ibm\_db.fetch\_assoc(stmt)

print("$$$$$$$Fetched$$$$$$$$$")

print(account)

return render\_template('login.html', msg="user Data saved successfuly..")

@app.route('/signin', methods=['POST','GET'])

def signin():

if request.method == 'POST':

name = request.form.get("fullname")

password = request.form.get("password")

print("Trying to Login in")

sql = "SELECT \* FROM user WHERE name = ?"

stmt = ibm\_db.prepare(conn, sql)

ibm\_db.bind\_param(stmt, 1, name)

ibm\_db.execute(stmt)

account = ibm\_db.fetch\_assoc(stmt)

print("$$$$$$$$$$$$$$$$$")

print(account)

if not account:

return render\_template('login.html', msg="You not yet registered")

else:

if(password == account['PASSWORD']):

print("Password Matched")

return redirect(url\_for('index'))

else:

return render\_template('login.html', msg = "Password Incorrect")

if '\_\_name\_\_' == '\_\_main\_\_' :

app.run(debug=True)

**MODAL BUILDING:**

{

"nbformat": 4,

"nbformat\_minor": 0,

"metadata": {

"colab": {

"provenance": []

},

"kernelspec": {

"name": "python3",

"display\_name": "Python 3"

},

"language\_info": {

"name": "python"

}

},

"cells": [

{

"cell\_type": "code",

"execution\_count": 1,

"metadata": {

"id": "hcTNYk2gkEEz"

},

"outputs": [],

"source": [

"import keras \n",

"from keras.preprocessing.image import ImageDataGenerator"

]

},

{

"cell\_type": "code",

"source": [

"train\_datagen=ImageDataGenerator(rescale=1./255,shear\_range=0.2,rotation\_range=180,zoom\_range=0.2,horizontal\_flip=True)\n",

"\n",

"test\_datagen=ImageDataGenerator(rescale=1./255)"

],

"metadata": {

"id": "qv0tigKQkhCs"

},

"execution\_count": 2,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"from google.colab import drive\n",

"drive.mount('/content/drive')"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "vYYLFpV4pUow",

"outputId": "edbd61b7-8145-45ca-f44d-f0360d4fab82"

},

"execution\_count": 3,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Mounted at /content/drive\n"

]

}

]

},

{

"cell\_type": "code",

"source": [

"x\_train=train\_datagen.flow\_from\_directory('/content/drive/MyDrive/Dataset/Dataset/train\_set',target\_size=(128,128),batch\_size=32,class\_mode='binary')"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "QYDOgRVjkhL9",

"outputId": "3dfe1844-af22-41ee-c4b7-0cbf1fb9c642"

},

"execution\_count": 4,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Found 436 images belonging to 2 classes.\n"

]

}

]

},

{

"cell\_type": "code",

"source": [

"x\_test=test\_datagen.flow\_from\_directory('/content/drive/MyDrive/Dataset/Dataset/test\_set',target\_size=(128,128),batch\_size=32,class\_mode='binary')\n"

],

"metadata": {

"id": "m6\_p4ek6khPG",

"colab": {

"base\_uri": "https://localhost:8080/"

},

"outputId": "edab2410-be80-471e-da76-f6c5c4c72948"

},

"execution\_count": 5,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Found 121 images belonging to 2 classes.\n"

]

}

]

},

{

"cell\_type": "code",

"source": [

"x\_test.class\_indices"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "7hVHABgBARgk",

"outputId": "b32033e0-8865-4295-f6f6-246f7caf3f8e"

},

"execution\_count": 6,

"outputs": [

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"{'forest': 0, 'with fire': 1}"

]

},

"metadata": {},

"execution\_count": 6

}

]

},

{

"cell\_type": "code",

"source": [

"#import model building libraries\n",

"\n",

"#To define Linear initialisation import Sequential\n",

"from keras.models import Sequential\n",

"#To add layers import Dense\n",

"from keras.layers import Dense\n",

"#To create Convolution kernel import Convolution2D\n",

"from keras.layers import Convolution2D\n",

"#import Maxpooling layer\n",

"from keras.layers import MaxPooling2D\n",

"#import flatten layer\n",

"from keras.layers import Flatten\n",

"import warnings\n",

"warnings.filterwarnings('ignore')"

],

"metadata": {

"id": "CE\_pHGMUkhSI"

},

"execution\_count": 7,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"model=Sequential()\n"

],

"metadata": {

"id": "HKCjid3lkhU1"

},

"execution\_count": 8,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"#add convolutional layer\n",

"model.add(Convolution2D(32,(3,3),input\_shape=(128,128,3),activation='relu'))\n",

"#add maxpooling layer\n",

"model.add(MaxPooling2D(pool\_size=(2,2)))\n",

"#add flatten layer \n",

"model.add(Flatten())"

],

"metadata": {

"id": "ZNaaXJExkhXy"

},

"execution\_count": 9,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"#add hidden layer\n",

"model.add(Dense(150,activation='relu'))\n",

"#add output layer\n",

"model.add(Dense(1,activation='sigmoid'))"

],

"metadata": {

"id": "VqTGma-rkhaz"

},

"execution\_count": 10,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"model.add(Flatten())"

],

"metadata": {

"id": "v1NomAzfkhdm"

},

"execution\_count": 11,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"#add hidden layer\n",

"model.add(Dense(150,activation='relu'))\n",

"#add output layer\n",

"model.add(Dense(1,activation='sigmoid'))"

],

"metadata": {

"id": "hLQOkqxZkhgU"

},

"execution\_count": 12,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"model.compile(loss='binary\_crossentropy',optimizer=\"adam\",metrics=[\"accuracy\"])\n"

],

"metadata": {

"id": "j8dmqV6VnGuO"

},

"execution\_count": 13,

"outputs": []

},

{

"cell\_type": "code",

"source": [

"model.fit\_generator(x\_train,steps\_per\_epoch=14,epochs=10,validation\_data=x\_test,validation\_steps=4)\n"

],

"metadata": {

"id": "-n8ug6\_anGxr",

"colab": {

"base\_uri": "https://localhost:8080/"

},

"outputId": "eb7e2b87-a3ab-4c5c-da15-17a093f4edfb"

},

"execution\_count": 14,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Epoch 1/10\n",

"14/14 [==============================] - 146s 11s/step - loss: 0.6901 - accuracy: 0.6261 - val\_loss: 0.6889 - val\_accuracy: 0.5950\n",

"Epoch 2/10\n",

"14/14 [==============================] - 26s 2s/step - loss: 0.6832 - accuracy: 0.6445 - val\_loss: 0.6848 - val\_accuracy: 0.5950\n",

"Epoch 3/10\n",

"14/14 [==============================] - 28s 2s/step - loss: 0.6763 - accuracy: 0.6445 - val\_loss: 0.6810 - val\_accuracy: 0.5950\n",

"Epoch 4/10\n",

"14/14 [==============================] - 26s 2s/step - loss: 0.6700 - accuracy: 0.6445 - val\_loss: 0.6776 - val\_accuracy: 0.5950\n",

"Epoch 5/10\n",

"14/14 [==============================] - 27s 2s/step - loss: 0.6638 - accuracy: 0.6445 - val\_loss: 0.6755 - val\_accuracy: 0.5950\n",

"Epoch 6/10\n",

"14/14 [==============================] - 27s 2s/step - loss: 0.6586 - accuracy: 0.6445 - val\_loss: 0.6750 - val\_accuracy: 0.5950\n",

"Epoch 7/10\n",

"14/14 [==============================] - 26s 2s/step - loss: 0.6548 - accuracy: 0.6445 - val\_loss: 0.6752 - val\_accuracy: 0.5950\n",

"Epoch 8/10\n",

"14/14 [==============================] - 27s 2s/step - loss: 0.6529 - accuracy: 0.6445 - val\_loss: 0.6761 - val\_accuracy: 0.5950\n",

"Epoch 9/10\n",

"14/14 [==============================] - 25s 2s/step - loss: 0.6518 - accuracy: 0.6445 - val\_loss: 0.6772 - val\_accuracy: 0.5950\n",

"Epoch 10/10\n",

"14/14 [==============================] - 27s 2s/step - loss: 0.6512 - accuracy: 0.6445 - val\_loss: 0.6786 - val\_accuracy: 0.5950\n"

]

},

{

"output\_type": "execute\_result",

"data": {

"text/plain": [

"<keras.callbacks.History at 0x7f35fce35410>"

]

},

"metadata": {},

"execution\_count": 14

}

]

},

{

"cell\_type": "code",

"source": [

"model.save(\"forest1.h5\")"

],

"metadata": {

"id": "79jStSxbnG9W"

},

"execution\_count": 15,

"outputs": []

},

{

"cell\_type": "code",

"source": [],

"metadata": {

"id": "AIaNV\_A6nG\_v"

},

"execution\_count": 15,

"outputs": []

},

{

"cell\_type": "code",

"source": [],

"metadata": {

"id": "tI\_b4ZrInHDI"

},

"execution\_count": 15,

"outputs": []

},

{

"cell\_type": "code",

"source": [],

"metadata": {

"id": "Pt6BBg0PkhjM"

},

"execution\_count": 15,

"outputs": []

},

{

"cell\_type": "code",

"source": [],

"metadata": {

"id": "QjDhgAMJkhl0"

},

"execution\_count": 15,

"outputs": []

},

{

"cell\_type": "code",

"source": [],

"metadata": {

"id": "mGmaOj-Pkhoo"

},

"execution\_count": 15,

"outputs": []

},

{

"cell\_type": "code",

"source": [],

"metadata": {

"id": "BY8Ef1umkhrP"

},

"execution\_count": 15,

"outputs": []

},

{

"cell\_type": "code",

"source": [],

"metadata": {

"id": "egkD4uPikhuR"

},

"execution\_count": 15,

"outputs": []

},

{

"cell\_type": "code",

"source": [],

"metadata": {

"id": "tK2m-Du8khwn"

},

"execution\_count": 15,

"outputs": []

},

{

"cell\_type": "code",

"source": [],

"metadata": {

"id": "RDcKAPX2khzl"

},

"execution\_count": 15,

"outputs": []

},

{

"cell\_type": "code",

"source": [],

"metadata": {

"id": "f6ToyVMjkh2h"

},

"execution\_count": 15,

"outputs": []

},

{

"cell\_type": "code",

"source": [],

"metadata": {

"id": "wfuTHpW7kh6G"

},

"execution\_count": 15,

"outputs": []

}

]

}

**PREDICTION AND SMS ALERT WITH TWILIO:**

!pip install playsound

!pip install twilio

**from** google.colab **import** drive

drive**.**mount('/content/drive')

**import** cv2

**import** numpy **as** np

**from** keras.preprocessing **import** image

**from** keras.models **import** load\_model

**from** twilio.rest **import** Client

**from** playsound **import** playsound

*#import load\_model from keras.model*

**from** keras.models **import** load\_model

*#import image class from keras*

**from** tensorflow.keras.preprocessing **import** image

*#import numpy*

**import** numpy **as** np

**import** cv2

**import** keras

**from** keras.preprocessing.image **import** ImageDataGenerator

**from** keras.models **import** Sequential

*#To add layers import Dense*

**from** keras.layers **import** Dense

*#To create Convolution kernel import Convolution2D*

**from** keras.layers **import** Convolution2D

*#import Maxpooling layer*

**from** keras.layers **import** MaxPooling2D

*#import flatten layer*

**from** keras.layers **import** Flatten

**import** warnings

warnings**.**filterwarnings('ignore')

**from** tensorflow.keras.utils **import** img\_to\_array

!pip install pygobject

**from** google.colab.patches **import** cv2\_imshow

In [ ]:

!pip install playsound

!pip install pygobject

*#i havent imported*

*#can you tell me the cmd*

**from** playsound **import** playsound

model **=** load\_model(r'/content/drive/MyDrive/forest1 (2).h5')

video **=** cv2**.**VideoCapture('/content/drive/MyDrive/Dataset/Dataset/test\_set/with fire/599857.jpg')

name **=** ['forest','with fire']

**while**(1):

success,frame **=** video**.**read()

cv2**.**imwrite("/content/drive/MyDrive/Dataset/Dataset/test\_set/with fire/FORESTFIRE (1).jpg",frame)

img **=** image**.**load\_img("/content/drive/MyDrive/Dataset/Dataset/test\_set/with fire/Fire\_2\_696x392.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])

print(pred)

cv2**.**putText(frame,"predicted class = "**+**str(name[p]),(100,100),cv2**.**FONT\_HERSHEY\_SIMPLEX,1,(0,0,0),1)

pred **=** model**.**predict(x)

**if** pred[0]**==**1:

*#playsound(r'/content/drive/MyDrive/ALRMElec\_Smoke detector alarm (ID 0800)\_BSB.mp3')*

account\_sid **=** 'ACbd262485b92b4d4fe5a1078bd227c786'

auth\_token **=** 'e680bb6a80fdd76e7853b354f08dda2e'

client **=** Client(account\_sid, auth\_token)

myMessage **=** client**.**messages**.**create(

body**=**'Forest Fire detected,Stay alert!',

from\_**=**'+17128008239',

to**=**'+918072020392')

print(myMessage**.**sid)

print("Fire detected")

print("SMS Sent!")

**else**:

print("No Danger")

cv2\_imshow(frame)

**if** cv2**.**waitKey(1) **&** 0xFF **==** ord('a'):

**break**

video**.**release()

cv2**.**destroyAllWindows()