

PROJECT REPORT DOCUMENTATION

Team ID	PNT2022TMID04587
Project Title	Industry-specific intelligent fire management system

1. INTRODUCTION

1.1 Project Overview

In this project we will be developing a IoT based device to help in the fire management in the industry. Our device will be using various sensors like flame sensor, temperature sensor and gas sensor. The project will be highly efficient in extinguishing the fire in the industry and reduce the damages that is caused by fire.

1.2 Purpose

The main aim of the project is to provide safe and efficient industry-specific intelligent fire management system. This project will be used in fire prevention, temperature monitoring, gas and flame management.

2. LITERATURE SURVEY

Suwarjono, S. Wayangkau, I.H. Istanto, T. Rachmat, R. Marsujitullah, M. Hariyanto, H. Caesarendra, W. Legutko, S. Glowacz, A. “Design of a Home Fire Detection System Using Arduino and SMS Gateway”. Knowledge 2021, 1, 61-74. The main of their project was to help house owners and firefighters to quickly detect the fire and the leakage of gas using gas sensor and temperature sensor

M. Trinath Basu, Ragipati Karthik , J. Mahitha , V. Lokesh Reddy “IoT based forest fire detection system” International Journal of Engineering & Technology 7 (2.7) 2018. With the help of an RF module the data can be sent to the IoT cloud which is received from the gas sensors and temperature sensor. Using the readings if the values exceed a certain threshold the buzzer will start to ring

Hamood Alqourabah , Amgad Muneer , Suliman Mohamed Fati “A smart fire detection system using IoT technology with automatic water sprinkler” International Journal of Electrical and Computer Engineering (IJECE) Vol. 11, No. 4, August 2021. In this project the gas concentration level is constantly being measured by the gas sensor and uploaded to the cloud, if the flame sensor detects the presence of fire, then the water sprinkler is automatically activated and puts out the fire

Nyyssönen, T, Rajakko, J, and Keski-Rahkonen, O. “On the reliability of fire detection and alarm systems”. Exploration and analysis of data from nuclear and non-nuclear installations. Finland: N. p., 2005. They made a literature review on the reliability of fire detection data and which was resulting in rough estimation of some failure frequencies.

Ponnusamy V, Palanisamy S, Paulraj P, and Kanchana S. “Fire Safety in the Textile Industry”. DOI: 10.35530 / IT.070.06.1615 In their review they discussed the various causes of fire in the textile industry, using the thermometer and hygrometer they can keep the temperature below 308 K and humidity below 70 %

Matthew P Thompson, Donald G MacGregor, Christopher J Dunn, David E Calkin, John Phipps “Rethinking the wildland fire management system” *Journal of Forestry*, Volume 116, Issue 4, July 2018. Their primary focus was how to catalyze changes in the fire manager behavior. Using the forest service of the US Department of Agriculture as a test case, they deduced the fundamental changes to the community’s way of thinking and plans towards the wildfire

Laura Bravo Diaz, Xuanze He, Zhenwen Hu, Francesco Restuccia, Monica Marinescu, Jorge Varela Barreras, Yatish Patel, Gregory Offer and Guillermo Rein “Review of Fire Safety of Lithium-Ion Batteries: Industry Challenges and Research Contributions” *Journal of the Electrochemical Society*, Volume 167, Number 9. They discussed the safety challenges faced by the industry, and the ignition and propagation challenges. It is mentioned that gas sensors are essential for the detection of the initial thermal runaway along with heat and smoke sensors to support.

Ross D.Collins, Richard de Neufvillea, João Claro, Tiago Oliveira, Abílio P.Pacheco “ Forest fire management to avoid unintended consequences”*Journal of Environmental Management*, Volume 130, 2013. They explored how the physical and political dynamics affect suppression, they have used system dynamics to model the forest fire.

2.1 Existing Problem

The existing mechanism in the fire management system is ineffective and expensive. It also requires human interaction to take decision which cannot be feasible at the time of emergency. We will need a spontaneous device which can prevent fire damages before it happens or take appropriate decisions when an emergency occurs

2.2 Reference

Stipanicev D., Vuko T., Krstinic D., Stula M., Bodrozic L., Forest Fire Protection by Advanced Video Detection System- Croatian Experiences, Split, Croatia, 2006

Losso A., Corgnati L., Perona G., Early Forest Fire Detection: Smoke Identification through innovative Image Processing using Commercial Sensors, Environment Including Global Change, Palermo, Italy, 2009

Kovacs R., Kiss B., Nagy A., Vamos R., Early Forest Fire Detection System For Vegetable Fire in the Aggtelek National Park Budapest, Hungary, 2004

Kelha V., Rauste Y., Buongiorno A., Forest Fire Detection by Satellites for Fire Control, European Space Agency, Finland, 2000

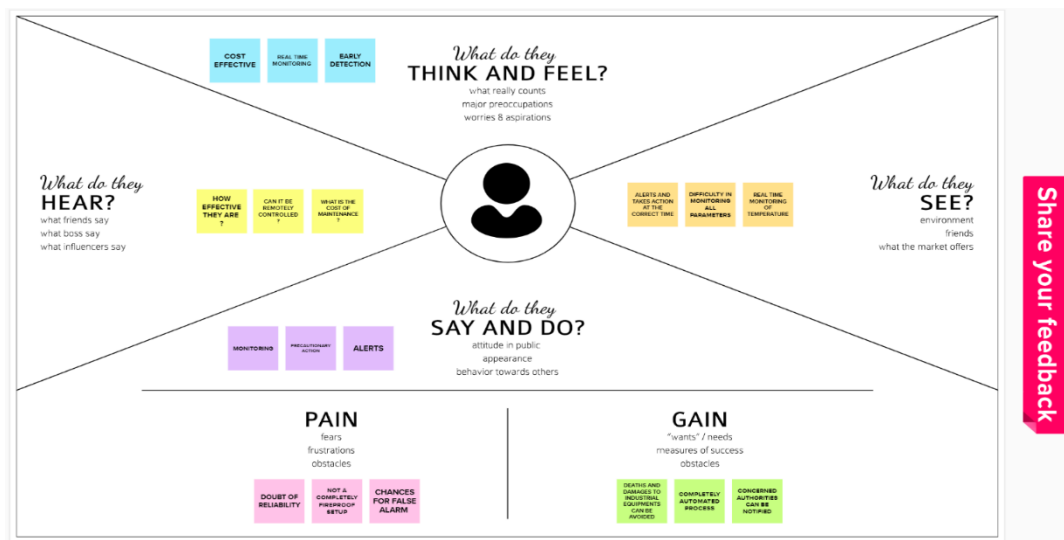
Manyangadze T., Forest Fire Detection for Near Real Time Monitoring using Geostationary Satellite, International Institute for Geo-information Science and Earth Observation, Enschede, Netherland, 2009

2.3 Problem Statement Definition

The fire management system in houses and industries are not very reliable, efficient, cost effective and does not have any advance processing and does not have any features like automatic alert system for admin and authorities and in many buildings there are using older fire safety system that doesn't can even activate the sprinkler system and all of they don't communicate with each other properly to prevent false alarm also monitor the entire system using a applications .

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Mapping Canvas



3.2 Ideation & Brainstorming

2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

 10 minutes

TIP

You can select a sticky note and hit the pencil icon to select a tool to start drawing!

Abhinaya

Using technology	Reduce distraction	Reduce complexity
Useful product development	Analytics	Reducing complexity of the product

Abhinavash

Lowest complexity	Lowest complexity	Low complexity
Lowest complexity	Lowest complexity of the product	

Akshaya

Lowest complexity of the product	Lowest complexity of the product	Lowest complexity of the product
Lowest complexity of the product		

Charan Surya

Lowest complexity of the product	Lowest complexity of the product	Lowest complexity of the product
Lowest complexity of the product	Lowest complexity of the product	Lowest complexity of the product

3.3 Proposed Solution

S. No	Parameter	Description
1	Problem Statement (Problem to be solved)	To improve the safety management system in industries. Improving the safety management system against the fire incidents in industries.
2	Idea / Solution description	To implement the fire safety management in industry based on IOT using Arduino uno board with fire detection and fire extinguisher system. And using some sensors (Humidity sensor, Flame sensor, smoke sensor) with GPS tracking system
3	Novelty / Uniqueness	Usage of liquid Nitrogen. Liquid nitrogen will immediately vaporize causing a cooling effect and makes the site deprived of oxygen. Using water sprinklers after liquid nitrogen can put off even intense fire effectively.

4	Social Impact / Customer Satisfaction	<p>1) Harmful gasses can be purified and released into atmosphere</p> <p>2) Cause of impact can be traced from the sensor data that can be analyzed to prevent future accidents.</p>
5	Business Model (Revenue Model)	<p>1) This is used to calculate the probability of ignition and spread of fires across a landscape.</p> <p>2) This outcome allows for a better understanding of how changes in one aspects of management can affect other aspects of management</p>
6	Scalability of the Solution	<p>It is trying to execute this technique as we need to introduce an arduino gadget which was modified with an Arduino that takes received signals from sensors .Easy reliability and maintenance. Required low time for maintenance. Cost is reasonable.</p>

3.4 Proposed Solution Fit

Project Title: Industry-Specific Intelligent Fire Management System
Team ID: PNT2022TMID04587

Project Design Phase-I - Solution Fit Template

Define CS, fit into CC	1. CUSTOMER SEGMENT(S) Who is your customer? My customers are those who own organization and industries which can be easily subjected into fire accident. Examples: oil Industries, cotton industries	6. CUSTOMER CONSTRAINTS What constraints prevent your customers from taking action or limit their choices of solutions? i.e. spending power, budget, no cash, network connection, available devices. Fire extinguishers hanging on the walls are weight and the industries or organization has less number of extinguishers so many of them could not use it when occurrence of fire.	5. AVAILABLE SOLUTIONS Which solutions are available to the customers when they face the problem or need to get the job done? what have they tried in the past? In most places, fire extinguishers are hung on the walls for safety purposes, when there is an occurrence of fire, extinguishers are make used it by using man power and in some places ,an automatic fire detection system is also there.	Evaluate AS differentials
	2. JOBS-TO-BE-DONE / PROBLEMS Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides If fire occur in any organization or industries this fire alarm detects the fire and indicates through the alarm and sprinkles the water to extinguish the fire.	9. PROBLEM ROOT CAUSE What is the real reason that this problem exists? What is the back story behind the need to do this job? i.e. customers have to do it because of the change in regulations. In crackers industries there used chemicals for crackers which are easily inflammable so it can cause fire accident even if a tiny sparks occurs as it happened once in a sivakasi.. Likewise many other problems can also be the root cause of fire	7. BEHAVIOUR What does your customer do to address the problem and get the job done? i.e. directly related find the right solar panel installer, calculate usage and benefit; indirectly associated: customers spend free time on volunteering(i.e. Green peace) Monitoring the fire with the help of the automatic fire detection system in the respective places to avoid fire accidents ,avoid using old machines and safety precautions are to be followed.	
Focus on J&P, fit into BE, understand RC				Focus on J&P, fit into BE, understand RC

	<p>3. TRIGGER</p> <p>What triggers customers to act?</p> <ul style="list-style-type: none"> Improving the fire management system Giving awareness to Employees 	<p>6. CUSTOMER CONSTRAINTS</p> <p>What constraints prevent your customers from taking action or limit their choices of solutions?</p> <ul style="list-style-type: none"> All industry workers need to have smart devices (smart phones) and also must have a connected network. Well monitoring system 	<p>9. PROBLEM ROOT CAUSE</p> <p>What is the real reason that this problem exists? What is the back story?</p> <ul style="list-style-type: none"> Failure of Machinery and other equipment. Electrical short circuiting or overloading Chemical Reaction/Runway Reaction. 	
	<p>10. YOUR SOLUTION</p> <p>Detecting the presence of fire and Alerting the industrial workers and its presence to the fire officials through the internet. and save the property and life of people. And Usage of automatic sprinklers along with liquid nitrogen is able to put off a large fire break out. In case of toxic gas leakage, they are purified and released into the atmosphere.</p>			

4. REQUIREMENT ANALYSIS

4.1 Functional Requirement

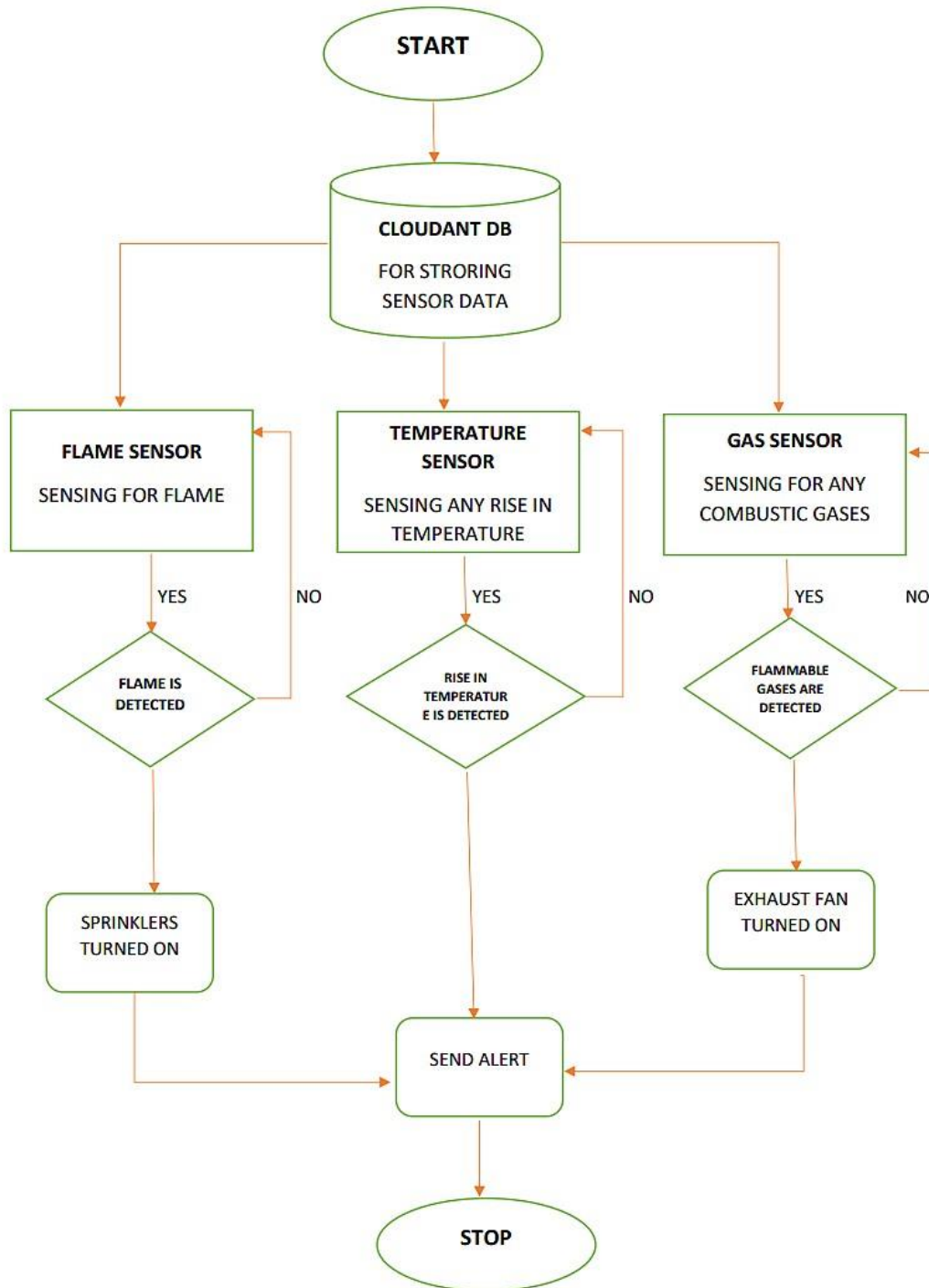
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Sensing function	Fire breakout has to be sensed by smoke detectors. Gas leakage has to be sensed by gas sensors.
FR-2	Alerting function	Blaring of alarms
FR-3	Actuation function	Activation of sprinklers. Turning ON the exhaust Fan.
FR-4	Notification	Sending SMS to the Authorities. Sending SMS with location to the fire station

4.2 Non-Functional Requirement

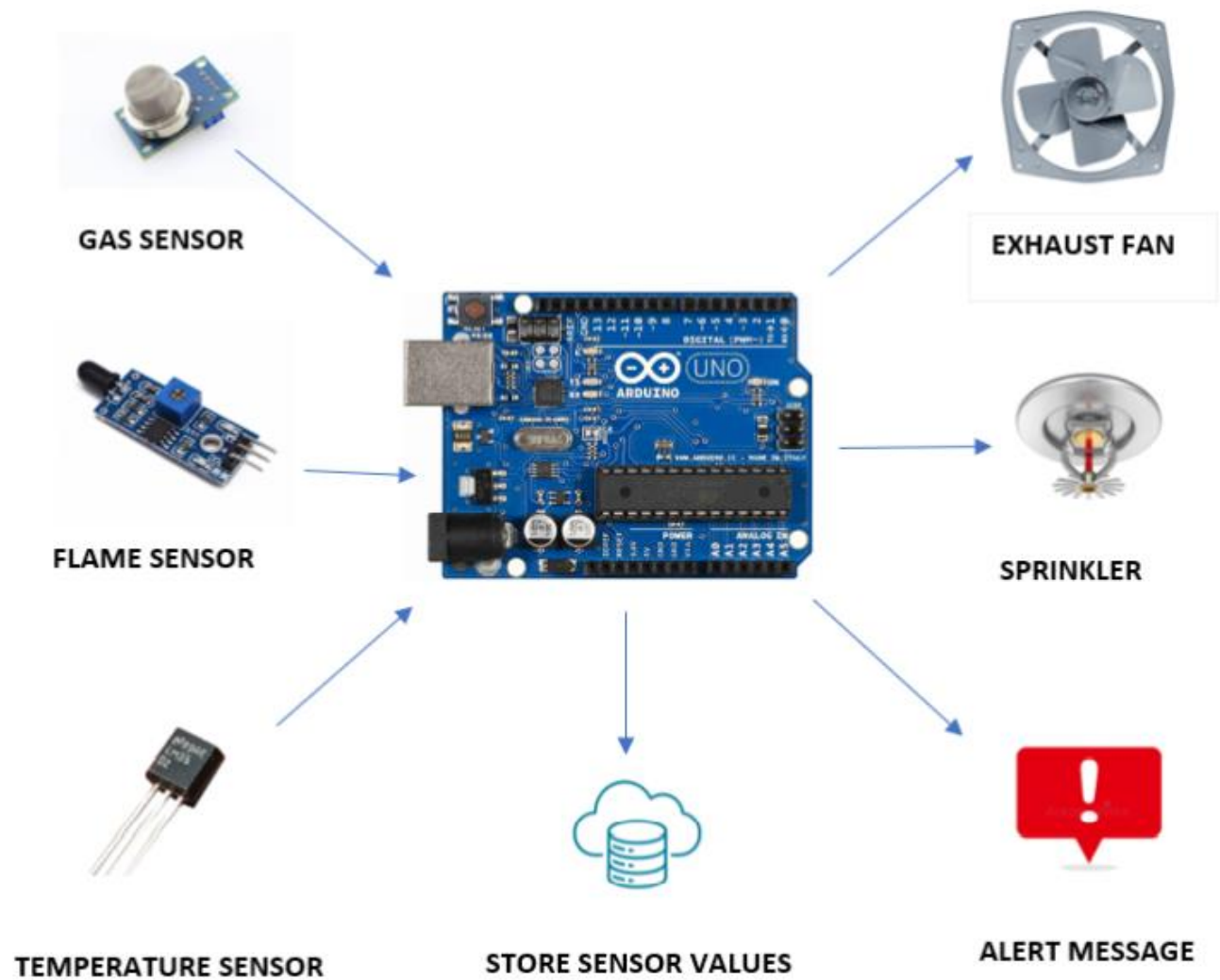
FR No	Non-Functional Requirement	Description
NFR-1	Usability	Ease of use and longevity of the system.
NFR-2	Security	Software remains secured in the face of attacks.
NFR-3	Reliability	High accuracy.
NFR-4	Performance	Faster response.
NFR-5	Availability	Availability of the systems for institutions, restaurants and other public places
NFR-6	Scalability	It accommodates easy modification for various requirements

5. PROJECT DESIGN

5.1 Data Flow Diagram



5.2 Solution & Technical Architecture



5.3 User Stories

User Type	Functional Requirement(Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Industry)	Assembling	USN-1	As a user, I must position the sensors in the appropriate location.	I have access to my sensor triggers	High	Sprint-1
		USN-2	As a user, I must test my hardware to ensure that it is operational.	I can use the serial monitor to keep track of all sensor values.	High	Sprint-1
	User Registration	USN-3	As a user, I can create user accounts for the model's essential software.	I can sign up and utilize user Login to access the dashboard.	Medium	Sprint-2
		USN-4	As a user, I can verify that notifications and SMS are delivered properly	I can verify the alerts via Fast2SMS	High	Sprint-1
	Cloud Monitoring	USN-5	As a user, I can keep track on how long data is kept in IBM Cloudant	I can constantly monitor and obtain sensor data.	High	Sprint-1

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Wokwi Simulation	USN-1	Making Hardware device or Using Worwi. Connect Temperature, Flame, Gas sensor to Arduino with python script	2	High	Charan Surya, Akshaya, Abinaya, Abinivesh
Sprint-2	IBM Cloud	USN-2	Create Device in the IBM Watson IOT Platform and link it to Noad-red	2	High	Charan Surya, Akshaya, Abinaya, Abinivesh
Sprint-3	Node Red service	USN-3	Create a dashboard and link the cloud platform	2	High	Charan Surya, Akshaya, Abinaya, Abinivesh
Sprint-4	MIT app inverter	USN-4	Link Device, IBM cloud and the developed application. Feed the data to sprinkler, exhaust fan.	2	High	Charan Surya, Akshaya, Abinaya, Abinivesh

6.2 Sprint Delivery Schedule

TITLE	DESCRIPTION	DATE
IDEATION PHASE		
Literature Survey & Information Gathering	Literature survey on the selected project & gathering information by referring the, technical papers, research publications etc.	18 SEPTEMBER 2022
Prepare Empathy Map	Prepare Empathy Map Canvas to capture the user Pains & Gains, Prepare list of problem statements.	19 SEPTEMBER 2022
Problem Statement	List of problems in the project.	25 SEPTEMBER 2022
Brainstorm And Idea Prioritization	List them by organizing the brainstorming session and prioritize the top 3 ideas based on the feasibility & importance.	27 SEPTEMBER 2022
Project Design Phase - I		
Proposed Solution	Prepare the proposed solution document, which includes then novelty, feasibility of idea, business model, social impact, scalability of solution, etc.	28 SEPTEMBER 2022
Problem Solution Fit	Prepare problem - solution fit document.	01 OCTOBER 2022
Solution Architecture	Prepare solution architecture document.	03 SEPTEMBER 2022

TITLE	DESCRIPTION	DATE
Project Design Phase - II		
Customer Journey	Prepare the customer journey maps to understand the user interactions & experiences with the application (entry to exit).	08 OCTOBER 2022
Functional Requirement	Prepare the functional requirement document.	15 OCTOBER 2022
Data Flow Diagrams	Draw the data flow diagram and submit for review.	20 OCTOBER 2022
Technology Architecture	Prepare the technology architecture diagram.	25 OCTOBER 2022
Project Planning Phase		
Prepare Project Planning & Sprint Delivery Plan	Prepare the Product Backlog, Sprint Planning, Stories, and Story points.	11 OCTOBER 2022
Prepare Milestone & Activity List	Prepare the milestones & activity list of the project.	11 OCTOBER 2022
Project Development Phase		
Project Development - Delivery of Sprint-1, 2, 3 & 4	Develop & submit the developed code by testing it.	IN PROGRESS

7. CODING & SOLUTIONING

7.1 Feature 1

Generating Random Input and Send it to IBM Watson Cloud

```
import time
import sys
import ibmiotf.device
import ibmiotf.application
import random
```

```
#Provide your IBM Watson Device Credentials
organization = "inbee2"
deviceType = "NodeMCU"
deviceId = "12345"
authMethod = "token"
authToken = "12345678"
```

```

# Initialize GPIO

def myCommandCallback1(cmd):
print("Command received: %s" % cmd.data['command']) status = cmd.data['command']
    if status == "sprinkleron": print("sprinkler is on")
else:
    print("sprinkler is off") print(cmd)

def myCommandCallback2(cmd):
print("Command received: %s" % cmd.data['command']) status = cmd.data['command']
    if status == "fanon":
        print("fan is on") else:
        print("fan is off") print(cmd)

try:
    deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod,
"auth-token": authToken}
deviceCli = ibmiotf.device.Client(deviceOptions) #.....

except Exception as e:
print("Caught exception connecting device: %s" % str(e)) sys.exit()

# Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greeting" 10 times
deviceCli.connect()

while True:
    #Get Sensor Data from DHT11

    temp=random.randint(0,70)
    gas=random.randint(0,100)
    flame=random.randint(0,1)

    data = { 'temp': temp, 'gas': gas, 'flame': flame }

```

```

# print data
def myOnPublishCallback():
    print ("Published Temperature = %s C" % temp, "Gas = %s %" % gas, "flame = %s %" %
% flame,"to IBM Watson")

success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish=myOnPublishCallback)
if not success:
    print("Not connected to IoTTF") time.sleep(1)

deviceCli.commandCallback1 = myCommandCallback1 deviceCli.commandCallback2 =
myCommandCallback2

# Disconnect the device and application from the cloud deviceCli.disconnect()

```

```

Python 3.7.4 Shell
File Edit Shell Debug Options Window Help
Python 3.7.4 (tags/v3.7.4:09359112e, Jul 8 2019, 19:29:22) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: E:\IBM\IBM.py =====
2022-11-19 22:38:24,864 ibmiotf.device.Client INFO Connected successfully: d:inbee2:NodeMCU:12345
Published Temperature = 34 C Gas = 0 % flame = 1 % to IBM Watson
Published Temperature = 34 C Gas = 0 % flame = 1 % to IBM Watson
Published Temperature = 34 C Gas = 0 % flame = 1 % to IBM Watson
Published Temperature = 0 C Gas = 77 % flame = 0 % to IBM Watson
Published Temperature = 33 C Gas = 84 % flame = 1 % to IBM Watson
Published Temperature = 29 C Gas = 27 % flame = 1 % to IBM Watson
Published Temperature = 7 C Gas = 1 % flame = 1 % to IBM Watson
Published Temperature = 33 C Gas = 99 % flame = 1 % to IBM Watson
Published Temperature = 5 C Gas = 60 % flame = 1 % to IBM Watson
Published Temperature = 32 C Gas = 25 % flame = 0 % to IBM Watson
Published Temperature = 28 C Gas = 78 % flame = 0 % to IBM Watson
Published Temperature = 23 C Gas = 3 % flame = 0 % to IBM Watson
Published Temperature = 64 C Gas = 70 % flame = 1 % to IBM Watson
Published Temperature = 22 C Gas = 63 % flame = 1 % to IBM Watson
Published Temperature = 17 C Gas = 28 % flame = 1 % to IBM Watson
Published Temperature = 31 C Gas = 52 % flame = 1 % to IBM Watson
Published Temperature = 19 C Gas = 42 % flame = 1 % to IBM Watson
Published Temperature = 10 C Gas = 87 % flame = 1 % to IBM Watson
Published Temperature = 16 C Gas = 30 % flame = 1 % to IBM Watson
Published Temperature = 48 C Gas = 71 % flame = 1 % to IBM Watson
Published Temperature = 47 C Gas = 87 % flame = 1 % to IBM Watson
Published Temperature = 65 C Gas = 7 % flame = 0 % to IBM Watson
Published Temperature = 32 C Gas = 67 % flame = 1 % to IBM Watson
Published Temperature = 36 C Gas = 87 % flame = 0 % to IBM Watson
Published Temperature = 64 C Gas = 83 % flame = 1 % to IBM Watson
Published Temperature = 17 C Gas = 87 % flame = 0 % to IBM Watson
Published Temperature = 69 C Gas = 98 % flame = 1 % to IBM Watson
Published Temperature = 24 C Gas = 80 % flame = 0 % to IBM Watson
Published Temperature = 49 C Gas = 39 % flame = 1 % to IBM Watson
Published Temperature = 69 C Gas = 32 % flame = 1 % to IBM Watson
Published Temperature = 52 C Gas = 65 % flame = 0 % to IBM Watson
Published Temperature = 58 C Gas = 27 % flame = 0 % to IBM Watson
Published Temperature = 37 C Gas = 4 % flame = 0 % to IBM Watson
Published Temperature = 26 C Gas = 2 % flame = 1 % to IBM Watson
Published Temperature = 52 C Gas = 20 % flame = 1 % to IBM Watson
Published Temperature = 17 C Gas = 11 % flame = 0 % to IBM Watson
Published Temperature = 15 C Gas = 31 % flame = 0 % to IBM Watson
Published Temperature = 11 C Gas = 31 % flame = 0 % to IBM Watson
Published Temperature = 49 C Gas = 25 % flame = 1 % to IBM Watson
Published Temperature = 13 C Gas = 30 % flame = 1 % to IBM Watson
Published Temperature = 9 C Gas = 60 % flame = 1 % to IBM Watson
Published Temperature = 65 C Gas = 100 % flame = 0 % to IBM Watson
Published Temperature = 61 C Gas = 3 % flame = 1 % to IBM Watson
Published Temperature = 17 C Gas = 46 % flame = 1 % to IBM Watson
Published Temperature = 19 C Gas = 57 % flame = 1 % to IBM Watson
Published Temperature = 66 C Gas = 94 % flame = 0 % to IBM Watson
Published Temperature = 58 C Gas = 53 % flame = 1 % to IBM Watson

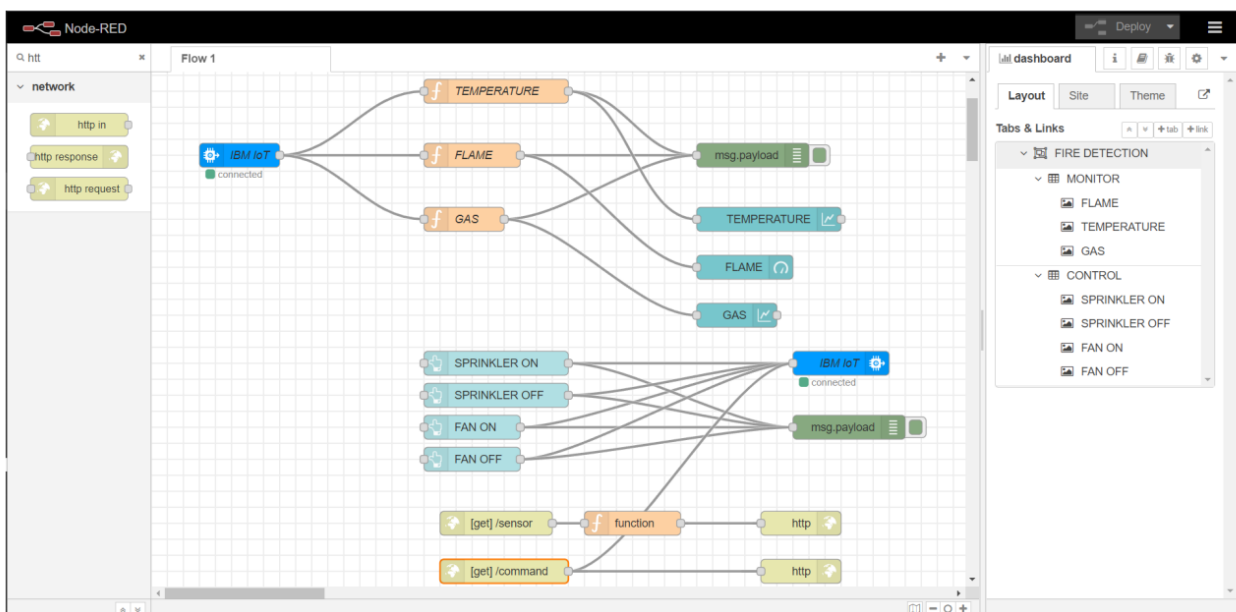
```

IBM Watson IoT Platform

<

7.2 Feature 2

Send the Data from IBM Watson Cloud to Node-RED and Integrate MIT App Inventor





Palette

Search Components...

User Interface

- Button
- CheckBox
- DatePicker
- Image
- Label
- ListPicker
- ListView
- Notifier
- PasswordTextBox
- Slider
- Spinner
- Switch
- TextBox
- TimePicker
- WebView

Layout

Media

Drawing and Animation

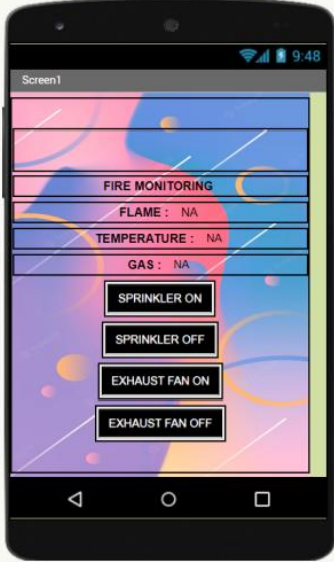
Maps

Charts

Viewer

☐ Display hidden components in Viewer

Phone size (505,320)



Non-visible components

- Web1
- Web2
- Clock1

Components

- Screen1
 - VerticalArrangement1
 - HorizontalArrangement1
 - Label1
 - HorizontalArrangement2
 - Label2
 - Label3
 - HorizontalArrangement3
 - Label4
 - Label5
 - HorizontalArrangement4
 - Label6
 - Label7
 - HorizontalArrangement5
 - Button1

Properties

Screen1

AboutScreen

AccentColor

AlignHorizontal

AlignVertical

AppName

BackgroundColor

BackgroundImage

BigDefaultText

BlocksToolkit

CloseScreenAnimation

DefaultFileScope

HighContrast

Icon

OpenScreenAnimation

MIT APP INVENTOR

Projects • Connect • Build • Settings • Help • My Projects View Trash Guide Report an Issue English • charansuriya2002@gmail.com •

FIRE_SAFETY_APP Screen1 Add Screen Remove Screen Publish to Gallery Designer Blocks

Blocks

- Built-in
 - Control
 - Logic
 - Math
 - Text
 - Lists
 - Dictionaries
 - Colors
 - Variables
 - Procedures
- Screen1
 - VerticalArrangement1
 - HorizontalArrangement1
 - HorizontalArrangement2
 - Label1
 - HorizontalArrangement3
 - Label2

Media

- mobile-w...-599.webp
- Upload File...

Viewer

when Clock1.Timer

- do
 - set Web1.Uri to http://159.122.179.91:31976/sensor
 - call Web1.Get

when Web1.GoToText

- uri responseCode responseType responseContent
- do
 - set Label3.Text to look up in pairs key flame pairs call Web1.JsonTextDecode jsonText get responseContent
 - notFound not found
 - set Label5.Text to look up in pairs key temp pairs call Web1.JsonTextDecode jsonText get responseContent
 - notFound not found
 - set Label7.Text to look up in pairs key gas pairs call Web1.JsonTextDecode jsonText get responseContent
 - notFound not found

Show Warnings

when Button1.Click

- do
 - set Web2.Uri to http://159.122.179.91:31976/command?command=sprinkler
 - call Web2.Get

when Button2.Click

- do
 - set Web2.Uri to http://159.122.179.91:31976/command?command=sprinkler
 - call Web2.Get

ADVANTAGES & DISADVANTAGES

Advantages:

1. Better and more accurate fire detection – With an IoT-based system, sensors can be placed in strategic locations around a building or area to more accurately detect a fire
2. Faster response times – An IoT-based system can immediately notify the relevant authorities when a fire is detected, resulting in faster response times.
3. Reduced false alarms – With an IoT-based system, false alarms can be reduced as the system can be programmed to only notify the authorities when certain conditions are met (e.g. a certain level of heat is detected).
4. Greater efficiency – An IoT-based system can automate many of the tasks associated with fire management, such as dispatching firefighters and keeping track of firefighting resources.
5. Increased Efficiency: By automating tasks and processes, an IoT system

can help fire departments become more efficient in their operations.

6. Improved Safety: With real-time monitoring and alerts, an IoT system can help keep firefighters safe by providing them with timely information about potential hazards.

7. Enhanced Collaboration: An IoT system can enable better collaboration between firefighters and other emergency responders, such as police and ambulance services.

8. Greater Transparency: An IoT system can provide greater transparency into the operations of a fire department, which can be beneficial for both the department and the public.

9. Reduced Costs: An IoT system can help reduce the costs associated with running a fire department, such as by reducing the need for manual labor.

Disadvantages:

1. Implementation costs – An IoT-based Intelligent Fire Management System can be expensive to implement, especially if a large number of sensors are required.

2. Maintenance costs – An IoT-based system can also be expensive to maintain, as sensors and other hardware will need to be regularly checked and replaced.

3. Security risks – As IoT-based systems rely on networked devices, they can be vulnerable to hacking and other security risks.

4. Privacy concerns – Some people may be concerned about the privacy implications of having an IoT-based system, as it will be constantly collecting data about people and their activities

5. Complexity: An IoT system can be complex to set up and maintain, and may require specialized skills and knowledge.

6. Reliability: An IoT system can be subject to outages and other disruptions, which could impact the reliability of the system.

7. Security: An IoT system can be vulnerable to security threats, such as

hacking and data breaches.

8. Privacy: An IoT system can collect and store large amounts of data, which could potentially be used to infringe on the privacy of individuals.

9. Cost: An IoT system can be expensive to implement and maintain.

CONCLUSION

The proposed system, presents the advancement of Internet technology in day to day life. The system is suitable for real time small scale industrial process monitoring and controlling applications. proposed module implemented on ESP32, one of the best solutions to implement IoT applications. The module outline was tried, actualized and the accuracy and working of the system was verified.

Benefits of use IoT in industry:

Elimination of long wiring.

Web based remote monitoring.

Immediate action on failures.

Ease of maintenance.

FUTURE SCOPE

The future scope of Iot based intelligent fire management system is to develop a system that can automatically detect and extinguish fires. The system should be able to identify the type of fire and provide the appropriate response. It should also be able to send alerts to the authorities in case of a fire.

The future of IoT based intelligent fire management system looks very promising. With the help of IoT, the system will be able to monitor the fire situation in real time and take appropriate action accordingly. The system will also be able to automatically detect the fire and send alerts to the concerned authorities.

APPENDIX

```
#include "DHTesp.h" const
int DHT_PIN = 15; DHTesp
dhtSensor;

void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200); Serial.println("Hello,
  ESP32!");
  dhtSensor.setup(DHT_PIN,DHTesp::DHT22);
}

void loop() {
  TempAndHumidity data = dhtSensor.getTempAndHumidity();
  Serial.println("Temp: "+ String(data.temperature,2)+"c");
  Serial.println("Humidity: "+ String(data.humidity,1)+"%");
  Serial.println(" -----");
  delay(100); // this speeds up the simulation
}
```

```

import time import sys
import ibmiotf.device import
ibmiotf.application import random

#Provide your IBM Watson Device Credentials organization =
"inbee2"
deviceType = "NodeMCU" deviceId =
"12345" authMethod = "token"
authToken = "12345678"

# Initialize GPIO

def myCommandCallback1(cmd):
print("Command received: %s" % cmd.data['command']) status = cmd.data['command']
if status == "sprinkleron": print("sprinkler is on")
else:
    print("sprinkler is off") print(cmd)

def myCommandCallback2(cmd):
print("Command received: %s" % cmd.data['command']) status = cmd.data['command']
if status == "fanon":
    print("fan is on") else:
    print("fan is off") print(cmd)

try:
    deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod,
"auth-token": authToken}
deviceCli = ibmiotf.device.Client(deviceOptions) #.....

except Exception as e:
print("Caught exception connecting device: %s" % str(e)) sys.exit()

# Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greeting" 10 times

```

```
deviceCli.connect()
```

```
while True:
```

```
    #Get Sensor Data from DHT11
```

```
    temp=random.randint(0,70)
```

```
    gas=random.randint(0,100)
```

```
    flame=random.randint(0,1)
```

```
    data = { 'temp': temp, 'gas': gas, 'flame': flame } #print data
```

```
    def myOnPublishCallback():
```

```
        print ("Published Temperature = %s C" % temp, "Gas = %s %" % gas, "flame = %s %" %
```

```
% flame,"to IBM Watson")
```

```
    success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish=myOnPublishCallback)
```

```
    if not success:
```

```
        print("Not connected to IoT") time.sleep(1)
```

```
    deviceCli.commandCallback1 = myCommandCallback1 deviceCli.commandCallback2 =
```

```
    myCommandCallback2
```

```
# Disconnect the device and application from the cloud deviceCli.disconnect()
```