



**IoT BASED SMART CROP PROTECTION
SYSTEM FOR AGRICULTURE
PROJECT REPORT**

Submitted By

TEAM ID PNT2022TMID45497

ISHWARYA T (TEAM LEAD) (812619205010)

RAMVAIRAVAN R M (812619205018)

SAJITH AHAMED S (812619205020)

GOWRI S (812619205007)

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

BACHELOR OF TECHNOLOGY

IN

**M.A.M COLLEGE OF ENGINEERING,
TRICHIRAPPALI,
TRICHY-CHENNAI TRUNK ROAD,SIRUGANUR,
TRICHIRAPPALI-621 105**

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE NO.
NO.	INTRODUCTION	4
1	1.1 PROJECT OVERVIEW	4
	1.2 PURPOSE	4
	LITERATURE SURVEY	5
2	2.1 EXISTING PROBLEM	5
	2.2 REFERENCES	5
	2.3 PROBLEM STATEMENT DEFINITION IDEATION	6
	AND PROPOSED SOLUTION	7
3	3.1 EMPATHY MAP CANVAS	7
	3.2 IDEATION AND BRAINSTORMING	7
	3.3 PROPOSED SOLUTION	8
	3.4 PROBLEM-SOLUTION FIT	9
	REQUIREMENT ANALYSIS	10
4	4.1 FUNCTIONAL REQUIREMENT	10
	4.2 NON- FUNCTIONAL REQUIREMENT	10
	PROJECT DESIGN	11
5	5.1 DATA FLOW DIAGRAM	11

5.3 USER-STORIES

13

ii

PROJECT PLANNING AND SCHEDULING

14

6

6.1 SPRINT PLANNING AND ESTIMATION

14

6.2 SPRINT DELIVERY SCHEDULE

15

6.3 REPORT FROM JIRA

15

CODING AND SOLUTIONS

25

7

7.1 FEATURE 1

25

7.2 FEATURE 2

26

7.3 DATABASE SCHEMA

27

TESTING

29

8

8.1 TEST CASES

29

8.2 USER ACCEPTANCE TESTING

29

RESULT

30

9

9.1 PERFORMANCE METRICS

30

ADVANTAGES AND DISADVANTAGES

33

10

CONCLUSION

34

11

FUTURE SCOPE

34

12

APPENDIX

35

13

13.1 SOURCE CODE

35

13.2 GITHUB & PROJECT DEMO LINK

35

REFERENCES

36

CHAPTER 1ⁱⁱⁱ

INTRODUCTION

1.1 Project Overview

- The device will detect the animals and birds using the Clarifai service.
- If any animal or bird is detected the image will be captured and stored in the IBM Cloud object storage.
- It also generates an alarm and avoid animals from destroying the crop .
- The image URL will be stored in the IBM Cloudant DB service.
- The device will also monitor the soil moisture levels, temperature, and humidity values and send them to the IBM IoT Platform.
- The image will be retrieved from Object storage and displayed in the web application.
- A web application is developed to visualize the soil moisture, temperature, and humidity values Users can also control the motors through web application.

1.2 Purpose

An intelligent crop protection system helps the farmers in protecting the crop from the animals and birds which destroy the crop. This system also helps farmers to monitor the soil moisture levels in the field and also the temperature and humidity values near the field. The motors and sprinklers in the field can be controlled using the mobile application. Here to solve this situation we are proposing a solution using IOT(Internet of Things) where we use various types of sensors to monitor the entire field and using the help of the internet we tend to send the message to the farmer or the person who is responsible for solving the crisis that is currently occurring. The types of sensors we use will also give the information of the humidity level in the field, the temperature of the field, and detection of animals using their thermal radiation and also we process the information and give them in the form of graphs and images to the farmers for easy understanding.

CHAPTER 2 LITERATURE SURVEY 2.1 Existing Problem

Most of the farmers are facing many problems nowadays due to many reasons. Our problem to solve is the invasion of various species such as birds and animals that harm the crops that are being cultivated. Various types of species such as birds and animals come to the cultivation field according to the crop that is being cultivated and also according to the season of cultivation. Some wild animals enter the field during night times when the field is near a forest region or when the farm cultivates some fruits and other crops that attract animals. Some animals cross the field in search of food and water and also the birds enter the field for food and they damage all the crops. When the animals enter the field they not only eat food but they also damage the entire field by walking upon the crops and also by spoiling the food crops. The birds, by entering the field they come to eat seeds of the crops and also they tend to drag the crops and ruin the entire field. Some birds enter the field to eat the insects and pests in the field.

2.2 REFERENCES

Shishir Bagal , Krunal Mahajan , Riya Parate , Ekta Zade , Shubham Khante (2021) have investigated the title of “Smart Crop Protection System Using IOT” . The Smart protection system defines that this project help to farmer for the protection of a farm. We have designed this project for the only secure from animals but we this project have the provision to secure from the human begins also. This can achieve by the help of IOT device that we are discuss in this paper. The SCPS work on the battery so that this project can be easily portable and also we are add solar panels and converter modules this can help the battery to charge from solar energy. The IOT device is used to indicate the farmer by a message while someone enter into the farm and we are used SD card module that helps to store a specified sound to fear the animals.

2.3 Problem Statement Definition

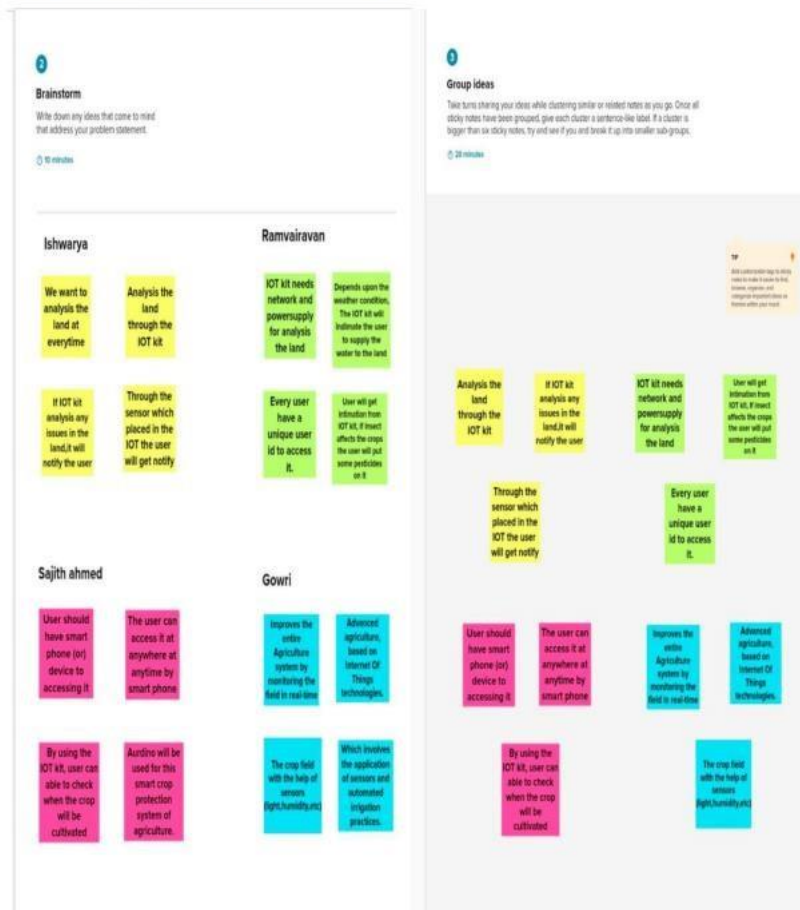
Most of the farmers are facing many problems nowadays due to many reasons. Our problem to solve is the invasion of various species such as birds and animals that harm the crops that are being cultivated. Various types of species such as birds and animals come to the cultivation field according to the crop that is being cultivated and also according to the season of cultivation. Some wild animals enter the field during night times when the field is near a forest region or when the farm cultivates some fruits and other crops that attract animals.

CHAPTER 3 IDEATION AND PROPOSED SOLUTION 3.1 Empathy Map Canvas

3.2 Ideation & Brainstroming

Crops in farms are many times ravaged by local animals like buffaloes, cows, goats, birds, and fire etc. This leads to huge losses for the farmers. It is not possible for farmers to barricade entire fields or stay on field 24 hours and guard it. So here we propose automatic crop protection system from animals and fire. This is aarduino Uno based system using microcontroller. This system uses a motion sensor to detect wild animals approaching near the field and smoke sensor to detect the fire. In such a case the sensor signals the microcontroller to take action.If there is a smoke, it immediately turns ON the motor. This ensures complete safety of crops from animals and from fire thus protecting the farmer's loss. This is aarduino. Uno based system using microcontroller. This system uses a motion sensor to detect wild animals approaching near the field and smoke sensor to detect the fire. In such a case the sensor.

Step-2: Brainstorm, Idea Listing and Grouping



Step-3: Idea Prioritization

4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

⌚ 30 minutes



3.3 Proposed Solution

Moisture sensor is interfaced with Arduino Microcontroller to measure the moisture level in soil and relay is used to turn ON and OFF the motor pump for managing the excess water level. It will be updated to authorities through IOT. Temperature sensor connected to microcontroller is used to monitor the temperature in the field. The optimum temperature required for crop cultivation is maintained using sprinklers. IOT based fertilizing methods are followed, to minimize the negative effects on growth of crops while using fertilizers.

The PIR sensor and UV sensors detect the motion of animals and birds for a particular arrange. The thermal radiation temperature of humans at different ages is fed to the system so there won't be any false alarm. If any invasion of animals is found, the camera focuses on the region and the processed image is sent to the farmer . After seeing the image of the animal that entered, they can decide to take any actions. A fence is built around the field to prevent large

animals from entering where the sensors are placed at all the corners of the field fully covering the entire region.

**Project Design Phase-I
Proposed Solution Template**

Date	29 September 2022
Team ID	PNT2022TMID45497
Project Name	Project - IoT based smart crop protection system for Agriculture
Maximum Marks	2 Marks

Proposed Solution Template:

Project team shall fill the following information in proposed solution template.

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Develop affordable app-based solution for Soil health monitoring and suggest which crop to be sown based on it. (Technology Bucket: IoT, AI, ML etc.)
2.	Idea / Solution description	<p>Create app-based solution to detect soil parameters like moisture content, temperature, relative humidity, nutrient, Ph, CEC, NPK etc. and provide crop suggestions to be produced based on soil parameters & environment values. Bonus Objective: Provide remedies & alerts on soil deficiencies like Watering for low Moisture level, Fertilizers for Nutrient deficiencies</p> <p>TECHNICAL ARCHITECTURE</p> <pre> graph LR IoT[IoT Device] --> Watson[IBM Watson IoT Platform] Watson --> NodeRED[Node-RED] NodeRED --> WebUI[Web UI] NodeRED --> User((User)) NodeRED --> ObjectStorage[Object Storage] NodeRED --> CloudantDB[(Cloudant DB)] Python[Python Code random data] --> NodeRED clarifai[clarifai] --> NodeRED </pre>
3.	Novelty / Uniqueness	<ul style="list-style-type: none"> Currently farmers follow Traditional Crop yielding pattern and irrespective of soil condition, farmers take routine crops. Farmers irrespective of whether soil nutrient requirement uses blanket fertilizers for crop.
4.	Social Impact / Customer Satisfaction	Agribusiness required the devotion of numerous regular asset including, land, water, and ecological condition, The quality and amount of characteristic asset has debased throughout the years because of monetary

3.4 PROBLEM-SOLUTION FIT

Define CS, fit into CC	1.CUSTOMER SEGMENTS CS Who is your customer? i.e., the customers are farmers <div>Customer are consuming who are all ready to buy their needs.</div>	5.AVAILABLE SOLUTIONS AS Which solutions are available to the customers when they face the problem? i.e., they should know the climate change and soil strength <div>If they knew the soil erosion of climate change, they can solve by using different kind of seeds on their land based on the climate</div>	8.CHANNELS OF BEHAVIOUR BH 8.1 ONLINE What kind of actions do customers take online? <div>The kind of action in online has some connection issues to the IOT kit and device.</div> 8.2 OFFLINE What kind of actions do customers take offline? <div>The kind of action in online is to get yield of crop and profit.</div>	Explore AS, differentiate into BE, understand
	2.JOBS TO BE DONE/PROBLEMS J&P Which jobs to be done (problems) do you address for your customers? i.e., there could be more crop are affected <div>customer recommended to put pesticide on their crops before it gets affected.</div>	6.CUSTOMER CONSTRAINTS CC Which constraints prevent your customer from taking action or limit of solutions? i.e., lack of knowledge in agriculture <div>The customer needs a proper internet connection and IoT kit.</div>	9.PROBLEM ROOT CAUSE RC What is the real reason that this problem exists? i.e., customer know to use the techniques for production <div>The root cause is to overcrowding in agriculture and poor techniques of production, by this every problem will be solved.</div>	
	3.TRIGGERS TR What triggers customers to act? i.e., experiencing the issues on land <div>There are many ways to refer to modern agriculture</div> 4.EMOTIONS: BEFORE/AFTER EM How do customers feel when they feel a problem? i.e., loss of money and mentally insecure <div>Sometimes the IoT devices may not work properly due to some technical issues.</div>	7.BEHAVIOUR BE What does your customer do to address the problem and get the job done? i.e., find the disease, analyse the land <div>Cone with climate change, soil erosion and biodiversity loss. Meet demand for moon food of higher quality</div>	10.YOUR SOLUTION SL If you are working on the agriculture land Using IoT sensor, gather the relevant details <div>Agriculture irrigation control stays unique of the determined significant interests in agriculture. The simulation result describes the aqua utilization according to the field parameters in the cultivation field. Guideline of horticultural water system stays restrictive to the set up significant interests of farming</div>	

CHAPTER 4

REQUIREMENT ANALYSIS

4.1 Functional Requirement

Following are the functional requirements of the proposed solution.

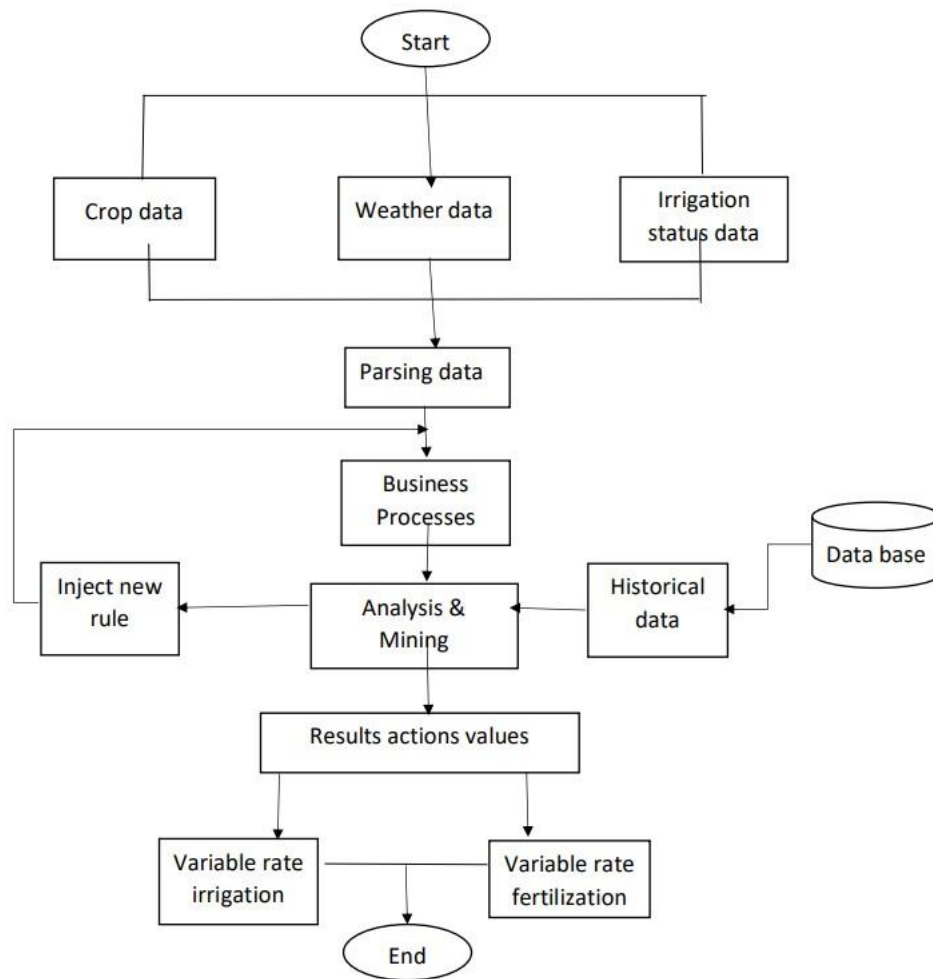
- User Registration ,Registration through Form Registration through Gmail Registration through LinkedIN
- User Confirmation ,Confirmation via Email Confirmation via OTP
- Tracking Expense Helpful insights about money management
- Alert Message Give alert mail if the amount exceeds the budget limit Category This application shall allow users to add categories of their expenses

4.2 Non Functional requirement

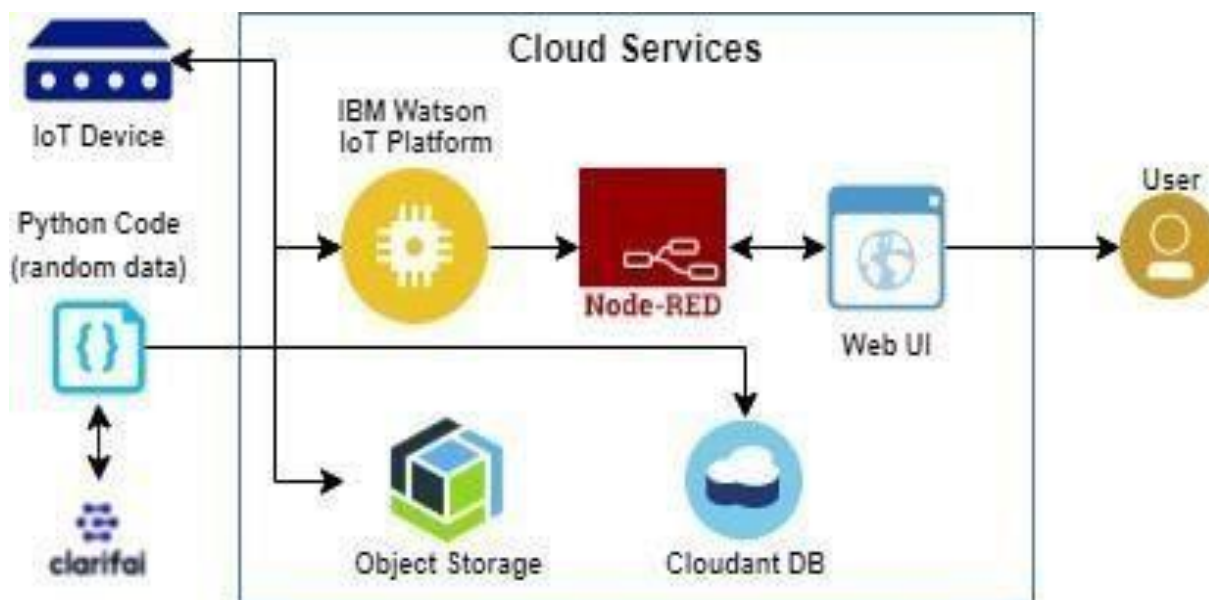
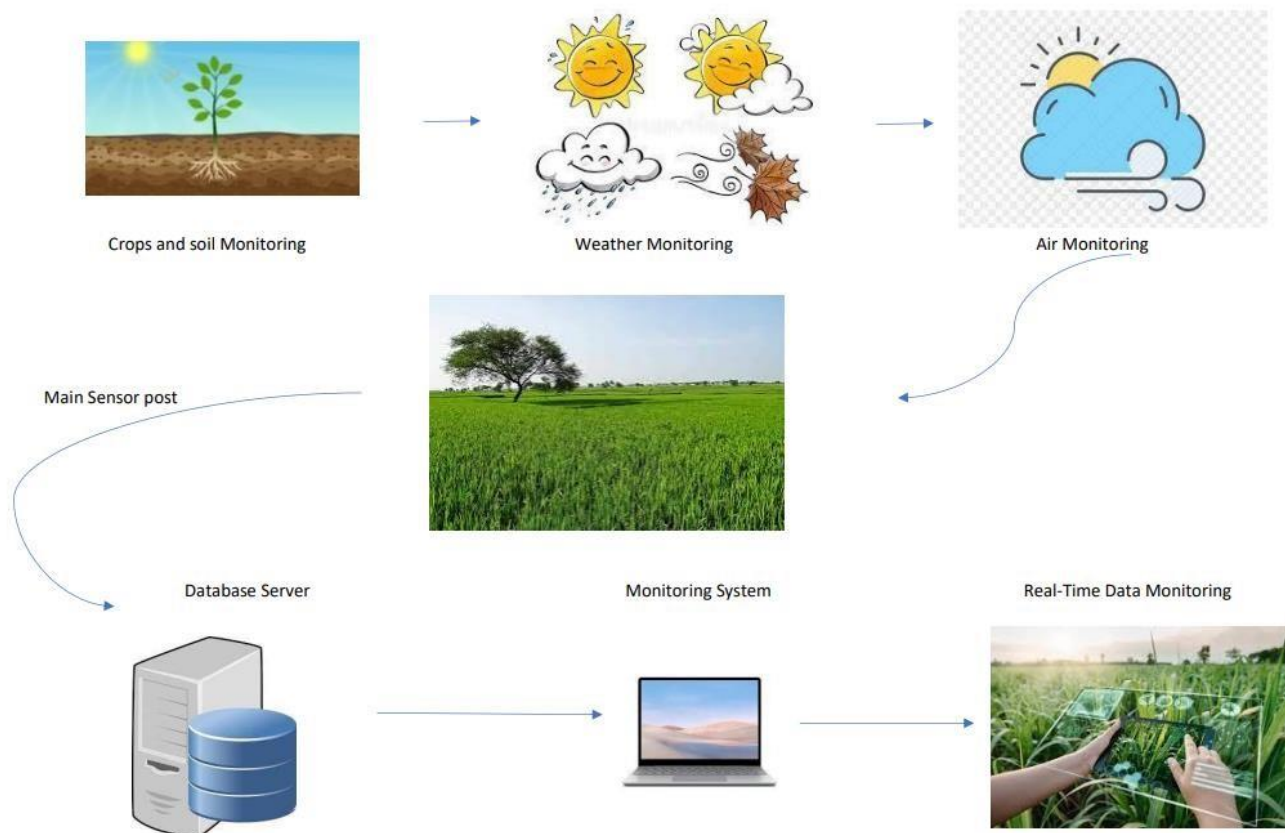
Following are the non-functional requirements of the proposed solution.

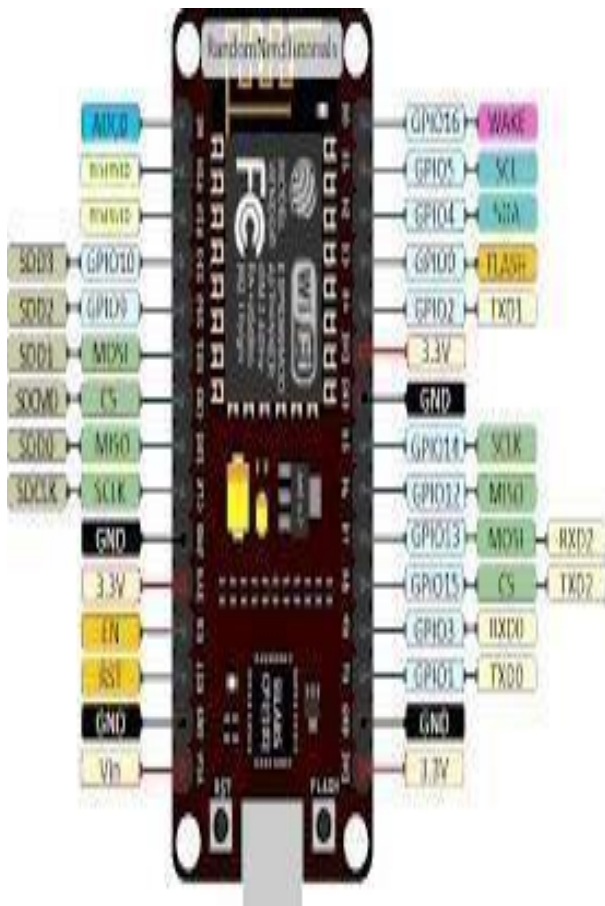
- Usability You will be able to allocate money to different priorities and also help you to cut down on unnecessary spending
- Security More security of the customer data and bank account details.
- Reliability Used to manage his/her expense so that the user is the path of financial stability. It is categorized by week, month, and year and also helps to see more expenses made. Helps to define their own categories.
- NFR-4 Performance The types of expense are categories along with an option .Throughput of the system is increased due to light weight database support.
- NFR-5 Availability Able to track business expense and monitor important for maintaining healthy cash flow. NFR-6 Scalability The ability to appropriately handle increasing demands.

CHAPTER 5 PROJECT DESIGN 5.1 DATA FLOW DIAGRAM



5.2 Technical Architecture





5.3 USER-STORIES

User Stories

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story/Task	Acceptance criteria	Priority	Release
Employee dashboard	Registration	USN-1	As an employee, I can register for the application by entering my email, password, and confirming my password	I can access my account/dashboard	High	Sprint-1
	Login	USN-2	As an employee, I can login to the application by entering correct email	I can access my account/dashboard	High	Sprint-2
o	Dashboard	USN-3	As an employee, according to my role, I will get notification about my task	I get the information about what I have to do in monsoons	High	Sprint-1
	Forget password	USN-4	As an employee, I can reset my password by this option in case I forgot my old password	I get access to my account again	Medium	Sprint-2
	Know more	USN-5	As a employee, I will be guided by expertise through online session once in a week about how to take care of the plant and all	Know something more	Low	Sprint-3
	Help me	USN-6	As an employee, I can post my problems and will get solution from Expertise	I can ask my query and all	High	Sprint-1
	Feedback	USN-7	As a user, if I faced any problem while using the app or want to give some suggestion about the app. That I can do by posting my issues in feedback	I can teel my problems	Medium	Sprint-2

CHAPTER 6 PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint1	SensorData (python script)	USN-1	The Data of sensor which are feed to the Raspberrypi. Here we are using python script to generate a random sensor data.	3	High	Jeya Surya (Teamleader)
Sprint1	Automation (python script)	USN-2	Some activities are made to automation to overcome insufficient of labour force in the field. Hence that also included in python script to implement automation .	5	High	Vigneshwaran (Team Member)
Sprint2	IBM IOT platform	USN-3	To send the raspberrypi data to IOT platform, we create an IBM IOT platform and connect the raspberry pi to the device created in IBM IOT.	5	High	Logamagesh (Team Member)

Sprint3	Node RED service	USN-4	To access the IBM IOT platform from external application or from external UI Node red service is established.	5	High	Vigneshwaran (Team Member)
Sprint3	API Key	USN-5	To protect the IBM IOT platform creating an API Key.		High	Damodharan (Team Member)
Sprint4	User Application	USN-6	To monitor and control the field sensors the User is provided with an User application created by MIT app inventor	8	High	Jeya Surya (Team Leader) Logamagesh (Team Member)

6.2 SPRINT DELIVERY SCHEDULE

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	8	6 Days	24 Oct 2022	29 Oct 2022	8	29 Oct 2022
Sprint-2	5	6 Days	31 Oct 2022	05 Nov 2022	5	05 Nov 2022
Sprint-3	8	6 Days	07 Nov 2022	12 Nov 2022	8	12 Nov 2022
Sprint-4	8	6 Days	14 Nov 2022	19 Nov 2022	8	19 Nov 2022

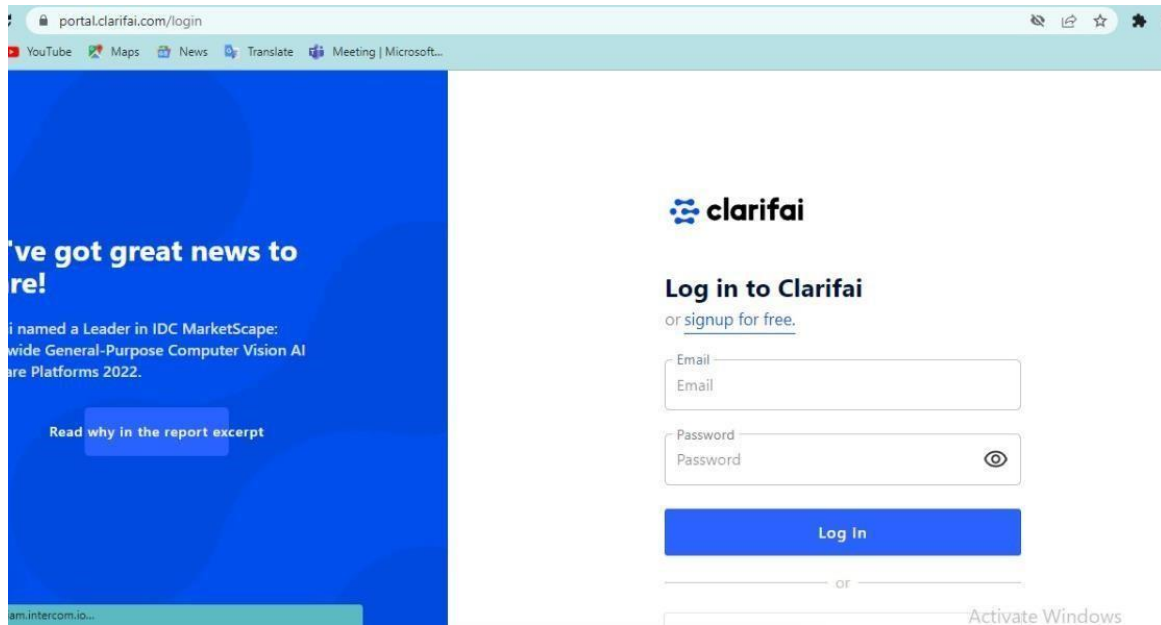
6.3 REPORT FROM JIRA REQUIRED SOFTWARE

- **CLARIFAI**
- **IBMWATSONIOTPLATFORM**
- **PYTHONIDLE**
- **NODERED**
- **MITAPPINVENTOR CLARIFAI:**

Clarifai provides an end-to-end platform with the easiest to use UI and API in the market. Clarifai Inc. is an artificial intelligence (AI) company that specializes in computer vision and uses machine learning and deep neural networks to identify and analyse images and videos. The company offers its solution via API, mobile SDK, and on-premise solutions.

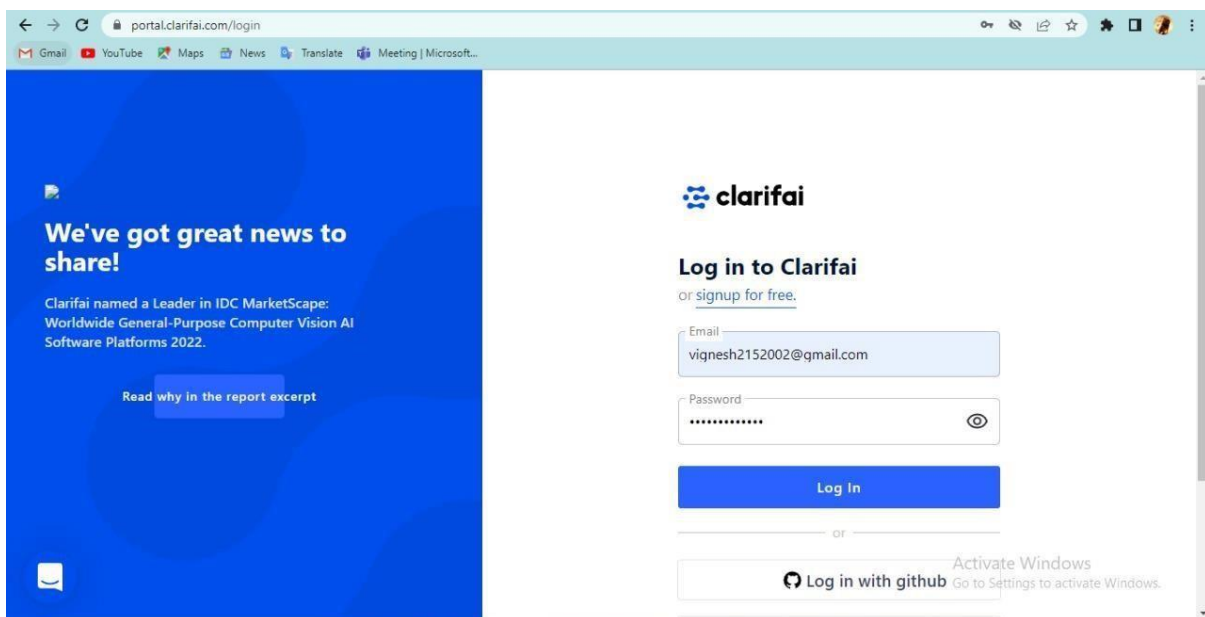
STEP 1:

- Open Clarifai portal in web browser.



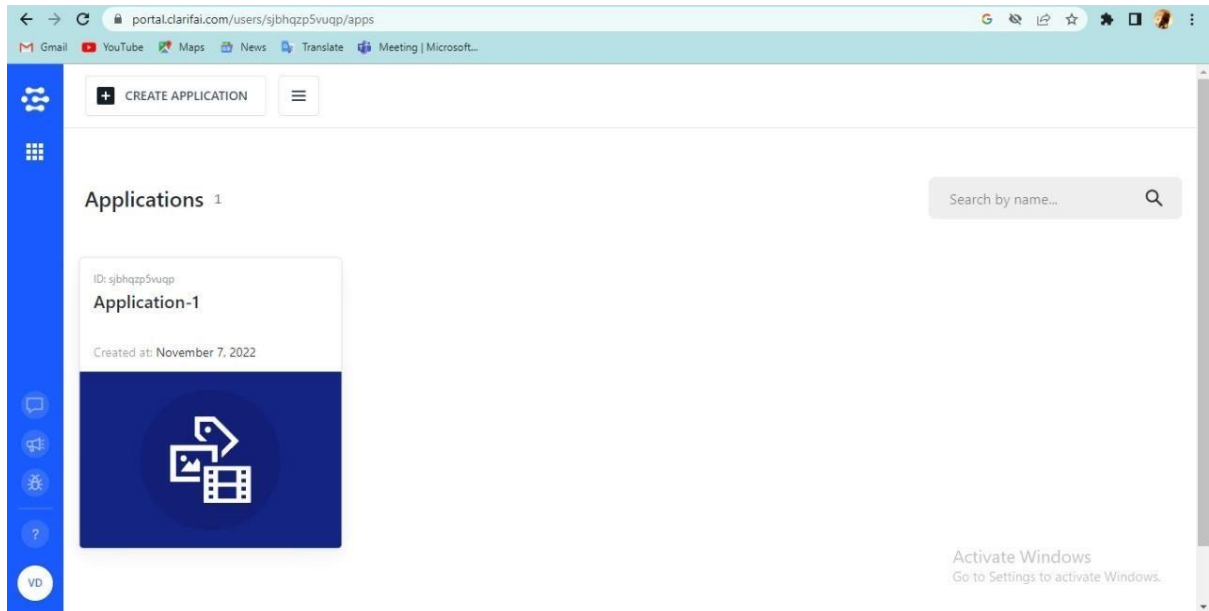
STEP 2:

- Signup using the required user mail and password.



STEP 3:

- Finally, Created an account



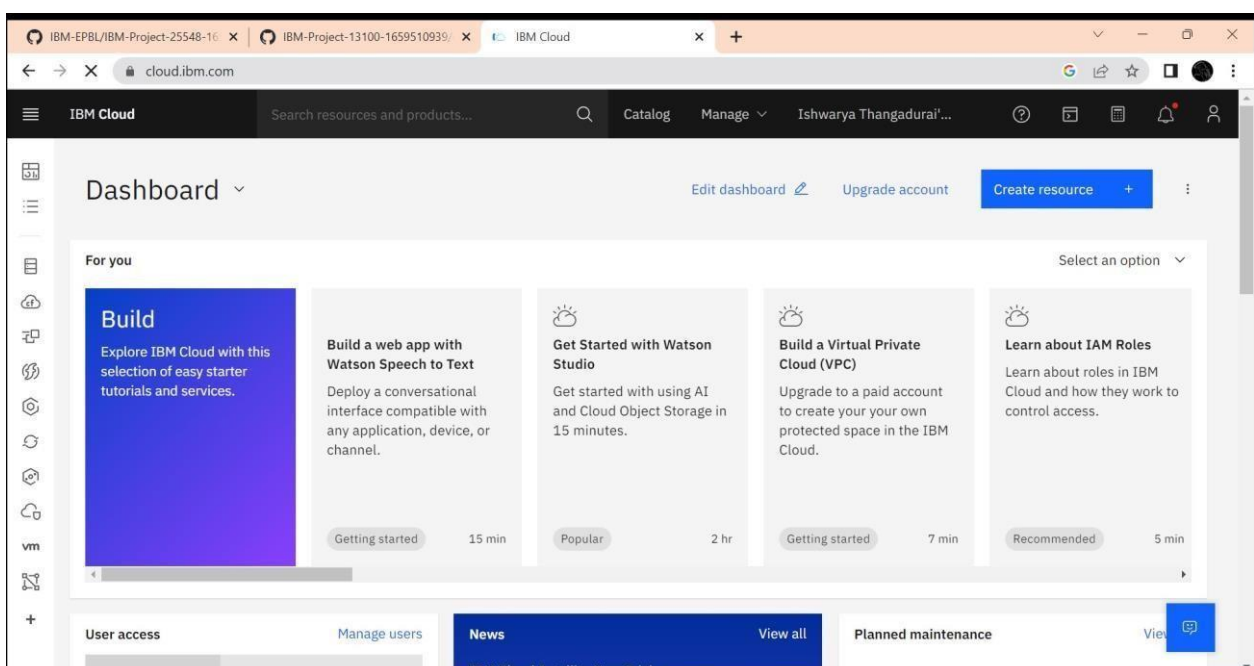
IBM WATSON IoT PLATFORM:

We need to have basic knowledge of the following cloud services:

- IBM Watson IoT Platform
- Node-RED Service
- Cloudant DB

We need to create an IBM Cloud Account to complete this project.

LOGIN:



The screenshot shows the IBM Cloud Dashboard for user Gowri S. The top navigation bar includes the IBM Cloud logo, a search bar, and links to Catalog, Manage, and the user's account. The dashboard features a 'For you' section with four cards: 'Build' (purple), 'Build a web app with Watson Speech to Text' (15 min), 'Get Started with Watson Studio' (2 hr), and 'IBM Watson Internet of Things Platform' (2 min). A 'Log out' button is visible in the user profile dropdown. The bottom of the dashboard shows 'User access', 'News', and 'Planned maintenance' sections. The Windows taskbar at the bottom displays the date as 19-11-2022 and the time as 09:55.

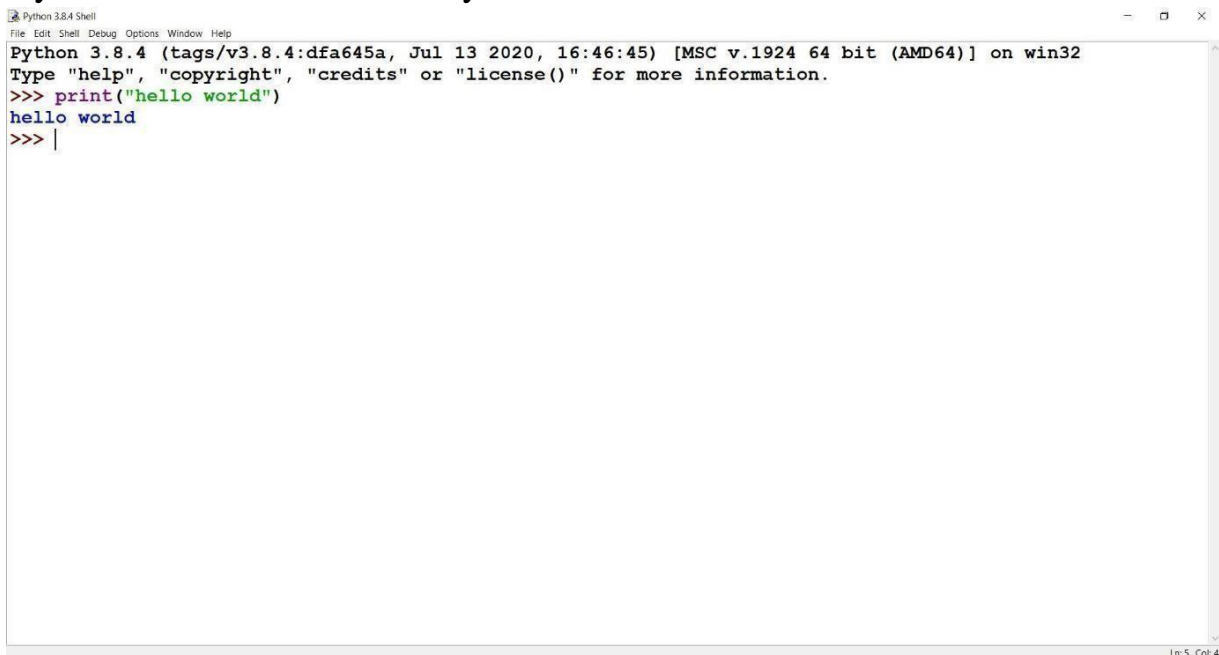
The screenshot shows the IBM Cloud Account settings page for user Vigneshwaran D. The left sidebar contains a navigation menu with options like Account, Account resources, and Account settings. The main content area displays account details: Account (Vigneshwaran D, ID: 80ad3cd9b9ec4cee9e023e6c0712dd01), Account Type (Trial (Free), 390 days remaining in Trial), and Account upgrade options (Pay-As-You-Go, Subscription, and Need help?). A 'Contact sales' link is provided under the 'Need help?' section. The Windows taskbar at the bottom shows the date as 19-11-2022 and the time as 09:55.

PYTHON IDLE INSTALISATION

Python is a computer programming language often used to build websites and software, automate tasks, and conduct data analysis. Python is a generalpurpose language, meaning it can be used to create a variety of different programs and isn't specialized for any specific problems.

STEP 1:

- Python is installed successfully

A screenshot of the Python 3.8.4 Shell window. The window title is 'Python 3.8.4 Shell'. The menu bar includes 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Window', and 'Help'. The main text area shows the following content: 'Python 3.8.4 (tags/v3.8.4:dfa645a, Jul 13 2020, 16:46:45) [MSC v.1924 64 bit (AMD64)] on win32', 'Type "help", "copyright", "credits" or "license()" for more information.', '>>> print("hello world")', 'hello world', and '>>> |'. The status bar at the bottom right indicates 'Ln: 5 Col: 4'.

STEP 2:

- The required python libraries are installed.
- Watson IoT Python SDK to connect to IBM Watson IoT Platform using python code is installed
- pip install wiotp-sdk

```
Command Prompt
Use quit() or Ctrl-Z plus Return to exit
>>> quit()

C:\Users\swast>pip --version
pip 20.1.1 from c:\users\swast\appdata\local\programs\python\python38\lib\site-packages\pip (python 3.8)

C:\Users\swast>pip install wiotp-sdk
Collecting wiotp-sdk
  Downloading wiotp-sdk-0.11.0.tar.gz (96 kB)
    |#####| 96 kB 130 kB/s
Collecting iso8601<=0.1.12
  Downloading iso8601-1.1.0-py3-none-any.whl (9.9 kB)
Requirement already satisfied: pytz<=2018.9 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from wiotp-sdk) (2020.1)
Collecting pyyaml<=3.13
  Downloading PyYAML-6.0-cp38-cp38-win_amd64.whl (155 kB)
    |#####| 155 kB 126 kB/s
Collecting paho-mqtt>=1.5.0
  Downloading paho-mqtt-1.6.1.tar.gz (99 kB)
    |#####| 99 kB 172 kB/s
Collecting requests>=2.21.0
  Downloading requests-2.28.1-py3-none-any.whl (62 kB)
    |#####| 62 kB 155 kB/s
Collecting requests-toolbelt<=0.8.0
  Downloading requests-toolbelt-0.10.1-py2.py3-none-any.whl (54 kB)
    |#####| 54 kB 281 kB/s
Collecting charset-normalizer<3,>=2
  Downloading charset-normalizer-2.1.1-py3-none-any.whl (39 kB)
Collecting idna<4,>=2.5
  Downloading idna-3.4-py3-none-any.whl (61 kB)
    |#####| 61 kB 40 kB/s
Collecting certifi<=2017.4.17
  Downloading certifi-2022.9.24-py3-none-any.whl (161 kB)
    |#####| 161 kB 261 kB/s
Collecting urllib3<1.27,>=1.21.1
  Downloading urllib3-1.26.12-py2.py3-none-any.whl (140 kB)
    |#####| 140 kB 177 kB/s
Building wheels for collected packages: wiotp-sdk, paho-mqtt
  Building wheel for wiotp-sdk (setup.py) ... done
  Created wheel for wiotp-sdk: filename=wiotp_sdk-0.11.0-py3-none-any.whl size=97110 sha256=7f29f0feae916d044b4c5368d6c81c3938d6a7416cb533cc52423a13021b571
  Stored in directory: c:\users\swast\appdata\local\pip\cache\wheels\46\al\69\352062eb129b1d46c4d32ec9d3835d3b5248ef4432dcfa3d9
  Building wheel for paho-mqtt (setup.py) ... done
  Created wheel for paho-mqtt: filename=paho-mqtt-1.6.1-py3-none-any.whl size=65428 sha256=6d15bdc481fc2e1dc91265a42f7ba6322395188013f91a32c5e225b108646
  Stored in directory: c:\users\swast\appdata\local\pip\cache\wheels\6a\48\01\c895c027e9b9367ec5470fbf371ee56e795a49ac6a19a4c9f
Successfully built wiotp-sdk paho-mqtt
Installing collected packages: iso8601, pyyaml, paho-mqtt, charset-normalizer, idna, certifi, urllib3, requests, requests-toolbelt, wiotp-sdk
Successfully installed certifi-2022.9.24 charset-normalizer-2.1.1 idna-3.4 iso8601-1.1.0 paho-mqtt-1.6.1 pyyaml-6.0 requests-2.28.1 requests-toolbelt-0.10.1 urllib3-1.26.12 wiotp-sdk-0.11.0
WARNING: You are using pip version 20.1.1; however, version 22.3 is available.
You should consider upgrading via the 'c:\users\swast\appdata\local\programs\python\python38\python.exe -m pip install --upgrade pip' command.

C:\Users\swast>
```

- Python client library for IBM Text to Speech is installed
- pip install --upgrade "ibm-watson>=5.0.0"

```
Command Prompt
C:\Users\swast>pip install --upgrade "ibm-watson>=5.0.0"
Collecting ibm-watson>=5.0.0
  Downloading ibm-watson-6.1.0.tar.gz (373 kB)
    |#####| 373 kB 142 kB/s
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing wheel metadata ... done
Collecting ibm-cloud-sdk-core<=3.16.0-py3-none-any.whl (83 kB)
  Downloading ibm-cloud-sdk-core-3.16.0-py3-none-any.whl (83 kB)
    |#####| 83 kB 152 kB/s
Requirement already satisfied, skipping upgrade: requests<3.0,>=2.0 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from ibm-watson>=5.0.0) (2.28.1)
Collecting websocket-client<1.1.0
  Downloading websocket-client-1.1.0-py2.py3-none-any.whl (68 kB)
    |#####| 68 kB 195 kB/s
Requirement already satisfied, skipping upgrade: python-dateutil>=2.5.3 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from ibm-watson>=5.0.0) (2.8.1)
Collecting PyJWT<3.0.0,>=2.4.0
  Downloading PyJWT-2.6.0-py3-none-any.whl (20 kB)
Requirement already satisfied, skipping upgrade: urllib3<2.0.0,>=1.26.0 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from ibm-cloud-sdk-core<=3.16.0-py3-none-any.whl) (1.26.12)
Requirement already satisfied, skipping upgrade: certifi<=2017.4.17 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0,>=2.0->ibm-watson>=5.0.0) (2022.9.24)
Requirement already satisfied, skipping upgrade: idna<4,>=2.5 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0,>=2.0->ibm-watson>=5.0.0) (3.4)
Requirement already satisfied, skipping upgrade: charset-normalizer<3,>=2 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0,>=2.0->ibm-watson>=5.0.0) (2.1.1)
Requirement already satisfied, skipping upgrade: six>=1.5 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from python-dateutil>=2.5.3->ibm-watson>=5.0.0) (1.15.0)
Building wheels for collected packages: ibm-watson
  Building wheel for ibm-watson (PEP 517) ... done
  Created wheel for ibm-watson: filename=ibm-watson-6.1.0-py3-none-any.whl size=370748 sha256=50648bc1c54ee0ba24e5cc521f68536db77a9cf975fccc975b9d49ba6956
  Stored in directory: c:\users\swast\appdata\local\pip\cache\wheels\34\b4\cd\829a351c802b7a578115fe7ddaedf62b29cae84e90882c7e2
Successfully built ibm-watson
Installing collected packages: PyJWT, ibm-cloud-sdk-core, websocket-client, ibm-watson
Successfully installed PyJWT-2.6.0 ibm-cloud-sdk-core-3.16.0 ibm-watson-6.1.0 websocket-client-1.1.0
WARNING: You are using pip version 20.1.1; however, version 22.3 is available.
You should consider upgrading via the 'c:\users\swast\appdata\local\programs\python\python38\python.exe -m pip install --upgrade pip' command.

C:\Users\swast>
```

- Required Libraries for cloud object storage is installed
- pip install ibm-cos-sdk

```
Command Prompt
C:\Users\swast>pip install ibm-cos-sdk
Collecting ibm-cos-sdk
  Downloading ibm-cos-sdk-2.12.0.tar.gz (55 kB)
    |#####| 55 kB 411 kB/s
Collecting ibm-cos-sdk-core==2.12.0
  Downloading ibm-cos-sdk-core-2.12.0.tar.gz (956 kB)
    |#####| 956 kB 251 kB/s
Collecting ibm-cos-sdk-s3transfer==2.12.0
  Downloading ibm-cos-sdk-s3transfer-2.12.0.tar.gz (135 kB)
    |#####| 135 kB 242 kB/s
Collecting jmespath<1.0.0,>=0.10.0
  Downloading jmespath-0.10.0-py2.py3-none-any.whl (24 kB)
Collecting python-dateutil<3.0.0,>=2.8.2
  Downloading python-dateutil-2.8.1-py2.py3-none-any.whl (247 kB)
    |#####| 247 kB 261 kB/s
Requirement already satisfied: requests<3.0,>=2.27.1 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (2.28.1)
Requirement already satisfied: urllib3<1.27,>=1.26.9 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (1.26.12)
Requirement already satisfied: six<=1.5 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from python-dateutil<3.0.0,>=2.8.2->ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (1.15.0)
Requirement already satisfied: certifi<=2017.4.17 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0,>=2.27.1->ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (2022.9.24)
Requirement already satisfied: charset-normalizer<3,>=2 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0,>=2.27.1->ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (3.4)
Requirement already satisfied, skipping upgrade: idna<4,>=2.5 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0,>=2.27.1->ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (2.1.1)
Building wheels for collected packages: ibm-cos-sdk, ibm-cos-sdk-core, ibm-cos-sdk-s3transfer
  Building wheel for ibm-cos-sdk (setup.py) ... done
  Created wheel for ibm-cos-sdk: filename=ibm-cos-sdk-2.12.0-py3-none-any.whl size=73926 sha256=a6f65ca073b69209a285e7f6b185c5bfaf721a71f35188f94c734e01cd36e
  Stored in directory: c:\users\swast\appdata\local\pip\cache\wheels\21\5f\fd\6a04fb45aad71bc8b300834368f9d39ef7c4fd1869d2244d
  Building wheel for ibm-cos-sdk-core (setup.py) ... done
  Created wheel for ibm-cos-sdk-core: filename=ibm-cos-sdk-core-2.12.0-py3-none-any.whl size=562952 sha256=c7f8e9dee7f311d484073c5082533731d8715bd59fbddeda4d6d38a3f99d7
  Stored in directory: c:\users\swast\appdata\local\pip\cache\wheels\ca\3d\78\48c57e7477898f3accbc1d482b1de8a8c8fb8c9d6d60ee7
  Building wheel for ibm-cos-sdk-s3transfer (setup.py) ... done
  Created wheel for ibm-cos-sdk-s3transfer: filename=ibm-cos-sdk-s3transfer-2.12.0-py3-none-any.whl size=89769 sha256=67c5983a4ebdcb33db07cbc1d35d7216ebef83fec9e5f0275d9fe851ceb777
  Stored in directory: c:\users\swast\appdata\local\pip\cache\wheels\c0\7a\37\13b5ca7d27a29a1062a47c5bba1c2ff3832795b68c8bd46
Successfully built ibm-cos-sdk ibm-cos-sdk-core ibm-cos-sdk-s3transfer
Installing collected packages: jmespath, python-dateutil, ibm-cos-sdk-core, ibm-cos-sdk-s3transfer, ibm-cos-sdk
  Attempting uninstall: python-dateutil
    Found existing installation: python-dateutil 2.8.1
    Uninstalling python-dateutil-2.8.1:
      Successfully uninstalled python-dateutil-2.8.1
Successfully installed ibm-cos-sdk-2.12.0 ibm-cos-sdk-core-2.12.0 ibm-cos-sdk-s3transfer-2.12.0 jmespath-0.10.0 python-dateutil-2.8.2
WARNING: You are using pip version 20.1.1; however, version 22.3 is available.
You should consider upgrading via the 'c:\users\swast\appdata\local\programs\python\python38\python.exe -m pip install --upgrade pip' command.

C:\Users\swast>
```


- pip install -U ibm-cos-sdk

```

C:\Users\swast>pip install -U ibm-cos-sdk
WARNING: You are using pip version 20.1.1; however, version 22.3 is available.
You should consider upgrading via the 'c:\users\swast\appdata\local\programs\python\python38\python.exe -m pip install --upgrade pip' command.
C:\Users\swast>pip install -U ibm-cos-sdk
Requirement already up-to-date: ibm-cos-sdk in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (2.12.0)
Requirement already satisfied, skipping upgrade: ibm-cos-sdk-core==2.12.0 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from ibm-cos-sdk) (2.12.0)
Requirement already satisfied, skipping upgrade: ibm-cos-sdk-core==2.12.0 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from ibm-cos-sdk) (2.12.0)
Requirement already satisfied, skipping upgrade: jmespath<1.0.0,>=0.10.0 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from ibm-cos-sdk) (0.10.0)
Requirement already satisfied, skipping upgrade: requests<3.0,>=2.27.1 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (2.28.1)
Requirement already satisfied, skipping upgrade: urllib3<1.27,>=1.26.9 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (1.26.12)
Requirement already satisfied, skipping upgrade: python-dateutil<3.0.0,>=2.8.2 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (2.8.2)
Requirement already satisfied, skipping upgrade: charset-normalizer<3,>=2 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0,>=2.27.1->ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (2.1.1)
Requirement already satisfied, skipping upgrade: idna<3,>=2.5 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0,>=2.27.1->ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (3.4)
Requirement already satisfied, skipping upgrade: certifi<2017.4.17 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0,>=2.27.1->ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (2022.9.24)
Requirement already satisfied, skipping upgrade: six>=1.5 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from python-dateutil<3.0.0,>=2.8.2->ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (1.15.0)
WARNING: You are using pip version 20.1.1; however, version 22.3 is available.
You should consider upgrading via the 'c:\users\swast\appdata\local\programs\python\python38\python.exe -m pip install --upgrade pip' command.
C:\Users\swast>

```

- pip install boto3

```

C:\Users\swast>pip install boto3
Collecting boto3
  Downloading boto3-1.26.0-py3-none-any.whl (132 kB)
    |#####| 132 kB 148 kB/s
Collecting s3transfer<0.7.0,>=0.6.0
  Downloading s3transfer-0.6.0-py3-none-any.whl (79 kB)
    |#####| 79 kB 113 kB/s
Collecting botocore<1.30.0,>=1.29.0
  Downloading botocore-1.29.0-py3-none-any.whl (9.8 MB)
    |#####| 9.8 MB 2.2 MB/s
Requirement already satisfied: jmespath<2.0.0,>=0.7.1 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from boto3) (0.10.0)
Requirement already satisfied: urllib3<1.27,>=1.25.4 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from botocore<1.30.0,>=1.29.0->boto3) (1.26.12)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from botocore<1.30.0,>=1.29.0->boto3) (2.8.2)
Requirement already satisfied: six>=1.5 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from python-dateutil<3.0.0,>=2.1->botocore<1.30.0,>=1.29.0->boto3) (1.15.0)
Installing collected packages: botocore, s3transfer, boto3
Successfully installed boto3-1.26.0 botocore-1.29.0 s3transfer-0.6.0
WARNING: You are using pip version 20.1.1; however, version 22.3 is available.
You should consider upgrading via the 'c:\users\swast\appdata\local\programs\python\python38\python.exe -m pip install --upgrade pip' command.
C:\Users\swast>

```

- pip install resources


```
Command Prompt
C:\Users\swast>pip install resources
Collecting resources
  Downloading resources-0.0.1.tar.gz (3.7 kB)
Building wheels for collected packages: resources
  Building wheel for resources (setup.py) ... done
  Created wheel for resources: filename=resources-0.0.1-py3-none-any.whl size=4370 sha256=3811eb1ac96cbfb54f0f22303a68ee6aac97621e26ae94f9b2441ec318e
  Stored in directory: c:\users\swast\appdata\local\pip\cache\wheels\b3\ld\00\45ae97c7b92d145a0963f711c6d22f9af5306e74c88f2f28fd
Successfully built resources
Installing collected packages: resources
Successfully installed resources-0.0.1
WARNING: You are using pip version 20.1.1; however, version 22.3 is available.
You should consider upgrading via the 'c:\users\swast\appdata\local\programs\python\python38\python.exe -m pip install --upgrade pip' command.

C:\Users\swast>
```

- pip install cloudant

```
Command Prompt
You should consider upgrading via the 'C:\Users\swast\appdata\local\programs\python\python38\python.exe -m pip install --upgrade pip' command.

C:\Users\swast>pip install cloudant
Collecting cloudant
  Downloading cloudant-2.15.0-py3-none-any.whl (80 kB)
    80 kB 305 kB/s
Requirement already satisfied: requests<3.0.0,>=2.7.0 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from cloudant) (2.28.1)
Requirement already satisfied: charset-normalizer<3,>=2 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0.0,>=2.7.0->cloudant) (2.1.1)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0.0,>=2.7.0->cloudant) (1.26.12)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0.0,>=2.7.0->cloudant) (2022.9.24)
Requirement already satisfied: idna<4,>=2.5 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0.0,>=2.7.0->cloudant) (3.4)
Installing collected packages: cloudant
Successfully installed cloudant-2.15.0
WARNING: You are using pip version 20.1.1; however, version 22.3 is available.
You should consider upgrading via the 'c:\users\swast\appdata\local\programs\python\python38\python.exe -m pip install --upgrade pip' command.

C:\Users\swast>
```

PROJECT DEVELOPMENT

STEP 1: Write a python code for randomize Soil Moisture ,Temperature, Humidity and Animal detection.

STEP 2: Run the python code it send data to IBM IoT Watson Platform.

```

int
t=2;

int e=3;

void setup()
{
  Serial.begin(9600);
  pinMode(t,OUTPUT);
  pinMode(e,INPUT);
  pinMode(12,OUTPUT);
}

void loop()
{
  //ultrasonic sensor
  digitalWrite(t,LOW);
  digitalWrite(t,HIGH);
  delayMicroseconds(10);
  digitalWrite(t,LOW);
  float dur=pulseIn(e,HIGH);
  float dis=(dur*0.0343)/2;
  Serial.print("Distance is: ");
  Serial.println(dis);

  //LED ON
  if(dis>=100)
  {
    digitalWrite(8,HIGH);
    digitalWrite(7,HIGH);
  }

  //Buzzer For ultrasonic Sensor
  if(dis>=100)
  {
    for(int i=0; i<=30000; i=i+10)
    {
      tone(12,i);
      delay(1000);
      noTone(12);
      delay(1000);
    }
  }
}

```

The screenshot displays the IBM Watson IoT Platform interface. The main panel shows the 'Recent Events' tab for a device named 'abcd'. A table lists recent events with columns 'Event' and 'Value'. The events are from an 'IoTSensor' and contain JSON payloads with temperature, humidity, moisture, and animal detection data.

On the right, a 'Device Type: abcd' configuration window is open. It shows the 'Events' section with 'event_1' selected. The 'Schedule' is set to '20' and 'Every Minute'. The 'Payload' section shows a JSON template for the event data, with fields for temperature, humidity, moisture, and animal detection, each using a random value generator.

Event	Value
IoTSensor	{"temp":91,"Humid":67,"Moist":25,"Animal_dect":1}
IoTSensor	{"temp":102,"Humid":78,"Moist":92,"Animal_dect":1}
IoTSensor	{"temp":106,"Humid":69,"Moist":25,"Animal_dect":1}
IoTSensor	{"temp":92,"Humid":79,"Moist":82,"Animal_dect":1}

Event type name: event_1 [Send]

Schedule: 20 Every Minute

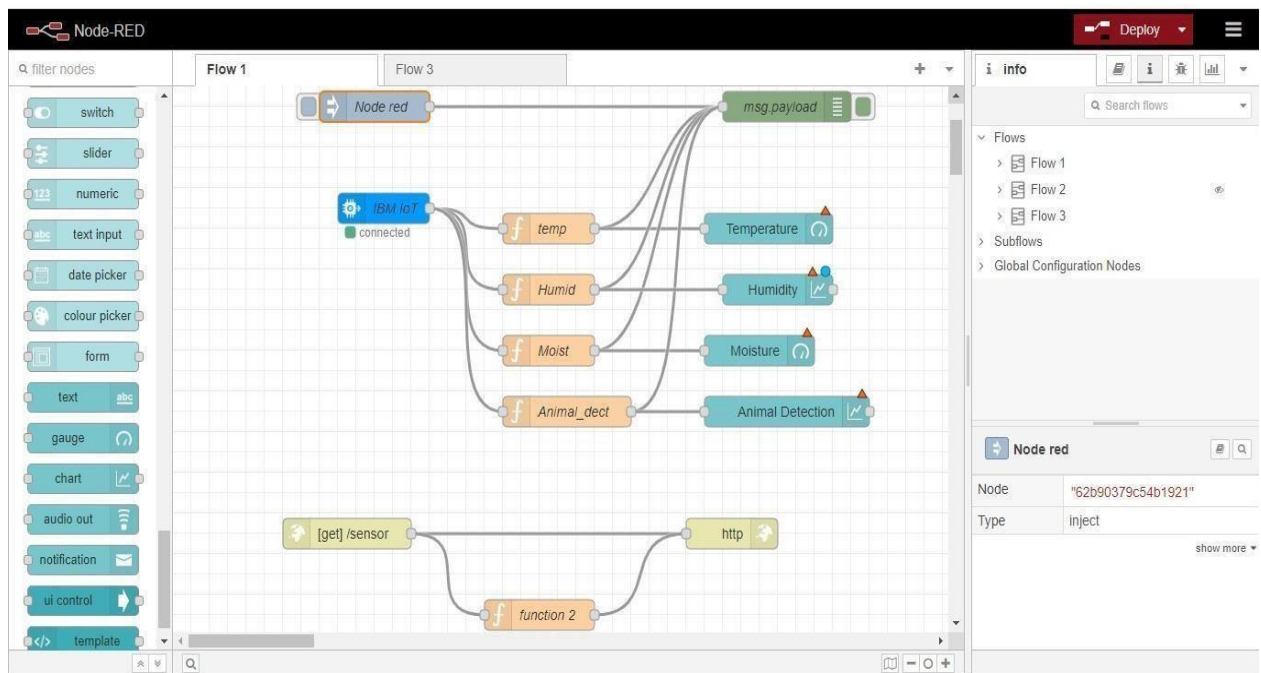
Payload:

```

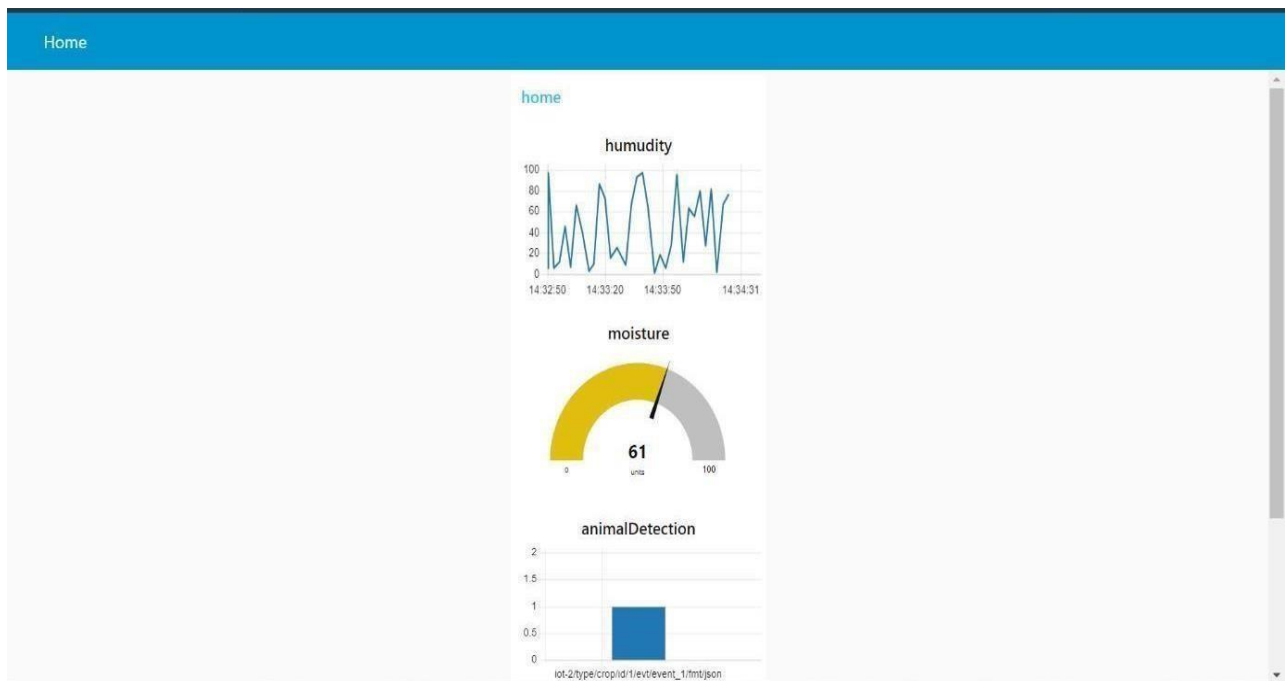
0 {
1   "temp": random(90,110)
2   "Humid": random(60,100)
3   "Moist": random(0,100)
4   "Animal_dect": random(0,2)
5
6 }

```

STEP 3: Open Node-RED flow dashboard.



STEP 4: Open Node-RED user interface to show the Soil Moisture, Humidity and Temperature value in gauge.



CHAPTER 7 CODING AND SOLUTIONS

7.1 FEATURE

Python code to generate random data and pass it to IBM Watson IoT platform

Source Code:

```
import time import sys import
ibmiotf.application import
ibmiotf.device import random #Provide your
IBM Watson Device
Credentialsorganization = "wu5b55" deviceType
= "crop1" deviceId = "1234" authMethod =
```

```
"token" authToken = "1234567890" #
```

```
Initialize GPIOtry:
```



```

        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:

```

```

    temp=random.randint(0,
100)
    Hum=random.randint(0,1
00)
    moisture=random.randint
(0,100)

```

```

        data = { 'temperature' : temp, 'Humidity': Hum, 'Moisture':moisture }
def myOnPublishCallback():

```

```

        print ("Temperature = " + str(temp)+" C Humidity = " +
str(hum)+ " moisture = " +str(moisture) + "to IBM Watson")

```

```

        success = deviceCli.publishEvent("IoTSensor",
"json", data, qos=0,on_publish=myOnPublishCallback) if not
        success:

```

```

            print("Not connected to IoTf")time.sleep(10) deviceCli.commandCallback =
myCommandCallback
# Disconnect the device and application from the clouddeviceCli.disconnect()

```

7.2 FEATURE 2

Source code is deployed on IBM Watson IoT platform to generate sensor data.SourceCode:

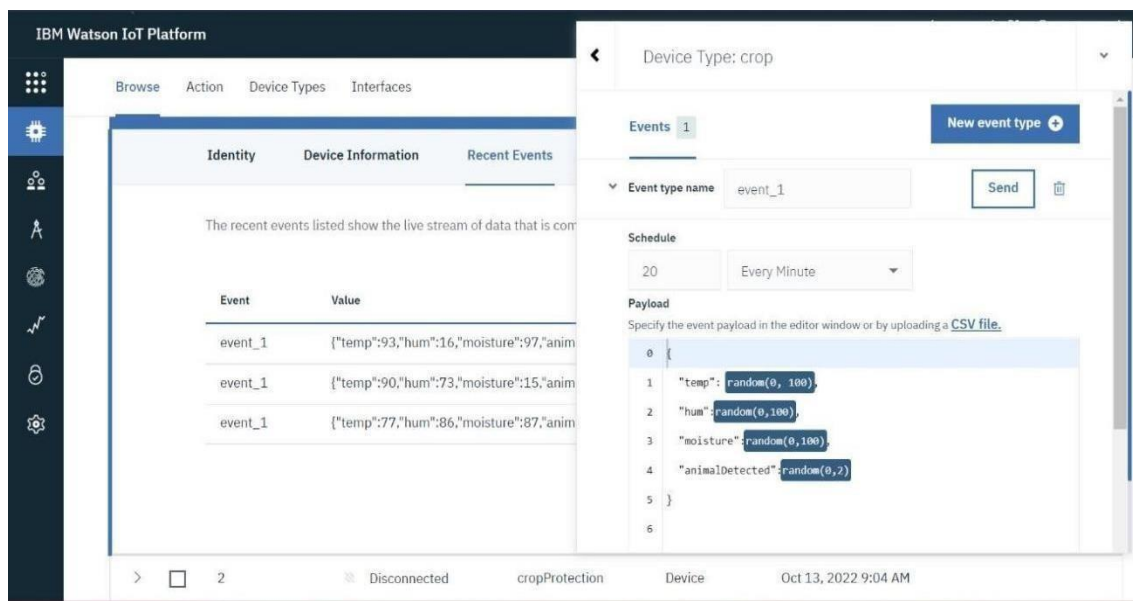
{

```

"temperature": random(0, 100),
"humidity": random(0, 100),
"moisture": random(0, 100),
"animalDetected":random(0,2)
}

```

Output:



7.3 DATABASE SCHEMA

PYTHON CODE TO IBM:

```

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

```

#Provide your IBM Watson Device Credentials organization =

"wu5b55" deviceType = "crop1" deviceId = "1234"

authMethod = "token" authToken = "1234567890" # Initialize

GPIO 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 thecloud as an event of type
"greeting" 10 times
deviceCli.connect() while
True:

    #Get Sensor Data from DHT11

    temp=random.randint(0,100) Hum=random.randint(0,100)
    moisture=random.randint(0,100)
    data = { 'temperature' : temp, 'Humidity': Hum, 'Moisture':moisture
    }

#print data
def myOnPublishCallback():
    print ("Temperature = " + str(temp)+" C Humidity = "
           + str(hum)+ " moisture = "
           + str(moisture) + "to IBM Watson")
    success = deviceCli.publishEvent("IoTSensor",
                                     "json", data,qos=0,
    on_publish=myOnPublishCallback)
    if not success:
        print("Not connected
              to IoT")
    time.sleep(10)

deviceCli.commandCallback = myCommandCallback

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

```


Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	10	4	2	3	20
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	11	2	4	20	37
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	5	2	1	8
Totals	24	14	13	26	77

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	0	0	7
Client Application	51	0	0	51
Security	2	0	0	2
Outsource Shipping	3	0	0	3

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

8.2 USER ACCEPTANCE TESTING

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

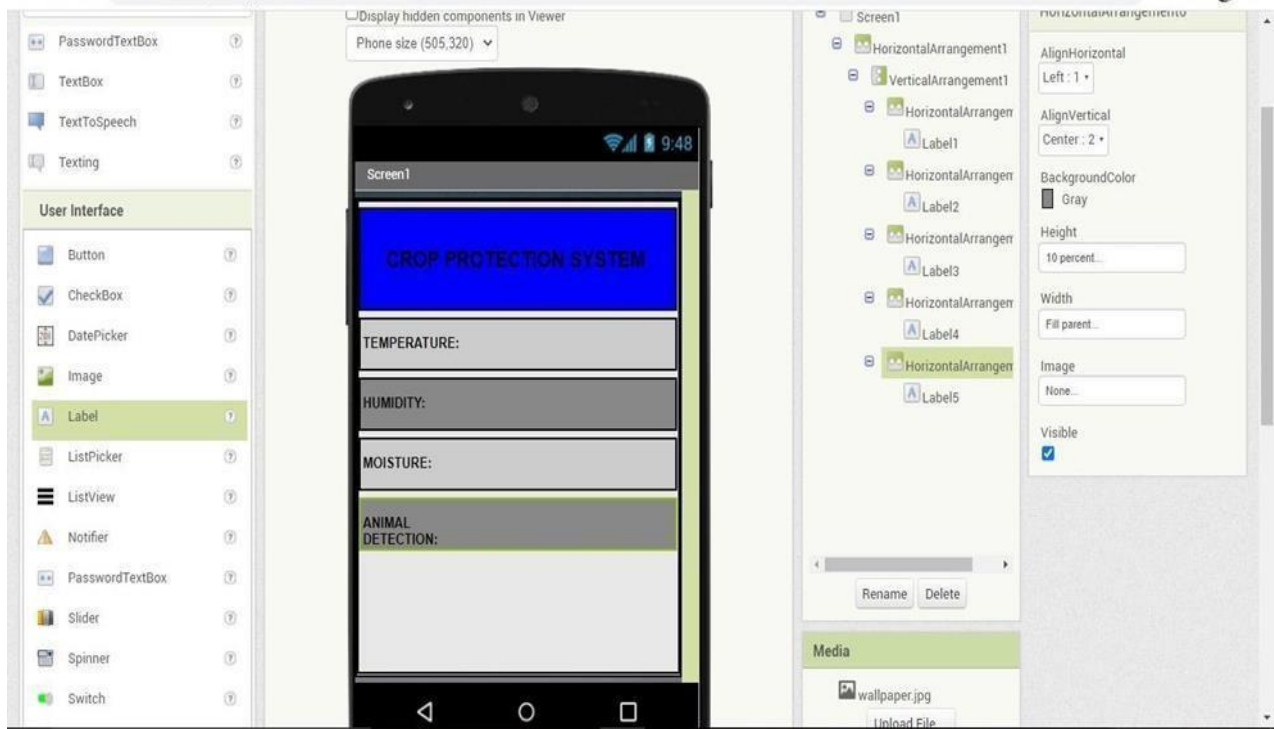
Exception Reporting	9	0	0	9
Final Report Output	4	0	0	4
Version Control	2	0	0	2

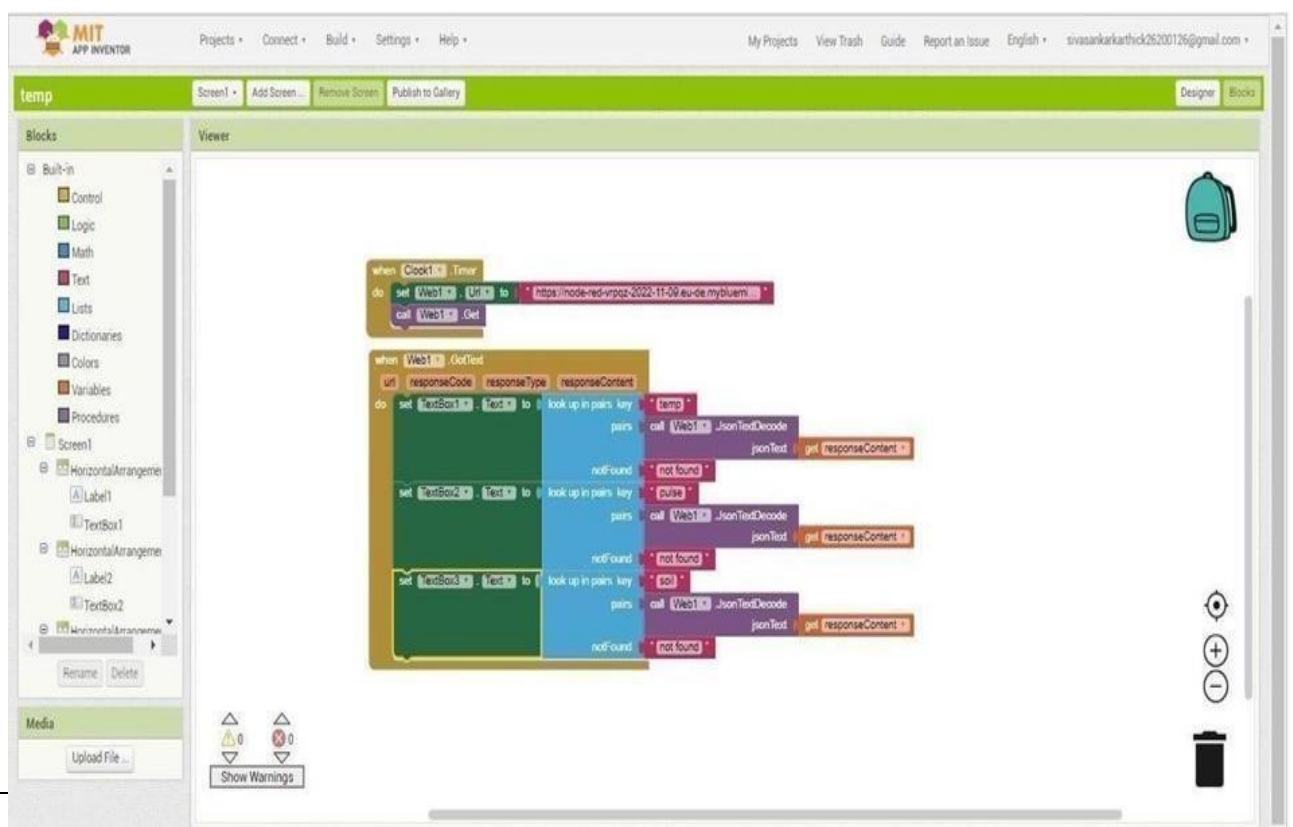
CHAPTER 9 RESULT

9.1 PERFORMANCE METRICS

MIT APP INVENTOR:

STEP 1: MIT APP inventor to design the APP.





STEP 2: Customize the App interface to Display the Values.

Screen1

CROP PROTECTION SYSTEM

TEMPERA
TURE:41

HUMIDITY:
80

MOISTURE
:61

ANIMAL
DETECTIO
N:0



ER 10

ISADVANTAGES

imals closely, even if they are physically

rove productivity and enable management of
note sensing.

far from access to the internet.

a reliably at any time from any location, so
ed monitoring system to be useless.

griculture is expensive.

sensors (light, humidity, temperature, soil

ductivity and pH)

CHAPTER 11 CONCLUSION

AS a result of this system, we can detect the changes in the field easily and intimate the farmers about it and also we can take precautions and do remedies accordingly. Here we use very low power consuming highly efficient components that give us accurate results and also they perform at low data rate conditions without any lag and help in finding the remedies. This crop protection system helps in detection of all kinds of external dangers and it saves time and money to the farmers before any loss that may occur. With the help of this system the farmers can be in a peaceful environment at ease without any pressure.

CHAPTER 12 FUTURE SCOPE

Study and analysis of the developed Crop protection systems for its costeffectiveness with the development of Arduino based variable frequency Ultrasonic birddeterrent circuit. outline of the crop damage caused by a particular Wild animal if thebehavioral features of the With the reduced cost in the smart phones.

CHAPTER 13

APPENDIX

13.1 SOURCE CODE

The source code has been uploaded in github. To refer the final source code click [“SOURCE CODE”](#)

13.2 GITHUB & PROJECT DEMO LINK

GITHUB LINK

The github link: "<https://github.com/IBM-EPBL/IBM-Project-25548-1659967044>

PROJECT DEMO LINK

The Project Demo link: <https://github.com/IBM-EPBL/IBM-Project-255481659967044>

CHAPTER 14

REFERENCE

[1]Priyanka Deotale, Prasad Lokulwar (2021) have presented the paper titled “Smart Crop Protection System from Wild animals Using IoT” . Crops in the agricultural land are destroyed by the domestic animals and wild animals ,it is one of the reason for low productivity . Farmers can't be there for entire 2 hours so we have make use of IOT to control the animals destroying the field . Once the animal is detected the system will larm and start lightning in the corner of the farm . It will not harm any animals and we can also protect the crops.

[2] N.S. Gogul Dev , K.S. Sreenesh , P.K. Binu (2019) has presented a paper titled “IoT Based Automated Crop Protection System” . Low productivity of crops is one of the main problems faced by the farmers in our country. This can be because of two main reasons. Crops destroyed by wild animals and because of bad weather condition. This paper provides a solution to the destruction of crops by animals. This system will provide a complete technical solution using the Internet of things (IOT) to the farmers to prevent their crops from wild animals and provide information to the farmers to maximize their production. Animals are detected using PIR sensors and cameras where animals are identified using TensorFlow image processing Techniques. Raspberry PI is used as the processing unit of the system and sound buzzers are used to emit the ultrasound frequencies