Siddaganga Institute of Technology, Tumakuru

(An Autonomous institution affiliated to Visvesvaraya Technological University, Belagavi,
Approved by AICTE, New Delhi, Accredited by NAAC and ISO 9001:2015 certified)

Embedded Android Solutions For Temperature Handling Using RFID Technology

A project report submitted to
Visvesvaraya Technological University. Belgaum, Karnataka
in the partial fulfillment of the requirements for the award of degree of
Bachelor of Engineering
in
Computer Science and Engineering
by

CHETHANA R	**1SI18CS407**
G KOTTRESHA	**1SI18CS409**
KIRAN B A	**1SI18CS411**
RAVICHANDRA B S	**1SI18CS416**

under the guidance of

Dr. SUMALATHA ARADHYA
Assistant Professor

Department of Computer Science & Engineering
(Program Accredited by NBA)
Siddaganga Institute of Technology
B.H Road, Tumakuru-572 103, Karnataka, India.
Web : www.sit.ac.in

July, 2021

Department of Computer Science and Engineering

Siddaganga Institute of Technology, Tumakuru

(An Autonomous institution affiliated to Visvesvaraya Technological University, Belagavi,
Approved by AICTE, New Delhi, Accredited by NAAC and ISO 9001:2015 certified)

Certificate

This is to certify that the Project Report entitled **”Embedded Android Solutions For Temperature Handling Using RFID Technology”** is a bonafide work carried out by **Chethana R(1SI18CS407)**, **G Kottresha(1SI18CS409)**, **Kiran B A(1SI18CS411)** and **Ravichandra B S(1SI18CS416)** in the partial fulfillment of the requirement for the award of the degree of Bachelor of Engineering in Computer Science and Engineering, Visvesvaraya Technological University, Belagavi during the year 2020-21. It is certified that all corrections/suggestions indicated for the internal assessment have been incorporated in the report.The project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the Bachelor of Engineering Degree.

....
Guide
Dr. Sumalatha Aradhya
Asst. Professor
Dept of CSE, SIT

....
Group Convener
Dr. Y S Nijagunarya
Professor
Dept of CSE, SIT

....
Dr. A S Poornima
Professor and Head
Dept of CSE, SIT

....
Dr. S V Dinesh
Principal
SIT, Tumakuru

Name of the Examiners   Signature with Date

1.

2.

Department of Computer Science and Engineering
Siddaganga Institute of Technology
Tumakuru - 572103

DECLARATION

I hereby declare that the entire work embodied in this dissertation has been carried out by me at **Siddaganga Institute of Technology** under the supervision of Dr. Sumalatha Aradhya. This dissertation has not been submitted in part or full for the award of any diploma or degree of this or any other University.

Chethana R 1SI18CS407
G Kottresha 1SI18CS409
Kiran B A 1SI18CS411
Ravichandra B S 1SI18CS416
Department of Computer Science and Engineering
Siddaganga Institute of Technology
Tumakuru - 572103

Acknowledgements

We consider this is a privilege to express a few words of gratitude to all those who guided and inspired us for successful completion of the project. We our humble pranams at the lotus feet of his holiness, Dr. Sree Sree Shivakumara swamigalu, Revered Founder, Sree Siddaganga Education Society, Sree Siddaganga Math for bestowing upon his blessings. We deem it as a privilege to thank Dr. M N Channabasappa, Director, SIT, Tumakuru and Dr. S V Dinesh, Principal, SIT, Tumakuru for fostering an excellent academic environment in this institute, which made this endeavor fruitful. We would like to express our sincere gratitude to Dr. A S Poornima, Professor and Head, Department of CSE, SIT, Tumakuru for her encouragement and valuable suggestions. We thank our guide Dr. Sumalatha Aradhya, Assistant Professor and group convener Dr. Y S Nijagunarya, Professor, Department of CSE, SIT, Tumakuru for valuable guidance,advice and encouragement. We are conscious of the fact that we have received cooperation in many ways from the teaching and non-teaching staff of Department of CSE and we express our gratitude to all who helped us in the completion of this project.

Abstract

This project implements a temperature monitoring and controlling system in the remote. Some real-time applications need a solution to control and monitor the temperature. For examples green farmhouses, cold storages, smart cooking induction stoves, and so on.

The IoT platform supports the implementation of this project by connecting to these applications. IoT devices get temperature and location information from the sensors send to the internet. IoT servers running in cloud service get data from IoT devices and store it in a database. The android smartphone help to monitor and control the device by receiving data from the device and sending commands to the device. We also locate the device using location details from an android phone. The android phone enables the user to monitor the status and temperature of the IoT device and control the device by sending a start or stop signal. We perform some analysis on data available in the database for future prediction using a machine learning algorithm.

In the existing advanced RFID technology, there is a passive tag available with built-in temperature sensors. The IoT device read a temperature from that tag by adding an RFID reader into the IoT device. The RFID passive tag operates at 13.56Mhz and is small in the size, we can adapt anywhere comfortable. Some of the applications already use RFID technology to control the temperature, but it is limited to the local area. We can provide a solution by connecting the existing system to remote by introducing cloud service.

The Amazon Web Services(AWS) provides cloud services for its users to build a complex digital computing system. We take a service from AWS to provide server functionality to our project. The server is responsible for receiving data from the device and store it in the database, sending the command to the device. The IoT device was implemented by using Raspberry Pi. The Raspberry Pi supports Linux version operating system and provided tools and functions to develop an IoT-based device. The IoT device sending data into the server and control the device.

After completion of the project, We can monitoring and control the IoT device from users android phones. The user receiving notifications when was device started and reaches its threshold value. We can monitor the status, location, and temperature of the device and update notification settings. Finally, analyze and visualize the previous data that help to control the device in different situations.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In the existing system such as buffet tray, dishes, or cold storage plant, an RFID technology is used with sensors to interact with sensors and control for their operations. In such a system there exist some devices or units, for example, induction stove in buffet try or storage unit in cold storage plants. The devices are responsible for producing heat or cold to control the operation of a system. The device reads the temperature from the sensor and stores it in the RFID tag. A controller is another device used to control the temperature in a system. The controller consists of an RFID reader and other user interface components. User interface component responsible displaying the current temperature and the threshold value of temperature and other kinds of information. The RFID reader enables a feature for the controller to control the temperature in the device by reading temperature data from the RFID tag. The controller performs some computations or operations based on temperature data to handle the device or unit.

When the controller enables RFID reader through the hardware interface pins. Once the RFID reader gets activated, it starts to emit RF signals from the antenna. The RFID tag is placed nearly to the RFID reader within the range of 5-10 meters. The RFID tag is a passive device that contains no battery. Some of the plants may use active or semi-active tags. In the case of passive tags, when the RF signals reach the RFID tag, the tag gets activated and starts sending back the data like UID and real-time temperature data to an RFID reader. RFID reader gathers data from a tag that will be redirected to the controller. While gathering data, the data is in the form of RF signals that are converted to digital forms, and displays on the user interfaces.

We have proposed a system that can handle the existing system remotely. We are connecting the RFID tag and RFID reader to Raspberry pi rather than a controller to provide a remote system through the IoT server interface, that can handle user commands with the hardware devices which are configured with. when the RFID reader sends the data to Raspberry pi, that data is being uploaded and stored in the AWS cloud via an IoT server. Users can view the stored data on android mobile phones with the help of a web server. The web server sends the Temperature data, location, and notification to users to know the status of temperature is being read from the sensor, and also it receives the command such as turn on/ turn offing the device. Users can view the location of the sensor and set a threshold limit for temperature value must be within, otherwise turning off the device. It shows the graph visualization for how the temperature is increasing and decreasing with respect to time.

1.1 Background Study

The concept of the project when we are thinking of IoT Devices,a real-time system for more industries working on IoT Platforms only. For the IoT devices, we are choosing Radio frequency identification(RFID). Using RFID, has several projects are done in real-time systems like Fastag, electronic passport system, Library automation using barcode reader, etc. We are also choosing it as the best technology in today's environment. We have studied some research papers and we are learning the RFID information about the RFID Tag and RFID Reader on its working concepts, design and hardware setup, all the information we are taken.

1.1.1 Motivation

In the existing system, the RFID tag is not active for a longer period, and data sent by the RFID tag is not in a readable format. The action needs to perform after the basic criteria are done by humans manually. So by adding these features into the existing project helps to human beings not to go further work than the primary work.

1.2 Related Works

Title of the work: RFID Technology and Its Applications in Internet of Things (IOT)
Authors: Xiaolin Jia1,2, Quanyuan Feng2, Taihua Fan1, Quanshui Lei1
Publication details: Southwest university of science and technology (researchgate.net)-2012
Page no: 02-07
Description: Radio Frequency Identification System (RFID) automated technology and assistive devices or computers to identify objects, record metadata, or control individual objectives with radio waves. RFID technology began to emerge in 1945, as a spy tool of the Soviet Union, which also sent radio frequency radio with detailed information. Similarly, the IFF (Identification Friend or Foe) transponder made in the United Kingdom was commonly used by World War II aircraft operators to acquire aircraft as a companion.

The standard RFID system consists of tags (senders/respondents) and readers (senders/recipients) A tag is a microchip connected to an antenna, which can be attached to an object as an identifier. The RFID reader communicates with the RFID marker using radio waves. The main advantage of RFID technology is the automatic identification and capture of data that promises more flexibility in the broader areas of business operations and aims to reduce the cost of already used systems such as bars.

As predicted in RFID it is one of the biggest opportunities in information technology, which will change the world more broadly and deeply. When RFID students adhere to the relevant rules of communication connected to the Internet, students distributed worldwide can view, track and monitor tagged items globally, automatically, and in real-time, if required. This is called the Internet of Things (IoT).

RFID tags: also known as transponders (transmitter/responder) attached to items to be counted or identified. Tags may or may not work. Active tags are those that are partially or fully battery-powered, have the ability to connect to other tags, and can initiate their dialogue with the tag reader. Passive tags, on the other hand, do not require an internal power source but are enabled by the tag reader. The tags consist mainly of an antenna connected to a microchip, for the sole purpose of storing data.

Reader: also known as a transceiver (transmitter / receiver) consisting of a unit of radio frequency interface (RFI) and a control unit. Its main functions are to activate the tags, to set the contact sequence with the tag, and to transfer data between the app's software and tags.

Title of the work: RFID-enabled Temperature Sensing Devices
Authors: Joe Dowling (1), Manos M. Tentzeris (1, 2), and Nick Beckett (3)
Publication details: Georgia Institute of Technology (IEEE-2017)
Page no: 02-08
Description: RFID devices do nothing, so they fall into the RFID category without tools. The transmission power comes from the reader and therefore needs to be large enough to enlarge the tag, and indicate the signal back to the reader. The need to strengthen the marker is limited in terms of the scope of traditional RFID artificial intelligence. For wireless RFID systems, the distance is increased because the action tag does not require power, so the range is limited by the back distribution response only. This allows wireless RFID systems to reach a higher level compared to traditional RFID systems. Temperature sensors are integrated with traditional RFID tags.

Title of the work: RFID-Controlled smart range and method of cooking and heating
Authors: Brian L. Clothier, Grand Forks AFB, ND (US)Thermal Solutions, Inc., Wichita, KS
Publication Details: Cooking and Laundary Technology(2005)
Page No: 11-34
Description: The system and how to provide multiple recipes and the ability to automatically heat cooking utensils too other features that use RFID technology, as well as learning ability and write instructions for heating and assisting jointly their murder. The import temperature range is given two antennas per hob, and includes a user interface and how to install. The vessel inserts the RFID tag once Heat sensor.

In the first cooking mode, the recipe is read width and distance assists the user in using recipe for automatic ship heating in Specified temperatures and by telling the user to add ingredients. The recipe is labeled RFID so that if the ship is moved to another hob, where the recipe had never been read, the new hob can read the recipe from the RFID tag again continue its operation.

Internet of Things By Arshdeep Bahga[1]
Amazon Elastic Compute Cloud Documentation[2]
RFID-Controlled smart range and method of cooking and heating[3]
Android Cookbook: Problems and Solutions for Android Developers[4]
Github documentation[5]
Learning at home with the Raspberry Pi Foundation – Raspberry Pi[6]
RFID-enabled Temperature Sensing Devices[7]
MFRC522 RFID Reader[8]
Python from the Very Beginning: with more than 100 exercises and answers[9]
RFID Technology and Its Applications in Internet of Things (IOT)[10]

1.3 Project Problem Statement and Objectives

The controller enables RFID reader through the hardware interface pins. Once the RFID reader gets activated, it starts to emit RF signals from the antenna. The RFID tag is placed nearly to the RFID reader within the range of 5-10 meters. The RFID tag is a passive device that contains no battery. Some of the plants may use active or semi-active tags. In the case of passive tags, when the RF signals reach the RFID tag, the tag gets activated and starts sending back the data like UID and real-time temperature data to the RFID reader. RFID reader gathers data from the tag that will be redirected to the controller. While gathering data, the data in the form of RF signals that are converted to digital forms, and displays on the user interfaces.

1.3.1 Objectives

a. Making the system IoT based automated, by connecting the existing to a remote environment.
b. Sending the commands to a remote device for manage and control of the temperature reading devices.
c. Displays the physical location of remote devices located at.
d. Predicts the amount of time required to reach the temperature threshold value.

1.4 Organization of the Report

The first phase is the introduction of the project which tells about the project title and abstract of the project. the introduction of the project which is the tools are required and their descriptions. this phase we will research some papers.
In the second phase, we have to look into the component information and how it works, protocol setup information. The Hardware components are RFID Tag, RFID Reader, Raspberry pi Board, and Temperature sensor. We have to check the component level working conditions and circuit level designs.
In the third phase, we are writing the architectural diagram and how it works in a remote environment, and how to connect the IoT device to a server. We are writing the Unified Modelling Language Diagrams like use case diagram, data flow diagram, class diagram, and the database.
The fourth phase, the hardware setup and working functionalities are explained. We are implementing the code for the Android application, IoT server, and database. We have also observed the code integration information. Finally, we have observed the result of the project.

Chapter 2

High-level Design

The second chapter is a high-level design which includes how to upgrade software (which describes which model you use from software program engineering to do your job), and functional and non-functional requirements.

2.1 Software development methodology

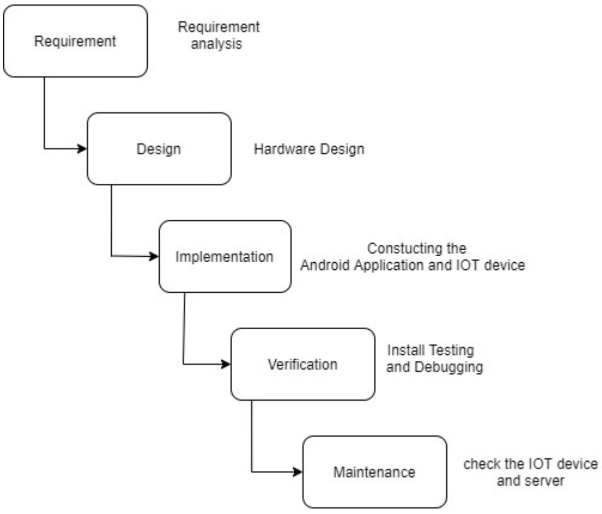

Requirement
Requirement analysis
Design
Hardware Design
Implementation
Constucting the Android Application and IOT device
Verification
Install Testing and Debugging
Maintenance
check the IOT device and server

Figure 2.1: Waterfall model

We are using Waterfall Model to understand the needs and usability of the system. This model shows the use of software in a consistent manner. In this model, any phase begins when the previous phase is completed.

The Waterfall model is the first step in the SDLC model to ensure the success of our project. The complete software development process is divided into 5 categories, namely Requirement Analysis, Design, Implementation, Verification, Maintenance.

Activities on Requirements

A key part of this project needs specification. While analyzing the need for hardware by reading real-time data and location readings, both on a single sensor device With remote sensor device management, a controller was needed to manage one or more sensors for easy configuration and update of audio data in the cloud. So all of this the interface should give you a complete GUI connection between the user and the remote device.

Activities on Design

After the required documentation, the program is built on the hardware component, Later with software. This section fully describes the visual acuity, of the Hardware build process focused on contact with the sensor and control device. The process of connecting the controller to a storage server is the value of the sensor in the cloud database. The workflow of the software component with the required user actions should be followed.

Activities under Implementation

Implementation phase follows the detailed design document, by identifying flow of software segment and conditions that have to be followed. code the design phase by using tools to implement activities on user needs. some of that are.
1) Connect the controller with server and store data on cloud database.
2) Login interface for user to fetch data from server.

Activities on Verification

In the verification phase, test the actual working of implementation of our work. The verification process was tested for the android application activity such as Login, Register, and Add new sensor. We perform verification for data types and formats

stores in the server database.
Verification Activities
a. Network connection between IoT device and server.
b. Login, Register, and Add sensor activities on the android application.
c. Format for temperature and location data.
d. All possible status of the device.
e. Notification performance.

Activities in Maintenance

The maintenance section belongs to how the project will help you in the industries and how it is working in the real-time system. the user can how to use the project like IoT design and user have any problem related to the project and updates the project.
Maintenance Activities
a. Log files maintenance and rotation in both IoT device and server.
b. AWS cloud services server down reporting.
c. Database disk usage alerts.
d. Event Manager maintenance.

2.2 Architecture

The above architecture diagram shows the Temperature sensor circuited with the RFID Tag that is placed nearly by to the RFID reader which connected to raspberry pi board. The Raspberry pi board enables the GPIO(general-purpose input/output) pin which is connected to RFID reader after it emits the radio frequency signals. when the data received that will uploaded to IOT server and server will stores the data into mysql database, user can view the data with the help of web server and they can command for turning on/off the device.

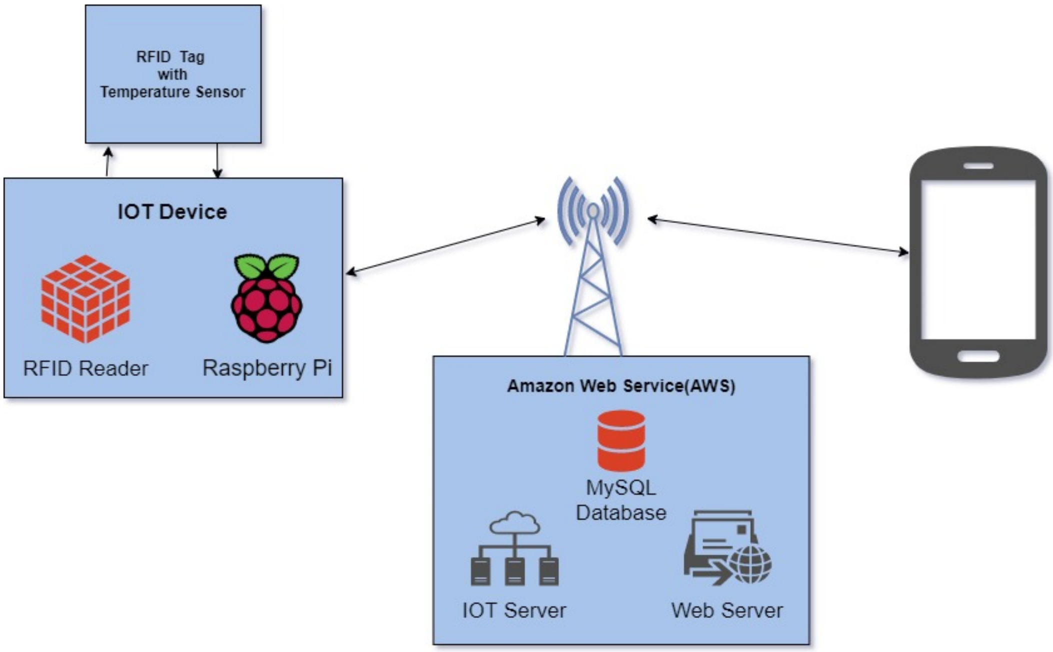


Figure 2.2: Architecture

2.3 Functional Requirements

2.3.1 Reading Activity

Name of the module:Temperature reading
Parameters:none
Purpose: The module shall read the temperature from the RFID tag which is connected to the Raspberry pi through the RFID Reader by enabling respective GPIO pins.

2.3.2 Uploading Activity

Name of the module:Uploads Data to Server
Parameters:temperature data
Purpose:The Raspberry pi shall upload the gathered data from RFID tag to IoT server for ensuring remote access for user whenever and wherever they wants.

2.3.3 Registration Activity

Name of the module : Registering user
Parameters : full name, username, password, confirm password
Purpose : User shall register himself by entering his name, username, and password, to get permission for manage the IoT device.

2.3.4 Login Activity

Name of the module : Login
Parameters : username, password
Purpose : User shall login remote system by entering username and password to keep track of IoT device.

2.3.5 Dashboard Activity

Name of the module : Dashboard list
Parameter : sensor id, sensor name
Purpose : User shall have enter Id and reference name to add or edit the sensor information to dashboard. he can remove the sensor from the dashboard if he wants to delete.

2.3.6 Sensor Activity

Name of the module : Sensor Information
Parameter : temperature id, temperature name, temperature value, device status, minimum limit, maximum limit, longitude, latitude, command.
Purpose :

a. User shall start the device remotely by sending start command.

b. User shall view the status and temperature value once the device get started and he can set minimum and maximum limit for reading temperature.

c. User shall view the location of device where it is located by the value of longitude and latitude.

d. User shall configure the system for getting notification and also set auto command for stopping device when it reaches maximum and minimum range.

e. User shall view the analysis report in graphical visualization.

f. User shall stop the device by send stop command.

2.3.7 Profile Activity

Name of the module : Profile Information
Parameter : user name, full name
Purpose : User shall view his name and user name.

2.3.8 Change Password Activity

Name of the module : changing password
Parameter : new password, confirmation password
Purpose : This helps user to change his password.

2.3.9 Logout Activity

Name of the module : Logout user
Parameter : none
Purpose : Logging out from the Android application.

2.4 Non-Functional Requirements

In the SDLC (Software Development Life Cycle) the non-functional requirements to monitor the application work well. These requirements tell us about our application and whether they are suitable for the application.

2.4.1 Scalability

This system is capable of connecting more than one temperature sensor into IoT device. It can also adopts various sensors like flame, rain gauge, motion detector and more, with a bit of changes in android applications. It provides control to those newly added sensors through remotely.

2.4.2 Accessibility

In this feature user can login and control the IoT device. User can access real time temperature value being uploaded and also can view the location of sensor where it is located. User can access and modify the status of sensor.

2.4.3 Usability

This system can be used in industries where the temperature is constantly increasing or decreasing, and also to read the weather information of environment only on temperature with temperature sensor.

2.4.4 Availability

his feature is about server and System. There should be an IoT user working on it. if the server does not respond the task will stop.

Chapter 3

Detailed Design

3.1 Interface Design

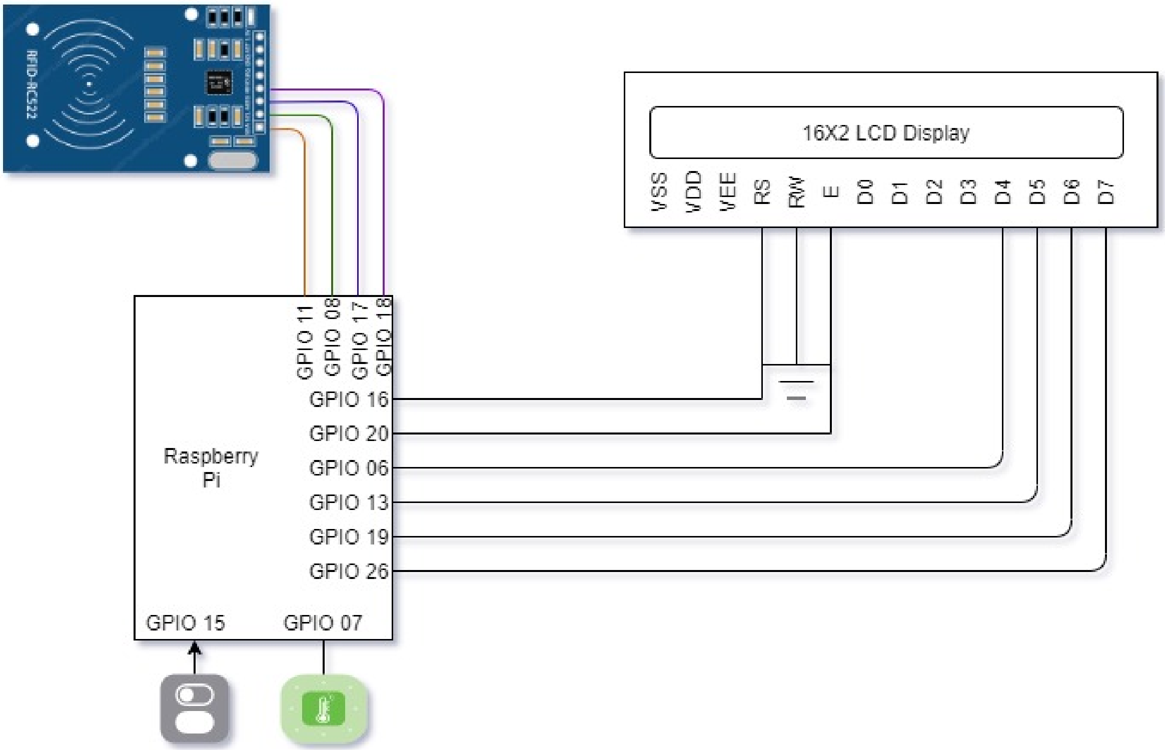


Figure 3.1: Hardware Interface

Raspberry Pi Pin	**Conneted To**	**Description**
GPIO 06	LCD D4	LCD Data Line for first pin
GPIO 13	LCD D5	LCD Data Line for second pin
GPIO 19	LCD D6	LCD Data Line for third pin
GPIO 26	LCD D7	LCD Data Line for four pin
GPIO 16	LCD EN	LCD Enable signal pin
GPIO 20	LCD RS	LCD Register Select Line pin
GPIO 11	RFID Tx	RFID Transmitter pin
GPIO 8	RFID Rx	RFID Receiver pin
GPIO 17	RFID MISO	RFID Master In Slave Out pin
GPIO 18	RFID MOSI	RFID Master Out Slave In pin
GPIO 15	Push Button	For Manually control device
GPIO 07	Temperature Sensor	Temperature Sensor

Table 3.1: Circuit Pin Description

The figure 3.1 shows the design an IoT device use a raspberry pi 3.0 controller board to communicate with the IoT server. The raspberry pi contains 40 GPIO(General Purpose Input Output) pins, which including power control, input, and output purpose pins. The 16x2 LCD display was used to display the current status of the IoT device. The RFID reader is connected to the raspberry pi board, which reads temperature from the RFID tag and sends it to raspberry pi to process. The raspberry pi board also has a temperature sensor that reads temperature from the environment and stores it in an RFID tag. Finally which includes a push button to manually control the IoT device when we press that button the device performed the turn on and off function of the IoT device.

3.2 Data Structures and Algorithms

3.2.1 IoT Server Algorithm

Purpose: Connect, Communicate and Update the receiving data into the data base. Sending notification alerts to users and signals to IoT devices.

Use Cases: The server starts automatically when the system is power up.

```
Algorithm:
//Implement an IoT server to control IoT devices.
//Input: IoT device with unique device id.
//Output: Sending and receiving data from IoT devices.
Repeat
    Waiting for IoT devices and do if IoT device connection found
    Creating a thread for each device and perform the following on each thread
        Repeat
            Receiving a Device ID and store.
            Receiving data from the device.
            Update the data into the database using the Device ID.
                If the temperature limit is out of range then
                    Send notification to user.
                If the user sends a stop signal then
                   Send stop signal to IoT device.
        Until IoT device Turn Off
Until the user interrupt the server to stop
```

Error handling:If encounter any error in between sever and device connection terminate the existing connection and go to waiting for state for the new connection. The device again sends a new request for the connection.

3.2.2 IoT Device Algorithm

Purpose: Sending data to IoT server and control the IoT device by receiving the command from IoT server.

Use Case: When the power-up device the program starts device function and display the status on LCD.

Algorithm:
//Implement an IoT device to sending data to the server.
//Input: IoT device with unique device id.
//Output: Sending and receiving data from IoT server and control the device.
Step 1: Connect to IoT server if the connection is not available raise the exception and retry.
Step 2: Sending Device ID to IoT server.
Step 3: Check connection status if it is not found raise the exception and go to step 1.
Step 4: If the server sets the STOP signal then turn off the device and go to ideal status.
Step 5: Else sending data to the server.
Step 6: If the device is in ideal status then if receive the START command then turn on the device and set status active.
Step 7: Else continue as ideal status.

Error Handling: If encounter any error in between sever and device connection terminate the existing connection and create a new connection.

3.3 UML diagrams with discussions

3.3.1 Use Case Diagram

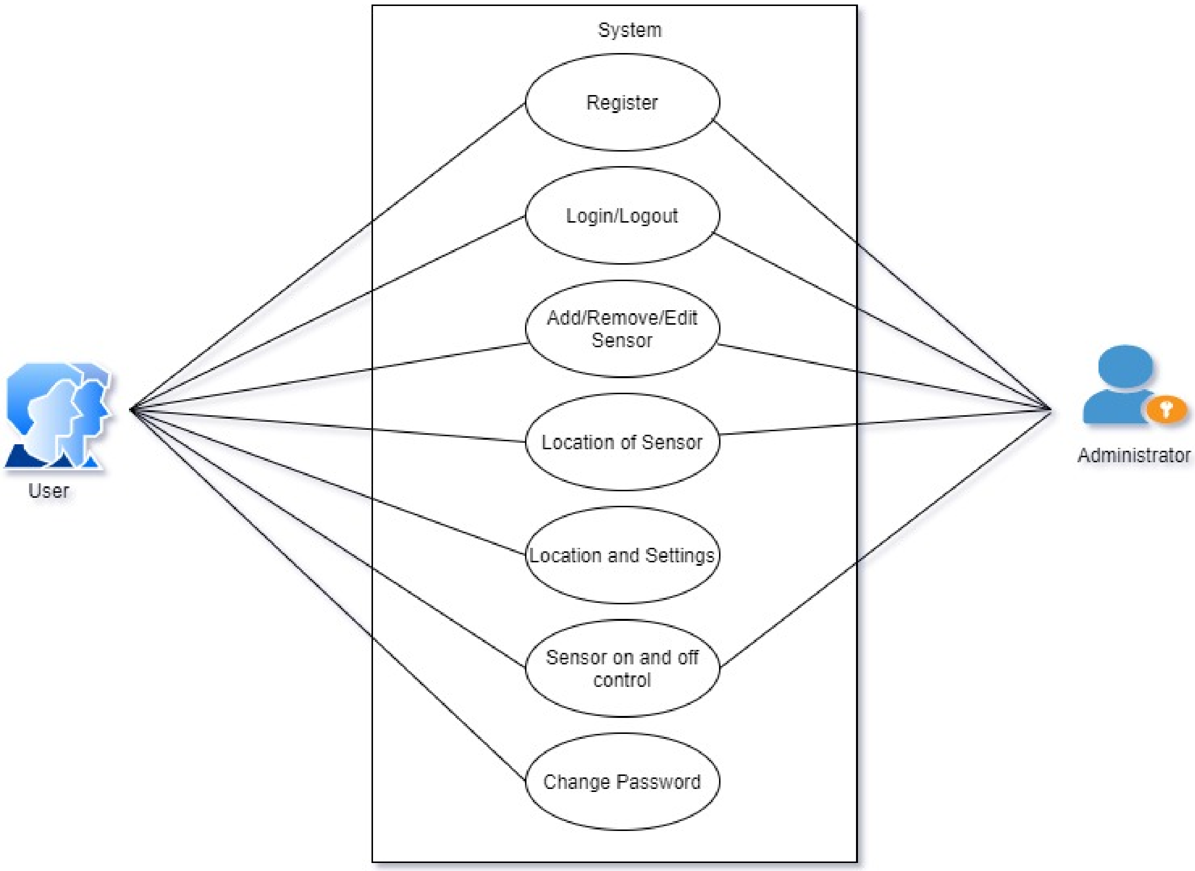


Figure 3.2: Use Case Diagram

The Figure 3.2 shows the user interacts with the system with respect to a server controlling admin and user who are access the data from the sensor. The administrator adds the sensor information and configures the IoT device with the sensor which is being connected and the newly added device to IoT system. The server enables the user to register and access the data. once the sensor is added successfully the user can view the current status, temperature value, and location of the sensor. Users can control the sensor by giving the command to IoT server. Users can also change the password if needed.

3.3.2 Data Flow Diagram

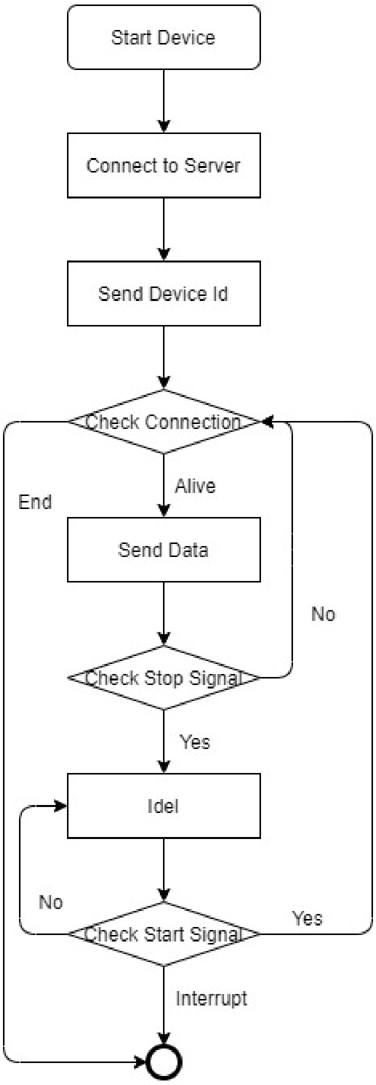


Figure 3.3: IoT Device Data Flow Diagram

The Figure 3.3 is process of IoT device, mainly We have to start the device to connect to the server. Once the server gets connected, the device is being configured with the device id with the server and check for commands from the user to gets started with the respective sensor. If the sensor connection or command has resulted in approval, the sensor will send the data to the server and waits for the stop command.

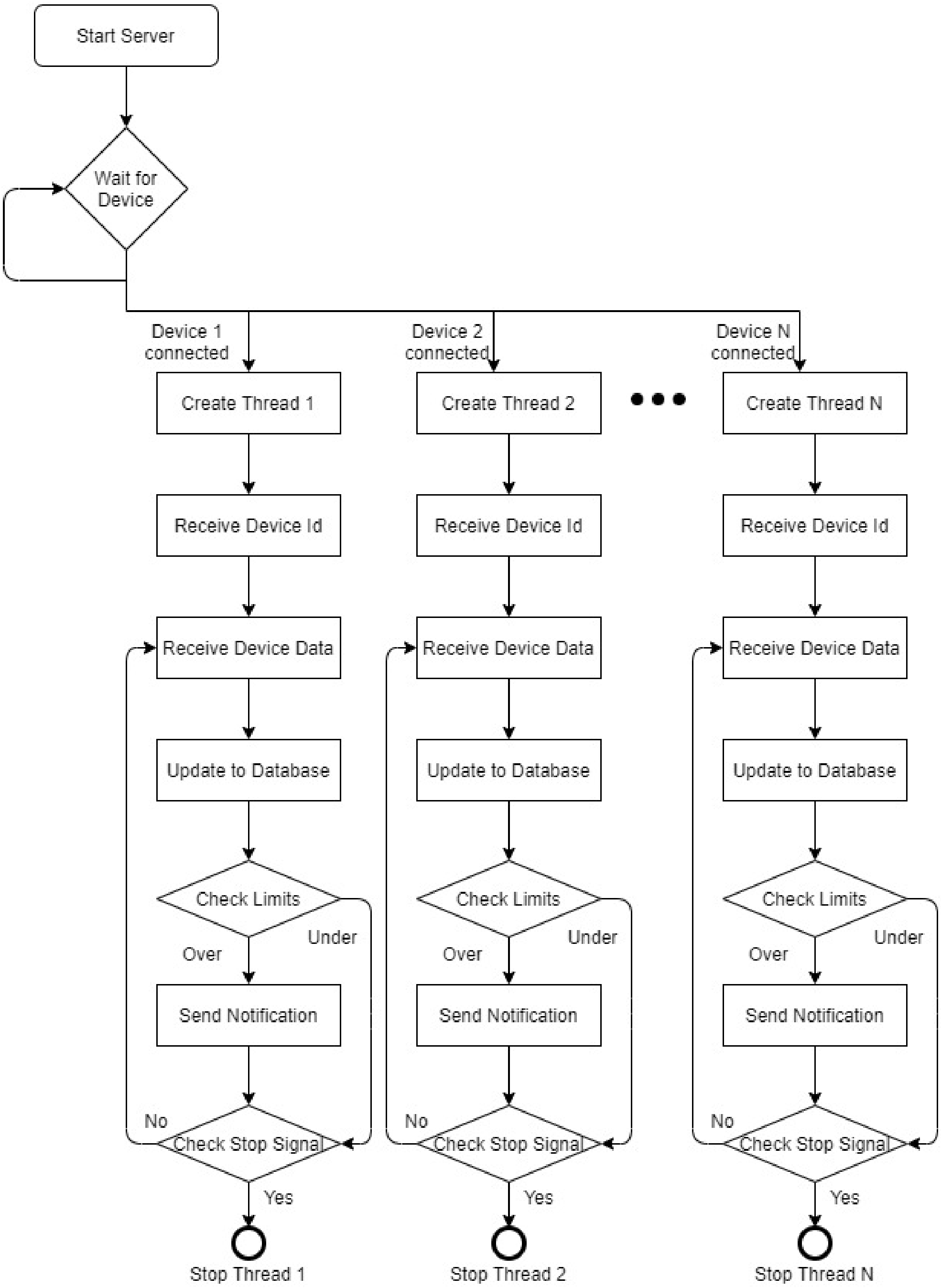


Figure 3.4: IoT Server Data Flow Diagram

The Figure 3.4 shows the working of IoT server. Server is in start condition at all the time and waits for the sensor id from the IoT device. If the sensor id is stored in the database, the server will create a thread to keep tracking the sensor information. The server receives the sensor id from the user, to send that information to the user. The server will keep on receiving information from the sensor to update its current status. When the updated temperature value has reached the limit specified in the android app, the server sends a notification to the user about the temperature value is reached the limit. Then server keeps sending the temperature value to the user till receives the stop command from the user. If the server receives the stop command, the thread will stop its operation and sends a stop command to IoT device to turn off the device.

3.3.3 Class Diagram

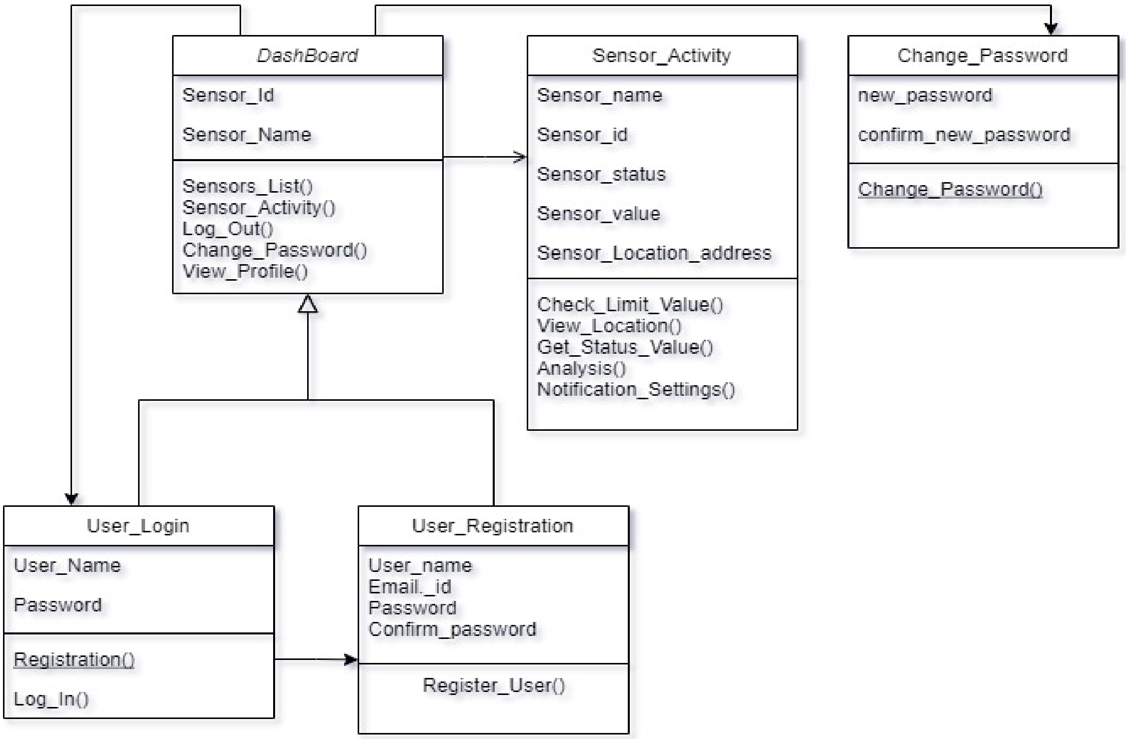
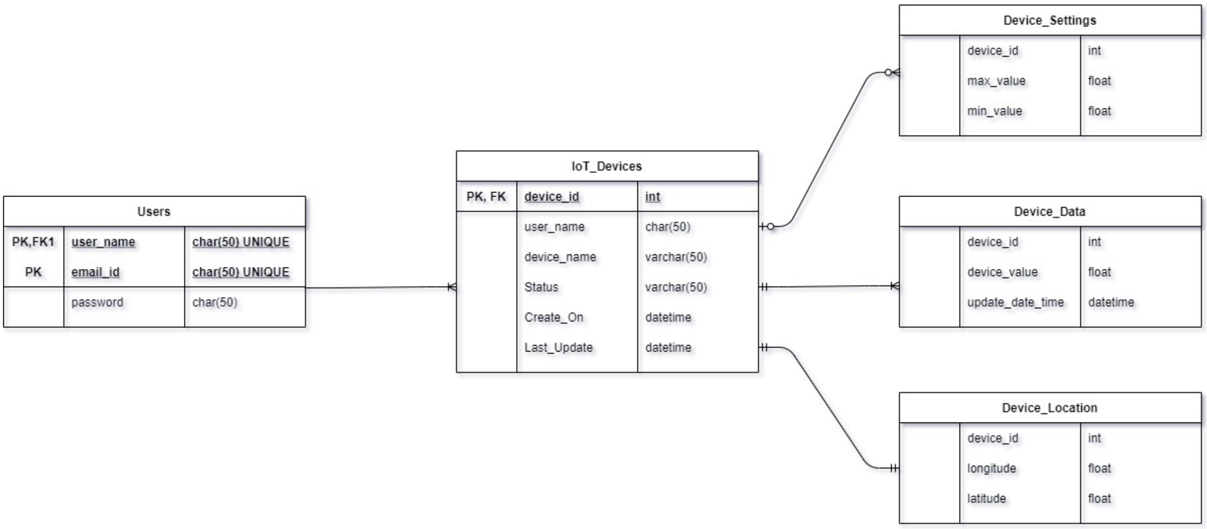


Figure 3.5: class diagram

The Figure 3.5 shows how the user will interact with the system. Mainly users have to log in to the system to get information about the sensor's current status and temperature value. If the user is not registered, must have to register to the system. once user register or login to the system, the user will be redirected to the dashboard which contains n number of sensor information's, which are added. After clicking on the sensor tab he gets the current status of the sensor value being read from the sensor in the IoT environment in the class of sensor activity.In sensor activity, the user can set a threshold value for temperature and also can handle notifications by auto-turning off the device. In the dashboard activity, the user can also change the password to a new password if needed.

3.4 Data Source/Database used and Formats

3.4.1 Database Design



Users

PK,FK1	user_name	char(50) UNIQUE
PK	email_id	char(50) UNIQUE
	password	char(50)

IoT_Devices

PK, FK	device_id	int
	user_name	char(50)
	device_name	varchar(50)
	Status	varchar(50)
	Create_On	datetime
	Last_Update	datetime

Device_Settings

	device_id	int
	max_value	float
	min_value	float

Device_Data

	device_id	int
	device_value	float
	update_date_time	datetime

Device_Location

	device_id	int
	longitude	float
	latitude	float

Figure 3.6: Database Design

The database design is shown in Figure 3.6, for design the database we used we used AWS EC2 Linux/windows instance. It consists of 5 classes that are Users, IoT Devices, Device Settings, Device Data, Device Location. Users tables have attributed user name for the full name of the user, email id for user name, and password. IoT Devices consists of user name for the device is belonging to and other device information like name, id, status, created on and the Last Update for accessing current temperature value. Device settings is a weak entity that contains device id and max, min value to notify the user, the temperature reaches its threshold value. Device Location is about longitude and latitude value to show the exact location of the device is being located.

Chapter 4

Implementation

In this chapter, we discuss some of the most needed tools to do our project. This chapter covers the coding standards, implementation of code, and tools and technologies used.

4.1 Tools and Technologies

4.1.1 Hardware Requirements

a. RFID Tag(ISO/IEC 1443 A)
b. RFID Reader(MFRC522)
c. Raspberry Pi Board(3 model B)
d. Temperature sensor(DS18B20 Waterproof)
e. Android Smartphone(Android 10, 4GB RAM, 64GB Disk)

a. RFID Tag(ISO/IEC 1443 A)
We use RFID Tag with High Frequency 13.56 MHz which provides a low profile and is designed for use on assets, servers, pallets, or solid attachments, and is able to adapt to the survival of high-temperature RFID applications. this RFID tag is a transaction tag.

b. RFID Reader(MFRC522)
We are using the RFID reader to read the data from the RFID tag. the RFID reader

to send the data to Raspberry pi board.

c. Raspberry Pi Board
We are using the IoT Raspberry Pi 3 Model B board for the RFID reader will send the data to this board the raspberry pi will send the IoT server.

d. Temperature Sensor
A temperature sensor is an electronic device that measures the temperature of its environment and converts input data into electronic data to record or monitor temperature changes. this temperature sensor is circuited with an RFID tag the RFID reader will take data from the Tag.

e. Android Phone
Android smartphone that monitors and controls the device status and temperature of the IoT device. It uses the data stored in IoT server and also receiving notifications when IoT device is a start or temperature cross its limit.

4.1.2 Software Requirements

a. Java Programming
b. Android Studio
c. AWS Cloud
d. Python Programming
e. PHP

a. Java Programming (JDK 1.8)
Java is a high-level programming language. That support various inbuilt libraries for developing application. To develop an android phone application we are using java programing in our project.

b. Android Studio (Version 3.5)
We are using Android Studio for Building the Android Application in our project. Android Studio provides the fastest tools for building apps on every type of Android

device.

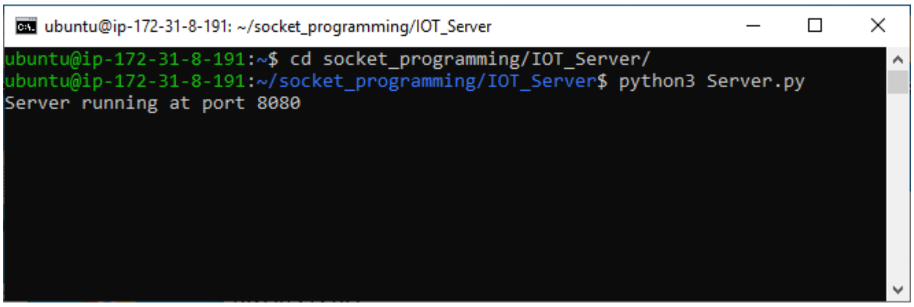
c. AWS Cloud (EC2 Linux Insatance)
We use AWS Cloud to store data in the cloud for our project. AWS Cloud provides cloud computing platforms and APIs, these cloud-based web services offer a variety of technical infrastructure and deploy building blocks and tools.

d. Python Programming (Version 3.9)
Python Programming is a high-level interpreted language. We are using Python programming in our project to develop IoT server and IoT devices, which uses socket programming for network communication.

e. PHP (Version 7.2)
PHP is a server programming language using for the back-end purpose. PHP sends the data from the server database to the android application.

4.2 Experimental Setup

first, we need to verify experimental set up the RFID Tag and Reader whether it is working or not. then we have to do hardware setup using IoT raspberry pi board the reader and tag is working then we have to move the complete hardware setup.We have to move to the Android application development we are using the java programming language to developing the Android application.



```
ubuntu@ip-172-31-8-191:~$ cd socket_programming/IOT_Server/
ubuntu@ip-172-31-8-191:~/socket_programming/IOT_Server$ python3 Server.py
Server running at port 8080
```

Figure 4.1: The IoT Server Start

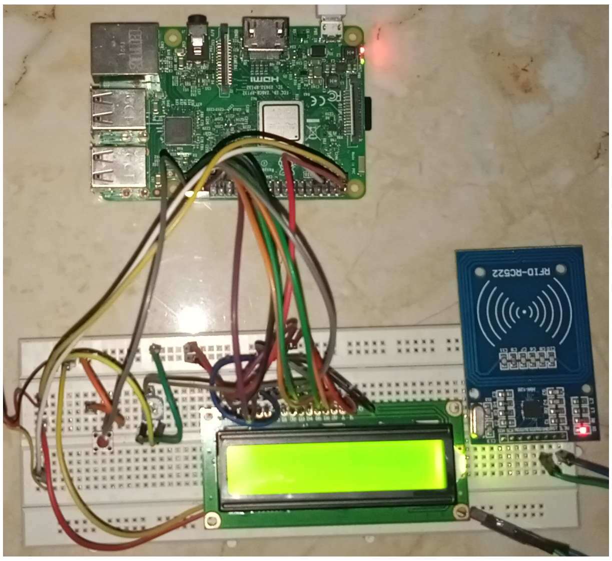


Figure 4.2: The IoT device start

4.3 Coding Standards followed

The coding standards of Java programming in a typical project. The following requirements are:
- Annotation for a field.
- Block and Switch indentation(4 spaces).
- Wildcard imports not permit.
- Class names: in upper camel case.
- Method names: in lower camel case.
- Constant names: use CONSTANT CASE.
- Column limit: 120 characters.
- File encoding: UTF-8.
- Braces: always used, even when the body is empty or contains a single statement.

The coding standards of Python programming in a typical project. The following requirements are:
- Indentation(4 spaces).
- Maximum line length of 79 character.
- For classes two blank lines for the method single blank line.
- UTF-8 source file and coding.
- Descriptive naming styles for variables and methods.

4.4 Code Integration details

4.4.1 IoT Server

```
#IoT Server for receiving data from device and update that data into database
#Sending signal to stop / strat device.
#Sending Notification to user when device is started

from Sensor import Sensor
import socket
from socket import socket, SOCK_STREAM, AF_INET, SO_REUSEADDR,
SOL_SOCKET, SO_KEEPALIVE
from threading import Thread
from time import sleep
import json


#communication function for each device to control & monitorint device.
#input: device socket and device socket address
def communication(client_sock, client_addr):
    END_MONITOR = False
    print(client_addr, "connected")
    device_id = client_sock.recv(5).decode()
    print("device "+device_id+" started")
    reconnect = client_sock.recv(1).decode()
    print(reconnect)
    sensor = Sensor(device_id)
    if reconnect == "N": sensor.start()
    else: sensor.reconnect()

    #monitor the device and send start/stop signal to device
    #send notification to user
    def stop_monitor():
        print("---START---")
        while not END_MONITOR:
            try:
                status = sensor.status()
                if status == 4:
                    client_sock.send("START".encode())
                    sensor.reconnect()
                if status == 2:
                    client_sock.send("STOP".encode())
                    sensor.stop(1)
                if sensor.check_limit():
                    if sensor.auto_mode():
```

```
                    client_sock.send("STOP".encode())
                    sensor.stop(1)
                sensor.send_limit_notification()
            sleep(1)
        except:
            break
    print("---End---")

t = Thread(target=stop_monitor)
t.start()
while True:
    try:
        data = client_sock.recv(1024)
        print("Data Receviced  from "+device_id+" : "+data.decode())
        if data.decode().startswith("OFF"):
            sensor.stop(1)
        elif data:
            sensor_data = json.loads(data.decode())
            sensor.set_data(sensor_data.get(’temp’), sensor_data.get(’lon’)
            , sensor_data.get(’lat’))
        else:
            break
    except SyntaxError as ex:
        pass
    except TypeError as ex:
        pass
    except Exception as ex:
        print("communication : ", ex)
        break
client_sock.close()
sensor.stop(0)
END_MONITOR = True
print("device "+device_id+" stop")
```

```
HOST = "" #Same host address
PORT = 8080 #Run server at port 8080
server_sock = socket(AF_INET, SOCK_STREAM)
server_sock.setsockopt(SOL_SOCKET, SO_REUSEADDR, 1)
server_sock.bind((HOST, PORT))
server_sock.listen(5)
print("Server running at port %d"%PORT)
while True:
    client_sock, client_addr = server_sock.accept()
    client_sock.settimeout(5)
    client_sock.setsockopt(SOL_SOCKET, SO_KEEPALIVE, 1)
    client_thread = Thread(target=communication, args=(client_sock
    , client_addr))
    client_thread.start()
```

4.4.2 IoT Device

```
#IoT device send data to server and receiving data from server to
control the device

from w1thermsensor import W1ThermSensor
from time import sleep
import RPi.GPIO as GPIO
import socket
import threading
import requests
import lcd


#Function for push button to update status and turn on/off of the device
def push_button():
    global DEVICE_STATUS
    while True:
        if GPIO.input(BUTTON_PIN) == GPIO.HIGH:
            DEVICE_STATUS ^= True
        sleep(0.1)


#Add content to LCD display (Device Id, Status, and Temperature)
def lcd_display(temp):
    lcd.command(lcd.LCD_LINE_1, lcd.LCD_CMD)
    lcd.message("Device #:"+str(DEVICE_ID))
    lcd.command(lcd.LCD_LINE_2, lcd.LCD_CMD)
    if DEVICE_STATUS: lcd.message("On   Temp:"+str(temp))
    else: lcd.message("Off")


#Send data to device (Status, Temperature, Location) every one sec.
def send_data(server_sock, reconnect):
    try:
        sensor = W1ThermSensor()
        lon, lat = requests.get("http://ipinfo.io/loc").text.strip().split(",")
        server_sock.send(str(DEVICE_ID).encode())
        print(lon, lat)
        if reconnect: server_sock.send("YES".encode())
        else: server_sock.send("NO".encode())
        while True:
            temp = sensor.get_temperature()
            temp = "%.2f"%temp;
            print("Temperature:", temp, "Status:", DEVICE_STATUS)
            lcd_display(temp)
            try:
                if DEVICE_STATUS:
                 data = ’{"lon":’+lon+’, "lat":’+lat+’, "temp":’+temp+’}’
                    server_sock.send(data.encode())
                else:
                    server_sock.send("OFF".encode())
            except:
                print("Reconnected")
                connect(True)
                break
            sleep(1)
    except:
        pass


#Receiving commands from server to control the device (START/STOP)
def recv_data(server_sock):
    global DEVICE_STATUS
    try:
        data = True
        while data:
            data = server_sock.recv(1024)
            print("\n"+data.decode())
            if data.decode() == "STOP": DEVICE_STATUS = False
            elif data.decode() == "START": DEVICE_STATUS = True
    except:
        print("Error in receive thread")


#Connect to server using socket and create thread for send and receive
functions
def connect(reconnect=False):
    server_sock = socket.socket(family=socket.AF_INET, type=socket.SOCK_STREAM)
    server_sock.connect((HOST, PORT))

    send_thread = threading.Thread(target=send_data, args=(server_sock
    , reconnect))
    recv_thread = threading.Thread(target=recv_data, args=(server_sock,))

    send_thread.start()
    recv_thread.start()
```

```
BUTTON_PIN = 15 #Push Button Pin GPIO15
DEVICE_ID = 1 #Device Id for the device
DEVICE_STATUS = False #Intial Status of device off

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(BUTTON_PIN, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)

t1 = threading.Thread(target=push_button, args=tuple())
t1.start()

lcd.init()

HOST = "13.59.218.57" # The server's hostname or IP address
PORT = 8080           # The port used by the server

connect() #Calling connect function
```

4.5 Execution Results and Discussions

At end implementation of project the controller able to reading data from RFID tag and send that data into cloud system and android phone user fetch the information and sending instruction to controller. The data which is transmitted from controller store it in the cloud and that's help to build machine learning model. Android phone user getting voice or text notification when the system reach's its maximum temperature level. If android phone user wish to monitor the information about device display information about temperature and location in their screen. The user wish to set maximum limit of temperature sending instruction to controller through cloud. The user can able to manage multiple devices in same phone. Estimating system behavior make use of existing data and ML model show the future prediction on the screen.

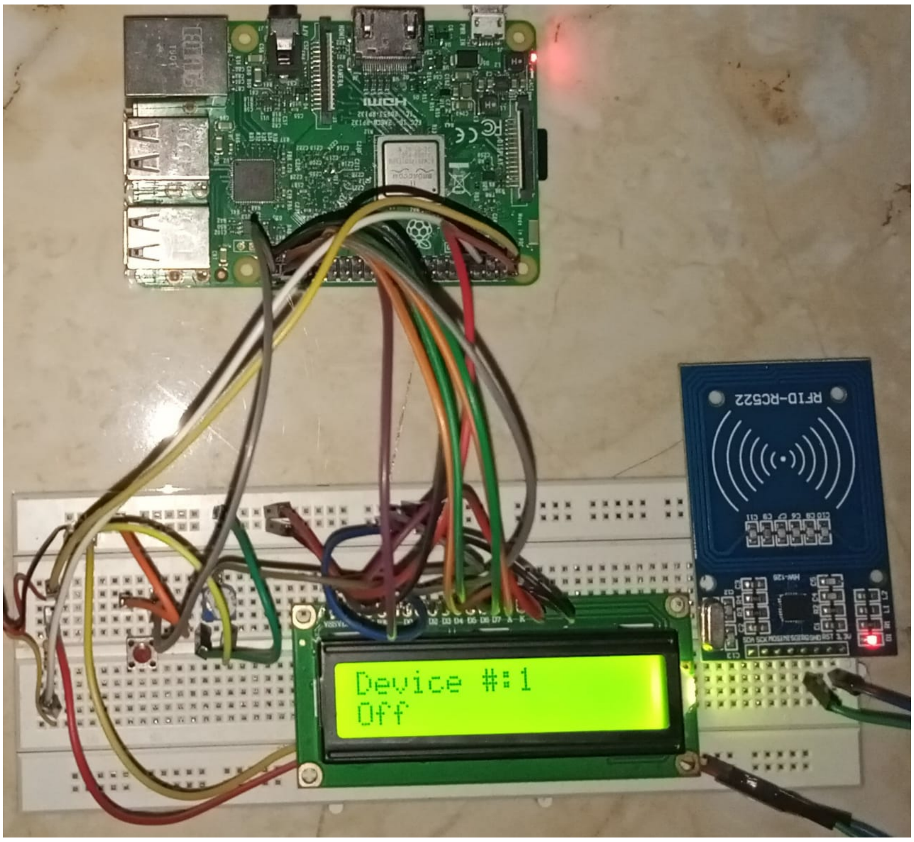


Figure 4.3: The IoT device is off

The above picture shows an IoT device connected with Raspberry Pi, RFID Reader, and LCD display when it's turned on for a first time. Its shows a unique Device Id, and the status of the device initially is off.

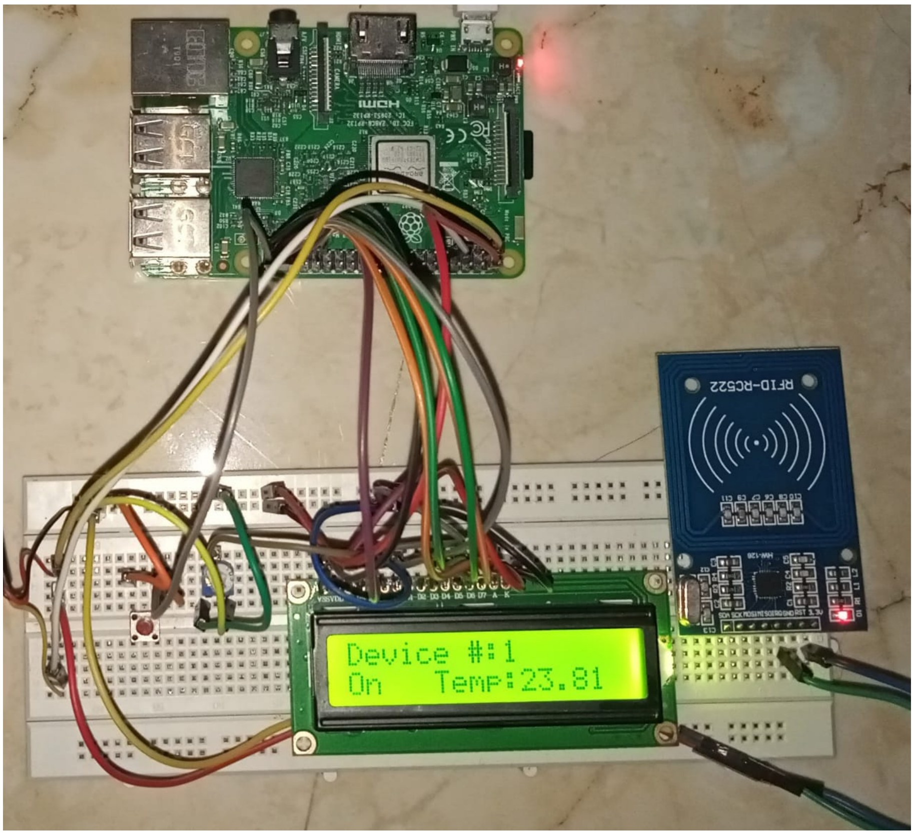
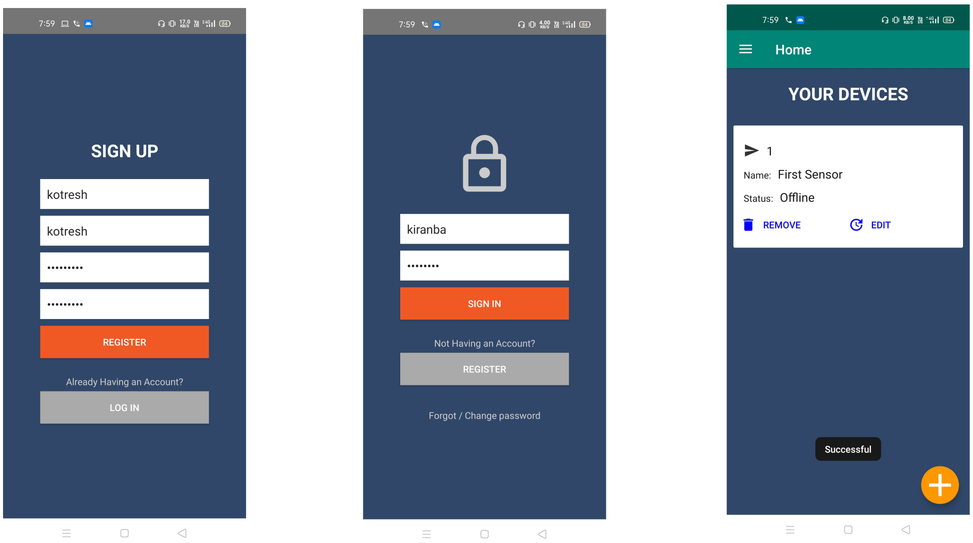


Figure 4.4: The IoT device is on

The above picture show when the IoT device is in on condition, which display the temperature on the LCD display.



kotresh

kotresh

kiranba

1

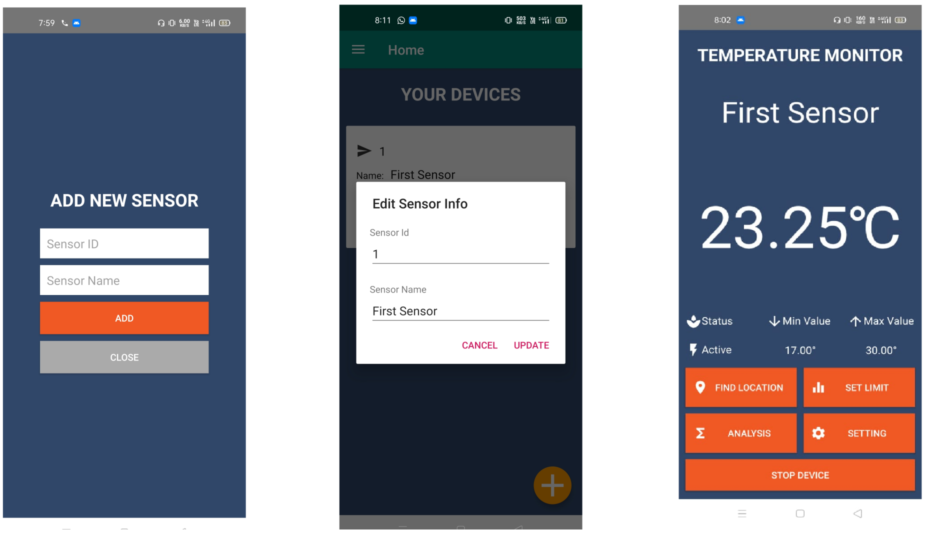
Name: First Sensor

Status: Offline

Successful

Figure 4.5: Result Register, Login, Dashboard

The above pictures show the android application with activities for sign up for the new user, log in for the registered user, and the main activity gets after successful login for the user. In the signup form request full name, username, password, and confirmation for the password then click the Register button to create a new user. In the login form request username and password then click the login button for the authentication. The main screen gets after successful login its shows all available devices.



1
Name: First Sensor

Edit Sensor Info

Sensor Id
1

Sensor Name
First Sensor

First Sensor

23.25℃

Status	Min Value	Max Value
Active	17.00°	30.00°

Figure 4.6: Result Add Sensor, Edit Sensor, Start Sensor

The above pictures show the android application with activities to add a new sensor, edit sensor details, and dashboard for the sensor. In adding new sensor activity for request sensor id (device id) and sensor name to add the sensor to application. The edit sensor activity contains a dialog box with sensor id and sensor name to update details. In sensor dashboard activity show device status, temperature, and other options.

Chapter 5

Testing

In this section dealing with shooting, investigate art broadcasting activities and experimental data. As we see, testing is a strategy that will increase the enjoyment of a product or application. photo shoot is a test work arrangement completed to give a glimpse of the product system to customers. the supply of firearms is achieved from the earliest stages and is firmly completed in later grades. In all practice behind the precise dash of the entire module where it is required to test that every module should be tried and installed.

5.1 Test workflow

Observing the process of transferring each defect of an object to an object. It also tests the lead module, sub-module and all calls for specific needs in the software system. We are developing test strategies in subjects known as test scenarios and these tests are completed to ensure that the entire graphic flow system responds appropriately to the business. The purpose of the workflow is to try to ensure that the functionality of the software and the flow of the system diagrams work as predicted by all operating systems. Works of art stream looking at is a specific sub-testing space that requires internal and external knowledge in the application of the business process. Looking at the UI, the components for joining and deliberately doing the product improvements. Presentation, fidelity, visual similarity and more ended when the product was created.

5.1.1 Unit Testing

We apply unit testing for IoT devices, servers, and Android applications. Android development has a built-in unit test framework called Junit 3.0. It is a open-source and automated unit testing tool for android developers. The unit testing for IoT device and server to consider network connectivity to communicate with each other.

5.1.2 Component Testing

a. Software Component Testing
The software component testing for Android applications considers User Interface(UI) components such as a key event, touch, button, menu, dialogs, images, toolbars, and textboxes. In this testing, we perform some validation for each textbox component for Login, Register and Add Sensor Activity in the android application.
b. Hardware Component Testing
In the hardware component testing apply for a component such as Raspberry Pi board, RFID Reader, LCD display, and temperature sensor. We need to confirm all the components are working properly before moving to the implementation part.

5.1.3 Integration Testing

In Integration Testing, on database, server, and device are tested, combined, and verified. In Android integration test checking the integration with android components such as service testing, activity testing, adaptor testing, and content provider testing. Also, test the data integration between the android application and backend application in the server. In the IoT device perform integration testing from the sensor to the database in the server.

5.1.4 Functional and Operational Testing

In Functional and Operational testing we did the following testing:
a. To validate whether all the required mandatory fields are working as required.
b. To validate that the mandatory fields are displayed in the screen in a distinctive way than the non-mandatory fields.
c. To validate whether the application works as per as requirement whenever the

application starts/stops.
d. To validate that the application resumes at the last operation in case of a hard reboot or system crash.

5.1.5 Usability and Scalability Testing

In usability testing for IoT server test, the number of devices connected simultaneously and track the timeline of each device. The usability testing in the android application test device status, temperature data, location, and a maximum number of device able to monitoring. In IoT device test time and memory usage during the communication between server. To test the scalability of the system we consider the number of devices connected and monitor at a same time.

5.1.6 Reliability Testing

From the reliability testing, we measure the failure rate of IoT server, device, and android applications. For that we consider the log files to calculate the reliability, where compare number of error connection and number of valid connections.

5.2 Test case details

5.2.1 Test case id : 01

Unit to test : Reading value from Sensor
Assumptions : Temperature value must be displayed on User Interface
Test data : sensor id, temperature value
Steps to be executed :
1) Check connectivity
2) Dump code to read sensor value
3) Display gathered value on User Interface
Expected result : Real time temperature value
Actual result : Real time temperature value

Pass/Fail : pass
Comments : System can handle more sensor simultaneously

5.2.2 Test case id : 02

Unit to test : Stabled connection with server, sending and accessing data.
Assumptions : Server must be connected with sensor for fetching data, and sending to user application
Test data : Variables and their values
Steps to be executed :
1) Check for connection
2) Check for continuous uploading data from Raspberry pi
Expected result : Data must be uploaded to server
Actual result : Data uploaded to server
Pass/Fail : pass
Comments :

5.2.3 Test case id : 03

unit to test : showing the sensed data on Android application
Assumptions : System must show temperature value on android application
Test data : temperature data
Steps to be executed : fetching data from server and showing that on application
Expected result : Displaying the temperature data
Actual result : Displayed the temperature data
Pass/Fail : pass
Comments :

Chapter 6

Conclusions and Future Scope

In this project, we built an remote environment to manage and control the devices attached with temperature sensors. The temperature data is being generated by RFID and uploaded to the cloud through Raspberry pi to provide user interface in android mobiles. The user can login/register in the android app to get connected with respective RFID device to gather temperature data. They can add n number of sensors and can start/stops the devices from mobile interface. The auto-mode feature works implicitly to control the temperature by commanding the device to turn off when it reaches the defined limits. Android app will displays the location of respective device on the map. Predicts the amount of time required to reach the temperature threshold value. For future scope of work, the multi-nodal sensors, tracking the multi-nodes, and multi-location handling features implementation is to be considered.

Bibliography

[1] Arshdeep Bahga. *Internet of Things.* universities press(india) private limited, 1^{nd} edition, 2015.

[2] Jeff Barr. Amazon elastic compute cloud documentation, 2021. https://docs.aws.amazon.com/ec2/index.html.

[3] ND (US)Thermal Solutions Inc. Wichita KS Brian L. Clothier, Grand Forks AFB. Rfid-controlled smart range and method of cooking and heating. In *Cooking and Laundry Technology ,2005*, pages 11–34, november 2005.

[4] Iran Darwin. *Android Cookbook: Problems and Solutions for Android Developers.* "O'Reilly Media, Inc.", 2012, 2^{nd} edition, 2011.

[5] Github.com. Github documentation, 2016. https://docs.github.com/en.

[6] Gareth Halfacree. Learning at home with the raspberry pi foundation – raspberry pi, 2018. https://www.raspberrypi.org/learn/.

[7] 2) Joe Dowling (1), Manos M. Tentzeris (1 and Nick Beckett (3). Rfid-enabled temperature sensing devices. In *IEEE Industrial Electronics, IECON 2017*, pages 2–8, november 2017.

[8] NXP. Mfrc522 rfid reader, 2016. https://www.nxp.com/docs/en/data-sheet/MFRC522.pdf.

[9] John Whitington. *Python from the Very Beginning: with more than 100 exercises and answers.* Coherent Press, 1^{nd} edition, 2020.

[10] Quanyuan Feng2 Taihua Fan1 Quanshui Lei1 Xiaolin Jia1, 2. Rfid technology and its applications in internet of things (iot). In *IEEE Industrial Electronics, IECON 2012*, pages 2–7, November 2012.

Project Planning AY-2020-21

Activites	No.Of Weeks	Plan/ Actual	Septmeber				October				November			December				January			Feb		March			April				May				June				July		
			1	2	3	4	1	2	3	4	1	2	3	1	2	3	4	1	2	3	1	2	1	2	3	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3
Problem Identification and Literature Survey	6W	Plan	🟦	🟦	🟦	🟦	🟦	🟦																																
		Actual		🟥	🟥	🟥	🟥	🟥																																
Software Requriments and Specifications	3W	Plan							🟦	🟦	🟦																													
		Actual								🟥	🟥																													
Architecture, Design and Prototype	4W	Plan										🟦	🟦	🟦																										
		Actual											🟥	🟥																										
Implementation	10W	Plan																		🟦	🟦	🟦	🟦	🟦	🟦	🟦								🟦						
		Actual																			🟥	🟥	🟥	🟥	🟥	🟥								🟥						
Testing and Validation	3W	Plan																									🟦	🟦	🟦						🟦	🟦	🟦			
		Actual																									🟥	🟥	🟥							🟥	🟥			
Project Closure - **Results Observations** - **Demonstration** -**Report Writing**	3W	Plan																												🟦	🟦	🟦								
		Actual																												🟥	🟥	🟥								

🟦 PLAN

🟥 ACTUAL

Cost Estimation

Project cost estimation is the process of predicting the quantity, cost, and price of the resources required by the scope of a project. Since cost estimation is about the prediction of costs rather than counting the actual cost, a certain degree of uncertainty is involved. The cost estimate is used to determine the size of the required investment to create or modify assets. It is also during the early phases that alternative plans are considered that need to be priced. It covers activities such as resource planning, cost estimating, budgeting and cost control. These activities are repeated in a closed loop and take place during the whole project life cycle.

In this Project most of the work is done on open source tools. We being the team of 4 divided the work equally among ourselves, it took total duration of 8 months to complete this project by working approximately 1 to 2 hours per day.

Name	**Cost**
Android Smartphone	₹ 14,000
Raspberry Pi Board	₹ 4,500
RFID Reader and RFID Tag	₹ 3,300
AWS Cloud Service	₹ 3,200
Electronics Component	₹ 2,000
Final Print and Binding	₹ 3,000
Total	₹ 30,000

PO ATTAINMENT

Programme Outcomes (POs):		Task Preformed	Attainment				
			Excellent 5	Very Good 4	Good 3	Fair 2	Poor 1
PO1	Engineering knowledge	1. Applied the knowledge of Internet of Things, Socket Programming, Java Programming, Machine Learning and Software Engineering.		✓			
PO2	Problem analysis	1. Literature Survey done on "RFID Technology" 2. The objectives of the project were Decided. 3. Knowledge of Internet of Things, Programming, Cloud Computing, Machine Learning and Software Engineering was found to be useful in implementing the project		✓			
PO3	Design/development of solutions	1. Hardware Design Using the Sensors, RFID Tag, RFID Reader, Raspberry Pi Board. 2. Server Connection Including the IOT Server and AWS Server. 3. Android Development.		✓			
PO4	Conduct investigations of complex problems	1. Requirements checking for the Hardware Design. 2. Suitable solutions to meet the requirements are developed. 3. To checking the Server connection through the IOT server to AWS.		✓			
PO5	Modern tool usage	1. Android Studio, AWS, RFID Technology is used.	✓				
PO6	The engineer and society	1. This project helps to Handling the Temperature in real time environment like cold storage, induction stove, buffet tray, Thermal stations, whether monitoring System.		✓			

Programme Outcomes (POs):		Task Preformed	Attainment				
			Excellent 5	Very Good 4	Good 3	Fair 2	Poor 1
PO7	Environment and sustainability	1. This project meets some of the current requirements like Raspberry pi, RFID tag , RFID Reader and Temperature sensor. 2. This project seems the Android application in the application we are viewing the Location, Analysis the graphical data about the temperature and on/off the device.			✓		
PO8	Ethics	1. This project is useful to Temperature Handling in the real time environment like Buffet tray, cold storage, induction stove and whether monitoring system. 2. References are quoted. 3. Report is prepared by students and plagiarism check is made with turn it in software.		✓			
PO9	Individual and team work	1. Each student took up the responsibility of executing one module of the project. 2. The report content was contributed by each of the team members. 3. Integration of the modules was done as a team work. 4. Incorporating the suggested changes was done. 5. As a team presentations and demo of the project was given.		✓			

Programme Outcomes (POs):		Task Preformed	Attainment				
			Excellent 5	Very Good 4	Good 3	Fair 2	Poor 1
PO10	Communication	1. Phase-wise presentation and Demo of progress of project work before the panel. 2. Presentation and Demonstration of project before Industry Experts. 3. Preparation of Report spread across the entire Semester. 4. Regular interaction with Guide and Panel members to incorporate the suggestions given during evaluations 5. Answering queries during presentations and Demos.		✓			
PO11	Project management and finance	1. Project Scheduling and meeting the Guide. 2. Maintaining Project Diary. 3. Time Management Using Available Resources. 4. Project Cost estimation.			✓		
PO12	Life-long learning	1. Team Work. 2. Deep Knowledge of Internet of Things, Machine Learning and python. 3. How to make a report.		✓			

rf

ORIGINALITY REPORT

12%	8%	7%	8%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.researchgate.net Internet Source	2%
2	Submitted to London Metropolitan University Student Paper	1%
3	Submitted to Visvesvaraya Technological University, Belagavi Student Paper	1%
4	Joe Dowling. "RFID-enabled temperature sensing devices: A major step forward for energy efficiency in home and industrial applications?", 2009 IEEE MTT-S International Microwave Workshop on Wireless Sensing Local Positioning and RFID, 09/2009 Publication	1%
5	Submitted to Siddaganga Institute of Technology Student Paper	1%
6	www.guru99.com Internet Source	1%
7	Submitted to Universiti Malaysia Pahang Student Paper	

		<1%
8	ia800402.us.archive.org Internet Source	<1%
9	Ying Bai. "Practical Database Programming with Visual C#.NET", Wiley, 2010 Publication	<1%
10	Submitted to Netaji Subhas Institute of Technology Student Paper	<1%
11	M. Mathankumar, N. Sugandhi. "A low cost smart shopping facilitator for visually impaired", 2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2013 Publication	<1%
12	Charles Bell. "Beginning Sensor Networks with Arduino and Raspberry Pi", Springer Nature, 2013 Publication	<1%
13	a4academics.com Internet Source	<1%
14	archive.org Internet Source	<1%
15	www.isendev.com Internet Source	<1%

16	www.shallowsky.com Internet Source	<1 %
17	Submitted to University of Lancaster Student Paper	<1 %
18	Submitted to Wakefield College Student Paper	<1 %
19	www.mgit.ac.in Internet Source	<1 %
20	Chi-Tsai Yeh, Ming-Chih Chen. "A combination of IoT and cloud application for automatic shrimp counting", Microsystem Technologies, 2019 Publication	<1 %
21	Submitted to Glyndwr University Student Paper	<1 %
22	defn.io Internet Source	<1 %
23	upcommons.upc.edu Internet Source	<1 %
24	Submitted to Kingston University Student Paper	<1 %
25	Submitted to University of Technology, Sydney Student Paper	<1 %
	www.iosrjournals.org	

26	Internet Source	<1 %
27	zir.nsk.hr Internet Source	<1 %
28	Submitted to Notre Dame Academy - CN-025946 Student Paper	<1 %
29	www.bugsearch.net Internet Source	<1 %
30	Loay Alzubaidi, Ghazanfar Latif, Jaafar Alghazo. "Affordable and Portable Realtime Saudi License Plate Recognition using SoC", 2019 2nd International Conference on new Trends in Computing Sciences (ICTCS), 2019 Publication	<1 %
31	prepinsta.com Internet Source	<1 %
32	am.b-ok.org Internet Source	<1 %
33	Dogan Ibrahim. "mikroC Pro for PIC32 Built-in Library Functions", Elsevier BV, 2014 Publication	<1 %
34	Xiaolin Jia, Quanyuan Feng, Taihua Fan, Quanshui Lei. "RFID technology and its applications in Internet of Things (IoT)", 2012 2nd International Conference on Consumer	<1 %

Electronics, Communications and Networks (CECNet), 2012

Publication

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		

VITAE

Name: Chethana R USN: 1SI18CS407 DOB: 01-06-1999 Permant Address: Agalerahatti, Kurubarahalli(post),Hiriyur(tq), Chitradurga(dist)-577599 Phone No. 8217264066 Email: chethancs250@gmail.com CGPA: 7.03(Upto 7th sem) Placed: No CTC:	
Name: G Kottresha USN: 1SI18CS409 DOB: 13-03-1998 Permant Address: No #0/00, Behind Maramma temple,Gandinagar Challakere-577522 Phone No. 9964938280 Email: gkotresh0@gmail.com CGPA: 7.07(Upto 7th sem) Placed: No CTC:	
Name: Kiran B A USN: 1SI18CS411 DOB: 10-06-1996 Permant Address: #02, Boochanahalli Pathagahalli post Koratagere Tq, Tumkur D-572129 Phone No. 7259626173 Email: kiranba491@gmail.com CGPA: 7.8(Upto 7th sem) Placed: Yes(Soroco) CTC: 6.5 L/PA	

Name: Ravichandra B S USN: 1SI18CS416 DOB: 12-05-1996 Permant Address: yaraganal(post), Nyamathi (tq), Davanagere (dist) Phone No. 8792981899 Email:ravichandrabs81@gmail.com CGPA: 6.96(Upto 7th sem) Placed: No CTC:	