University of Sri Jayewardenepura
Department of Computer Science

Project Report

On

KITCHEN INVENTORY MONITORING SYSTEM

Submitted by:

Name : N.L.M.Arachchi

Index No : AS2021926

(2020/2021 Batch)

Under the guidance of:

Mr.Dilum Perera

Senior Lecturer

Department of Computer Science

14.11.2024

DECLARATION

I, N.L.M.Arachchi hereby declare that the project work titled "Kitchen Inventory Monitoring System" is a record of original work carried out by me under the guidance of Mr.Dilum Perera. This project has not been submitted to any other institution or university for the award of any degree, diploma, or certification.

Navodya
14.11.2024
N.L.M.Arachchi

ACKNOLEDGMENT

I would like to express my sincere gratitude to Mr.Dilum Perera, Senior Lecturer at University of Sri Jayewardenepura for his invaluable guidance, encouragement and insightful feedback throughout the development of this project. His expertise and advice have greatly contributed to the success of this work. I am also grateful to the Department of Computer Science for providing the necessary resources and facilities to carry out this project.

I would also like to thank my friends and colleagues for their help and encouragement. Finally, I am deeply thankful to my family for their continuous support and understanding throughout this journey.

Table of Content

INTRODUCTION

In modern households, managing kitchen inventory efficiently is vital for maintaining a well-stocked kitchen, minimizing food wastage and optimizing grocery expenses. Tracking items manually can be time-consuming especially in busy households. The **Kitchen Inventory Monitoring System** resolves these challenges by automating inventory tracking with real-time information about current stock levels and providing information about market prices.

This project integrates ultrasonic sensors with baskets to monitor the levels of various kitchen items. The system is designed to update users visually through an LED screen and remotely via the Blynk app on a mobile device. Additionally, through web scraping techniques it retrieves real-time pricing and promotional offers from supermarkets enabling users to make cost-effective purchasing decisions. This system aims to simplify kitchen management and offer time-saving and cost-effective solutions by combining sensor technology, mobile connectivity and live price comparison.

PROJECT OBJECTIVES

1. Implement Inventory Tracking with Sensors
2. Display Real-Time Inventory Data
3. Incorporate Web Scraping for Price Comparison
4. Enable Remote Monitoring and Alerts
5. Develop a User-Friendly System

LITERATURE SERVEY

Several studies and solutions have been proposed for automated inventory management. Systems with IOT sensors and mobile apps for inventory tracking have shown potential for improving efficiency and reducing waste in domestic and commercial settings. Research has indicated that real-time data accessible through mobile applications significantly enhances the user experience in household inventory management. This project builds on these concepts by incorporating ultrasonic sensors and real-time web scraping for price comparison which is unique in its approach to provide both inventory tracking and cost-effective purchasing decisions.

SYSTEM DESIGN AND ARCHITECTURE

Block Diagram

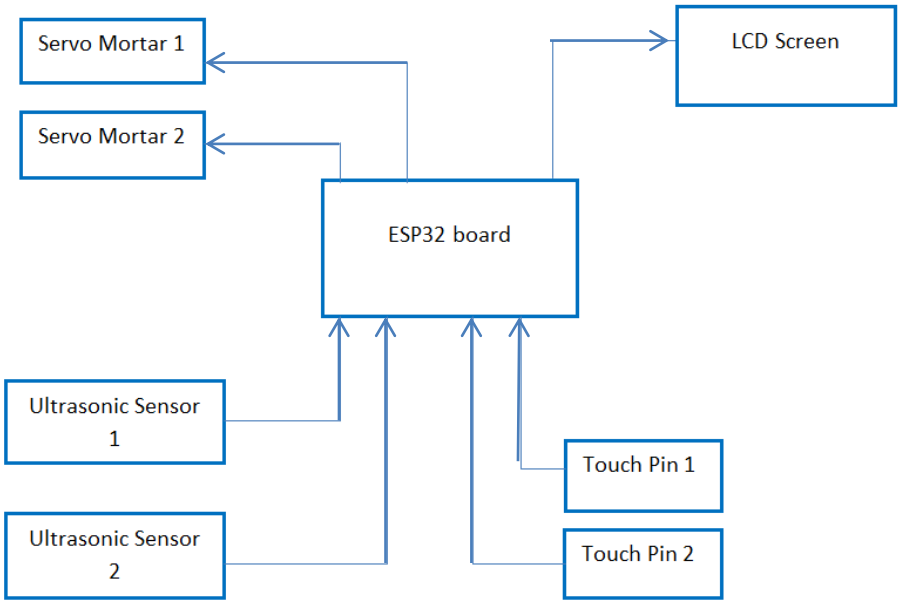


Figure 1: Block Diagram

System Components

Microcontroller:

The ESP32 is a versatile microcontroller with built-in Wi-Fi and Bluetooth capabilities that is ideal for IOT applications. Its processing power enables it to handle multiple sensors and data processing tasks efficiently. The Wi-Fi feature allows real-time data transmission to the mobile app enabling remote monitoring.

Figure 2 : ESP 32 board

Ultrasonic Sensor:

This sensor uses ultrasonic waves to measure the distance between the sensor and the items in the basket. By calculating the distance from the top of the basket to the item surface it can determine the fill level of the basket.

Figure 3 : Ultrasonic sensor

Servo Motor:

The servo motor is programmed to rotate and automatically open the basket lid when the touch sensor is activated providing convenient, hands-free access to items. It's controlled by signals from the ESP32 which determines when to trigger based on user interaction with the touch sensor.

Figure 4: Servo Motor

LCD I2C Display:

This 16x2 LCD display with I2C interface simplifies connections to the ESP32 and reduces wiring. It displays essential data like item levels and low-stock alerts. Real-time visibility of inventory levels on the display keeps users informed locally.

Figure 5: LCD Screen

Touch Pin:

The capacitive touch sensor acts as a user input for the system. When touched, it sends a signal to the ESP32 to activate the servo motor, which opens the basket. This feature enhances the user experience by allowing easy access to items without manual intervention

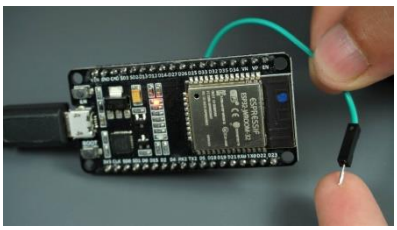


Figure 6: Touch Pin

Circuit Diagram and Pin Configuration

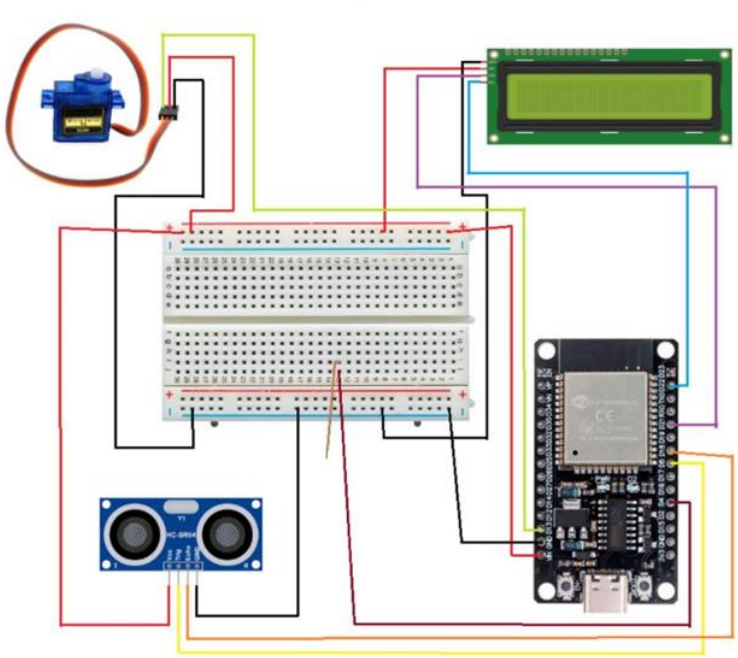


Figure 7: Circuit Diagram

- Ultrasonic Sensor 1 → ESP 32 board
  TrigPin —> GPIO 5
  EchoPin —> GPIO 18

- Ultrasonic Sensor 2 → ESP 32 board
  TrigPin —> GPIO 35
  EchoPin —> GPIO 34

- LCD Display (I2C) → ESP 32 board
  SDA —> GPIO 21
  SCL —> GPIO 22

- Servo Motor 1 → ESP 32 board
  SignalPin —> GPIO 13

- Servo Motor 2 → ESP 32 board
  SignalPin —> GPIO 12

- TouchPin → ESP 32 board
  GPIO 4
  GPIO 2

METHODOLOGY AND IMPLEMENTATION

System Operation

- Inventory Monitoring:

The ultrasonic sensor continuously measures the distance from the sensor to the items in the basket. The data is processed to calculate the inventory level as a percentage, which is then displayed on the LCD and Blynk app.

- Automatic Basket Opening:

When the user touches the touch pin, the servo motor is activated and the basket will open. This feature adds a layer of automation making it easier to access kitchen items.

- Real-Time Data Display:

The current inventory levels and alerts (e.g., low stock) are displayed on the LCD I2C display. This information is also transmitted to the Blynk app for remote access.

- Web Scraping for Offers:

The system includes a web scraping module that periodically checks specific supermarket websites for discounts and price updates on items identified as low in stock. The results are displayed in the Blynk app, helping the user take advantage of available deals.

Technologies Used

- Programming Environment :

Arduino IDE is used to develop and upload the program code to ESP32 using embedded C.

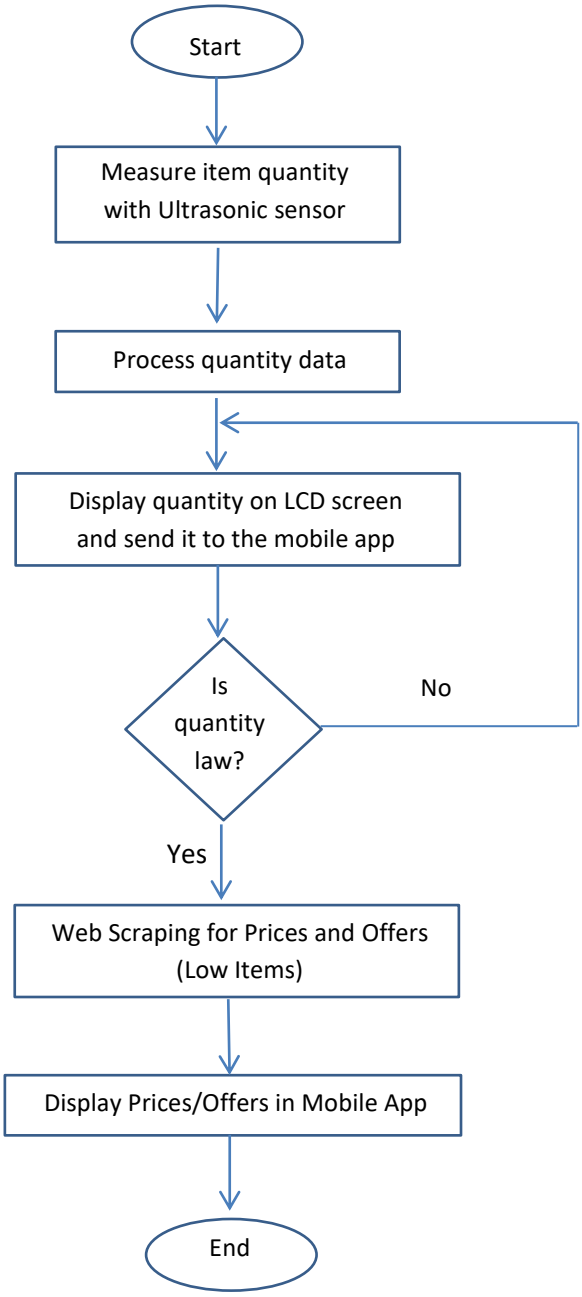
- Blynk Software :

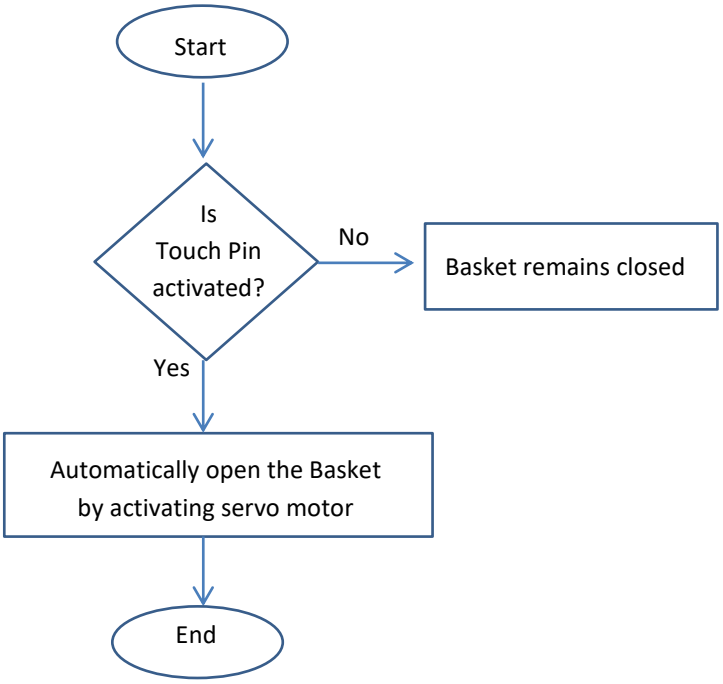
A mobile app interface that displays real-time data from the kitchen inventory system. The ESP32 board sends data to Blynk, enabling remote monitoring.

- Web Scraping Module :

Web scraping is data scraping used for extracting data from websites. Web scraping software may directly access the world wide web (www) using the Hypertext Transfer Protocol or a web browser. BeautifulSoup is the tool that used in this project. The data from this module is processed and sent to the Blynk interface.

Flow Chart


Start
Measure item quantity with Ultrasonic sensor
Process quantity data
Display quantity on LCD screen and send it to the mobile app
Is quantity law?
No
Yes
Web Scraping for Prices and Offers (Low Items)
Display Prices/Offers in Mobile App
End


Start
Is Touch Pin activated?
No
Basket remains closed
Yes
Automatically open the Basket by activating servo motor
End

Code Explanation

```
#define BLYNK_TEMPLATE_ID "TMPL6LLHs4YfH"
#define BLYNK_TEMPLATE_NAME "kitchen inventory system"
#define BLYNK_AUTH_TOKEN "gz31KOh7qmT3sGrdjTYcekOPutEinZKX"
#define BLYNK_PRINT Serial
```

These lines define the Blynk template information and authorization token required to connect to the Blynk platform and control the kitchen inventory system remotely.

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <BlynkSimpleEsp32.h>
#include <ESP32Servo.h>
```

These lines include the necessary libraries:

Wire.h : For communication with I2C devices (used by the LCD).
LiquidCrystal_I2C.h : To control the LCD screen.
BlynkSimpleEsp32.h : For connecting and interacting with Blynk using ESP32.
ESP32Servo.h : To control the servo motor.

```
Servo myservo;

#define basketHeight 13

//  ultrasonic sensor pins
const int trigPin = 5;  // Trigger pin
const int echoPin = 18; // Echo pin

long duration;
int distance;
int x_data[2];
int wholeSum;
```

myservo : Defines the servo motor.
basketHeight. : Defines the height of the basket as 13 cm.
trigPin and echoPin : Define the pins for the ultrasonic sensor to measure the distance.

Variables used for the measurements of ultrasonic sensor.
duration : the time that remains HIGH in the echo pin
distance : the calculated distance according to the duration
x_data[2] : two data points of distance that used calculate the average distance
wholeSum : sum of the two data points

```
// Initialize the LCD with I2C address 0x27 (this might need to be
changed depending on your LCD model)
LiquidCrystal_I2C lcd(0x27, 16, 2);

char auth[] = BLYNK_AUTH_TOKEN;
char ssid[] = "Malithi";
char pass[] = "11223344";
```

lcd : Initializes the LCD object with the I2C address 0x27 and a 16x2 character size.
auth, ssid, pass : Store the Blynk authentication token and Wi-Fi credentials for network connection.

```
BlynkTimer timer;

int MaxLevel = 100;
```

timer: Sets up a timer for handling recurring tasks in the Blynk system.
MaxLevel: Represents the maximum level percentage for the basket (100%).

```
void setup() {

  //ultrasonic
 pinMode(trigPin,OUTPUT);
 pinMode(echoPin,INPUT);

 // Start the serial communication
 Serial.begin(115200);

 Blynk.begin(auth, ssid, pass, "blynk.cloud", 80);

 myservo.setPeriodHertz(50);    // standard 50 hz servo(MG90 is a 50Hz Motor)
 myservo.attach(13, 1000, 2000); // attaches the servo on pin 13 to the servo object

 // Initialize the LCD
 lcd.begin();
 lcd.backlight();


}
```

pinMode(trigPin,OUTPUT) : define the trigPin as an output
pinMode(echoPin,INPUT) : define the echoPin as an input

Serial.begin(115200) : Initializes the serial communication at 115200 baud rate.

Blynk.begin() : Connects to the Blynk cloud using the auth token and Wi-Fi credentials.

Myservo : Configures the servo motor at 50 Hz (standard for MG90 servos) and attaches it to pin 13.
lcd.begin() and lcd.backlight() : I nitialize and turn on the LCD backlight.

```
void loop() {

  lcd.setCursor(0, 0);
  lcd.print("Level: ");
```

lcd.setCursor() and lcd.print() : Display the text “Level:” on the first line of the LCD.

```
 if(touchRead(4)<30){
     myservo.write(90);
}
 else if(touchRead(4)>30){
     myservo.write(0);
```

touchRead(4) ： Reads from the touch-sensitive pin (pin 4). If the touch reading is below 30, the servo opens (angle 90°); otherwise, it closes (angle 0°).

```
  //Ultrasonic sensor
 digitalWrite(trigPin, LOW);
 delayMicroseconds(2);
 digitalWrite(trigPin, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigPin, LOW);
 delayMicroseconds(10);
 duration = pulseIn(echoPin, HIGH);
 int dis = duration * 0.034 / 2;

//moving avarage filter for ultrasonic sensor
 for(int i =0;i<2;i++){
   x_data[i] = dis;
 }
 for(int j =0;j<2;j++){
   wholeSum = wholeSum + x_data[j] ;
 }
 distance = wholeSum / 2;


 x_data[2] ={0};
 wholeSum = 0;

 int persentage = ((basketHeight - distance) * 100)/ basketHeight;
```

Distance is measured by enabling the transmitter of the ultrasonic sensor and calculating the duration that get from the “pulseIn(echoPin, HIGH)”.

Then aplly moving average filter to get the stable data output.

Percentage : Calculates the percentage of the basket filled by subtracting the measured distance from the basket height and scaling it to 100%.

```
//int blynkDistance = (15-distance);
if (persentage <= MaxLevel) {
  Blynk.virtualWrite(V0, persentage);
} else {
  Blynk.virtualWrite(V0, 0);
}
```

Blynk.virtualWrite() : Sends the percentage value to Blynk's virtual pin V0 for display in the app.

```
// Print the distance on the LCD
lcd.setCursor(7, 0);
lcd.print(distance);
lcd.print(" cm  "); // Clear previous reading

lcd.setCursor(0, 1);
lcd.print("Presentage: ");

lcd.setCursor(11, 1);
lcd.print(persentage);
lcd.print(" % ");
```

Displays the measured distance and percentage on the LCD at appropriate positions.

```
// Print the distance to the serial monitor
Serial.print("Distance: ");
Serial.print(distance);
Serial.println(" cm");
```

Outputs the measured distance to the serial monitor for debugging.

```
// Add a short delay before the next reading
delay(100);
 lcd.clear();
}
```

delay(200) : Adds a 200 ms delay between readings.
lcd.clear() : Clears the LCD screen before the next reading.

WEB SCRAPING

```
import requests
from bs4 import BeautifulSoup
import blynklib
import schedule
import time


BLYNK_AUTH = "gz31KOh7qmT3sGrdjTYcekOPutEinZKX"


# URL of the Keells Super website
url = "https://www.keellssuper.com/productDetail?itemcode=8710&Brown_Sugar_Bulk_kg"

# Fetch the page
response = requests.get(url)
if response.status_code == 200:
    soup = BeautifulSoup(response.content, 'html.parser')

    # Example: Extract product names and prices
    products = soup.find_all('div', class_="productName col")  # Adjust based on the website's structure
    prices = soup.find_all('div', class_= "product-card-final-price")    # Adjust based on the website's structure

    for product, price in zip(products, prices):
        print(f"{product.text.strip()} - {price.text.strip()}")
else:
    print(f"Failed to fetch the website. Status Code: {response.status_code}")
```

```
def update_blynk_data():

# Initialize Blynk
blynk = blynklib.Blynk(BLYNK_AUTH)

# Data to send
product_data = "{product.text.strip()} - {price.text.strip()}"

# Virtual Pin to send the data
virtual_pin = 'V1'

# Push data to Blynk
blynk.virtual_write(virtual_pin, product_data)

    pass

# Schedule the task every hour
schedule.every(1).hours.do(update_blynk_data)

while True:
    schedule.run_pending()
    time.sleep(1)
```

Budget

Item	Unit Price	Quantity	Price
Basket 1	LKR 450	1	LKR 450
Basket 2	LKR 500	1	LKR 500
Ultrasonic sensor	LKR 240	2	LKR480
LCD screen	LKR 470	1	LKR 470
I2C module	LKR 200	1	LKR 200
Servo motor	LKR 350	2	LKR 700
Connecting wires	LKR 140	3 sets	LKR 420
Bread board	LKR 200	1	LKR 200
Dot board	LKR 170	1	LKR 170
ESP 32 board (from the Department)	LKR 0	1	LKR 0
Total Amount			LKR 3590

RESULT

CONCLUSION AND FUTURE WORKS

Conclusion

The Kitchen Inventory Monitoring System effectively automates inventory tracking in a kitchen environment providing real-time data on item levels through the ESP32 microcontroller, ultrasonic sensors and an LCD display. By integrating the Blynk app users can remotely monitor inventory levels and receive low-stock alerts enhancing convenience. The system's web scraping feature also retrieves current prices and offers for low-stock items supporting cost-effective restocking. This project demonstrates how IOT technology can simplify kitchen management, offering potential for future enhancements and broader applications in household automation.

Future Works

- **Expand Item Tracking Capabilities**:
  Including different types of sensors to monitor a wider range of items such as liquid-level sensors for bottles or weight sensors for bulk goods like grains making the system adaptable for various kitchen supplies.

- **Automate Shopping List Creation:**
  Based on the current inventory levels and predicted needs the system could automatically generate a shopping list or even integrate with online grocery platforms for direct ordering.

- **Expand Price Monitoring to Multiple Stores**:
  Broaden the web scraping feature to monitor prices from multiple stores allowing users to compare prices and find the best deals for items on their restocking list.