Project Report

Title: Wi-Fi-enabled Intelligent Vegetable Chopper

By:

V. Hima Vamsi

For More Information-Contact:

himavamsi20@gmail.com

Abstract

The fast-paced environment of modern restaurant kitchens demands lightning-fast efficiency and unwavering consistency. Every second counts during a dinner rush, and chefs require all hands on deck to deliver a flawless symphony of flavors on every plate. Traditional methods of vegetable chopping, while prevalent, become bottlenecks in this high-pressure environment. Manual chopping with knives is labor-intensive and time-consuming, hindering order fulfillment during peak hours. This not only slows down the kitchen but can also lead to inconsistencies in chop size and style, impacting both the visual presentation and overall quality of the dish. Furthermore, the use of knives introduces a safety risk for kitchen staff in the midst of the dinner rush frenzy. This project proposes the development of a Wi-Fi-enabled Intelligent Vegetable Chopper (wIVC) to address these difficulties. The wIVC leverages Internet of Things (IoT) technology to automate vegetable chopping tasks and improve kitchen efficiency. The system integrates with a Kitchen Display System (KDS) to present real-time order data. Based on this data, the wIVC utilizes a Wi-Fi-based microcontroller ESP8266- Nodemcu to control the chopping blades based on the vegetable type and desired chop size (e.g., dice, julienne) for optimal results. Additionally, it controls a gating mechanism within a dedicated vegetable storage unit whose temperature and the vegetable and ingredient spoilage are continuously monitored using DHT11 and Methane Gas sensors. The microcontroller instructs the gating mechanism to release the specific vegetables needed for preparing an order, ensuring precise ingredient selection. This project aims to eliminate the challenges of time-consuming and inconsistent vegetable chopping in restaurant kitchens. The wIVC offers several potential benefits: Increased chopping efficiency, Enhanced consistency, Reduced labor costs, and Improved safety.

Keywords: Wi-Fi-enabled, Internet of Things (IoT) Technology, Automate vegetable chopping, ESP8266-Nodemcu, Continuous Monitoring, DHT11, Gas Sensor

Table of Contents :

Chapter 1: Introduction

The heart of a restaurant throbs with the rhythm of efficiency and quality. But this symphony can be disrupted by the time-consuming task of vegetable chopping. Traditional methods, reliant on a chef's knife, create bottlenecks during peak hours.

A Modern Restaurant Kitchen: A Ballet of Efficiency

Before diving into the wIVC solution, let's take a moment to appreciate the intricate dance that unfolds within a modern restaurant kitchen. It's a ballet of efficiency, with different stations working together like a well-oiled machine to get food out fast and delicious. Here's a breakdown of the flow:

- **Orders In:**
  - Traditionally: Servers would verbally relay orders to the kitchen, which could be prone to errors.
  - Modern Kitchens: Orders are often sent electronically to a Kitchen Display System (KDS) in the kitchen. This digital board shows each order, with details and customizations, ensuring clarity.
- **The Kitchen Brigades:** Restaurants are divided into stations, each with a designated chef and sometimes assistants. These stations can vary depending on the menu, but some common ones include **Appetizer Station**: Prepares starters and salads. ***Entrée Station**: The main course assembly and cooking happens here. * Grill Station: Focuses on grilled meats, fish, and seafood. **Sauté Station**: Handles stir-frying, pan-searing, and delicate dishes. **Fry Station:** For all your deep-fried favourites.
- **The Order Journey:**
  - Once an order hits the KDS, the head chef, or a designated person, calls out the order.
  - Each station prepares its assigned part of the dish. Communication is key to ensure everything is cooked at the right pace.
  - The plated dishes are then assembled and reviewed by an expediter (sometimes the head chef) for quality and presentation before being sent out.
- **The Kitchen Staff:** The number of people working on an order depends on the size and complexity of the dish, but here's a general idea: * **Head Chef**: Oversees the entire kitchen, manages the flow, and ensures quality. * **Station Chefs**: Lead cooks responsible for their assigned areas. * **Line Cooks**: Assist station chefs with prep and cooking tasks. * **Expediter**: Ensures plates are assembled correctly and meet quality standards. * **Dishwashers (Plungers):** Maintain a clean and organized kitchen.

Back to the Disrupted Symphony: How Manual Chopping Hinders Efficiency

Now, let's return to the original disruption in this culinary symphony – vegetable chopping. Traditional methods, reliant on a chef's knife, create bottlenecks during peak hours. Manual chopping is labor-intensive, forcing skilled staff to focus on a repetitive task when their expertise could be better utilized elsewhere. Furthermore, this approach suffers from inconsistencies in chop size and style, affecting both the visual presentation and textural harmony of a dish. Perhaps the most concerning aspect is the inherent safety risk associated with wielding knives throughout a busy shift. The wIVC relieves line cooks of repetitive vegetable chopping, allowing them to focus on higher-level culinary tasks. As a result, they

can utilize their skills more effectively and contribute to a more creative and fulfilling workplace.

This project proposes the Wi-Fi-enabled Intelligent Vegetable Chopper (wIVC), a revolutionary solution that addresses these limitations. The wIVC leverages the power of Internet of Things (IoT) technology to automate vegetable chopping tasks, transforming kitchen efficiency. Seamlessly integrated with a Kitchen Display System (KDS), the wIVC receives real-time order data. This data is then used by a Wi-Fi-based microcontroller to control DC motors driving the chopping blades. The microcontroller can dynamically adjust motor speed based on the specific vegetable and desired chop size (e.g., dice, julienne), ensuring optimal results. Additionally, a servo motor manages a gating mechanism within a dedicated vegetable storage unit, allowing for the precise selection of ingredients needed for each order.

By automating chopping tasks, the wIVC offers a multitude of benefits:

- Unleashing Culinary Potential: Kitchen staff are freed from repetitive tasks, allowing them to focus on higher-level cooking and plating, ultimately elevating the overall dining experience.
- Consistent Culinary Creations: The wIVC guarantees uniform chop size and style, leading to a more visually appealing and texturally consistent presentation across every dish.
- Optimizing Labor Costs: In high-volume kitchens, the wIVC can potentially reduce the need for dedicated prep cooks, offering a path to reduced labor costs.
- A Safer Kitchen Environment: Eliminating the constant use of knives minimizes the risk of injuries for kitchen staff, fostering a safer work environment.

The wIVC represents a leap forward in kitchen automation. The following sections will delve deeper into existing methods, explore the proposed methodology of the wIVC, and unveil exciting possibilities for future development.

Chapter 2: Background and Proposed Methodology

Addressing the Limitations of Traditional Techniques: Existing Methods

Restaurant kitchens require a delicate balance between speed and precision, especially when it comes to vegetable preparation. However, traditional methods for chopping vegetables often fall short in this regard. Here, we explore the limitations of the two most common approaches:

- **Traditional Knife Chopping:** This remains the most common method, but it suffers from several drawbacks:

  - **Labor-intensive:** Requires significant prep cook time, hindering efficiency during peak hours.
  - **Inconsistency:** Chop size and style can vary depending on the skill and fatigue level of the chef, impacting dish presentation and quality.
  - **Safety Risk:** Wielding knives throughout a shift poses a potential danger of cuts and injuries.

- **Commercial Food Processors:** These are large, bulky machines better suited for large-scale vegetable preparation, not individual dish chopping. They lack the precision needed for specific cuts and are not ideal for a fast-paced kitchen environment.

A Vision for Efficiency: The Proposed wIVC System

The limitations of traditional methods highlight the need for innovative solutions. This section introduces the **Wi-Fi-enabled Intelligent Vegetable Chopper (wIVC)**, a novel system designed to revolutionize vegetable chopping tasks in restaurant kitchens. The wIVC leverages **Internet of Things (IoT)** technology to automate the chopping process, offering numerous benefits to improve efficiency, consistency, and safety in the kitchen.

Components:

- **Wi-Fi Microcontroller:** The brain of the system, connecting to the KDS via Wi-Fi to receive real-time order data. It controls the DC motors, and servo motor, and monitors system functions. For this system, the ESP8266-Nodemcu microcontroller was found to be best.
- **DC Motors (with variable speed control):** These motors drive the chopping blades. The microcontroller dynamically adjusts the motor's run time to achieve the desired chop size. Longer run times result in finer chops (smaller pieces), while shorter run times produce larger chunks.
- **Servo Motor:** This motor operates a **gating mechanism** within a dedicated **vegetable storage unit**. The microcontroller instructs the servo motor to open the specific compartment containing the vegetables needed for the order.
- **Chopping Blade Assembly:** Interchangeable blades designed for different chopping styles (dice, julienne, slice, etc.) allow for versatility in vegetable cuts.
- **Hopper:** Holds the vegetables to be chopped.
- **Discharge Chute:** Directs chopped vegetables to a collection bin.

- **DHT11 & Gas Sensors:** The wIVC incorporates a DHT11 sensor to monitor the temperature within the vegetable storage unit. Additionally, a gas sensor is integrated to detect gases emitted by spoiling vegetables. This proactive spoilage detection helps prevent the use of outdated ingredients, promoting food safety and reducing waste.

<u>System Architecture</u> (refer to Figure 1):

1. **Order Received:** The Waiters app transmits order details to a secure cloud platform. The wIVC microcontroller then retrieves this information.
2. **Ingredient Selection:** The microcontroller instructs the servo motor to activate the gating mechanism, releasing the specific vegetable from its designated compartment in the storage unit.
3. **Vegetable Loading:** The released vegetable falls into the hopper.
4. **Chopping Parameters:** The microcontroller selects the appropriate chopping time and sets the DC motor.
5. **Chopping Process:** The DC motors activate the chopping blades, processing the vegetables in the hopper.
6. **Discharge:** Chopped vegetables fall through the discharge chute into a collection bin.
7. **Storage Monitoring:** The microcontroller continuously monitors the inside temperature of the storage system and checks for ingredient spoilage.

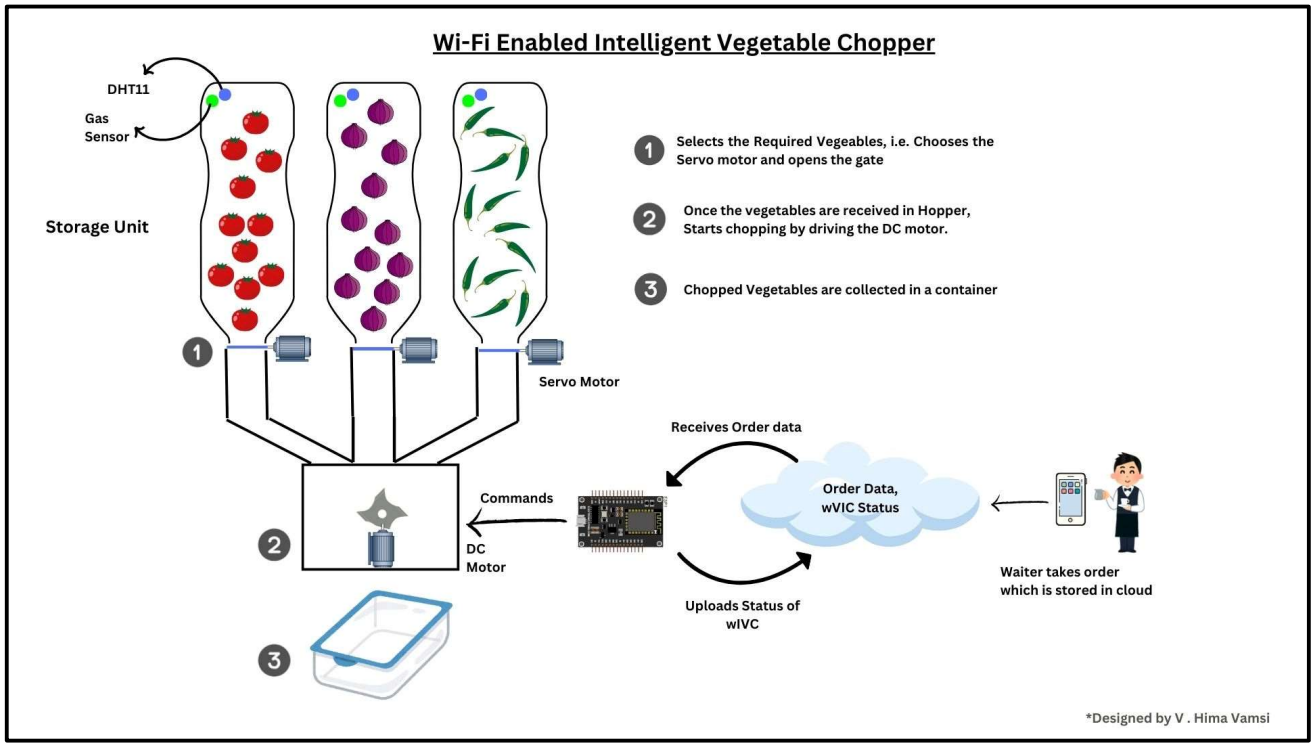

Figure 1 : System Architecture

Chapter 3: Hardware and Software Requirements

Hardware Requirements:

1. ESP8266-Nodemcu

The ESP8266-Nodemcu is a low-cost, Wi-Fi-enabled microcontroller board designed for IoT applications. It integrates a powerful ESP8266 microchip with full TCP/IP stack and microcontroller capabilities, making it an ideal choice for connected devices. Key features include a 32-bit RISC CPU running at 80 MHz, 4 MB of flash memory, and 64 KB of instruction RAM. The ESP8266 supports multiple interfaces such as UART, SPI, and I2C, allowing easy integration with various sensors and peripherals. It also includes a built-in Wi-Fi module, enabling seamless wireless communication. The NodeMCU firmware provides a user-friendly platform for programming in Lua or Arduino IDE, making it accessible for both beginners and experienced developers.

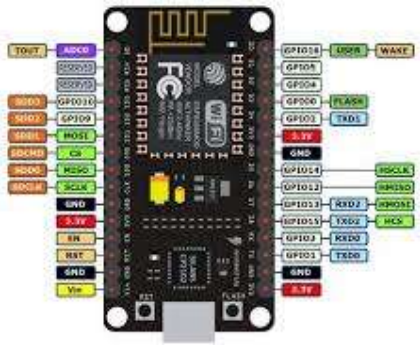

Figure 2: ESP8266-Nodemcu

2. L293D Motor Driver

The L293D Motor Driver is a quadruple high-current half-H driver designed to control DC motors and stepper motors. It is capable of driving up to 600 mA of continuous current per channel and up to 1.2 A of peak current per channel. The L293D has four H-bridge drivers, which allow for independent control of two DC motors or one stepper motor. It operates over a wide voltage range of 4.5V to 36V, making it versatile for various motor control applications. Key characteristics include built-in diodes for back EMF protection, thermal shutdown, and high noise immunity. The L293D is widely used in robotics and automation projects due to its reliability and ease of use.

Figure 3: L293D Motor Driver

3. DC Motor

A DC motor is an electromechanical device that converts direct current electrical energy into mechanical energy. DC motors are characterized by their ability to provide precise control of speed and position. They typically consist of a stator, rotor, brushes, and a commutator. DC motors are known for their high torque at low speeds, making them suitable for applications requiring strong and consistent force. They are commonly used in various applications such as robotics, automation, and household appliances. The speed of a DC motor can be controlled by adjusting the supply voltage or through pulse-width modulation (PWM), allowing for versatile performance in different scenarios.

Figure 4: DC Motor

4. Servo Motor

A servo motor is a rotary actuator that provides precise control of angular position, speed, and acceleration. It consists of a motor coupled with a sensor for position feedback and a control circuit. Servo motors are characterized by their high torque and accuracy, making them ideal for applications requiring precise movement, such as robotics, CNC machinery, and remote-controlled vehicles. They typically operate within a 0 to 180-degree range, but some advanced models offer continuous rotation. Servo motors receive control signals in the form of PWM, which dictates the desired position. Their ability to maintain a specified position makes them essential for tasks requiring exact movements.

Figure 5: Servo Motor

5. DHT11 Sensor

The DHT11 is a basic, low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air and provides a digital signal on the data pin. The DHT11 offers a temperature measurement range of 0 to 50°C with an accuracy of ±2°C and a humidity measurement range of 20 to 90% RH with an accuracy of ±5% RH. It operates with a supply voltage of 3.3V to 5V and has a low power consumption, making it suitable for battery-operated devices. The sensor outputs data in a digital format, simplifying integration with microcontrollers and other digital systems. Its compact size and simplicity make it popular for use in environmental monitoring projects.

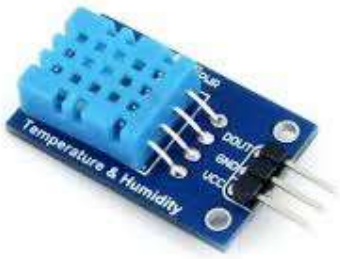

Figure 6: DHT11 Sensor

6. Gas Sensor

A gas sensor is a device that detects the presence and concentration of various gases in the environment. The specific gas sensor mentioned, likely a methane gas sensor, is designed to detect methane (CH4) and other combustible gases. It operates based on the principle of catalytic combustion or semiconductor sensing. Key characteristics of a gas sensor include high sensitivity, fast response time, and stability over time. Gas sensors are commonly used in industrial safety, environmental monitoring, and smart home applications to detect gas leaks and ensure air quality. They typically output an analog signal proportional to the gas concentration, which can be read by a microcontroller for further processing and action.

Figure 7: Gas Sensor

Software Requirements:

1. Arduino IDE

The Arduino Integrated Development Environment (IDE) will be our primary tool for programming the microcontrollers embedded within the streetlight monitoring devices. Arduino IDE offers a user-friendly interface and extensive libraries, making it ideal for developing code to interact with the sensors and communication modules.

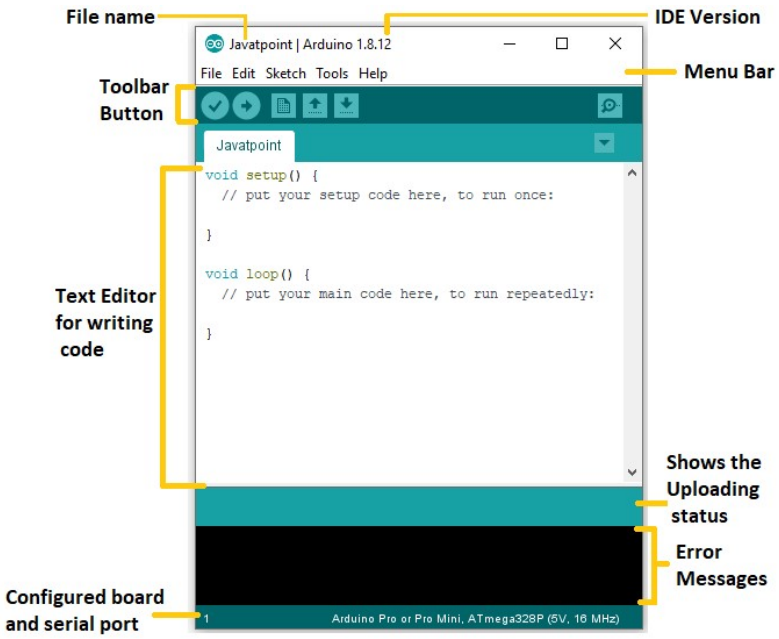

Figure: 8 Arduino IDE

- **Verify / Upload** - compile and upload your code to your Arduino Board.
- **Select Board & Port** - detected Arduino boards automatically show up here, along with the port number.
- **Sketchbook** - here you will find all of your sketches locally stored on your computer. Additionally, the latest versions can sync with the **Arduino Cloud**, and also obtain your sketches from the online environment.
- **Boards Manager** - browse through Arduino & third-party packages that can be installed. For example, using a MKR Wi-Fi 1010 board requires the
- **Library Manager** - browse through thousands of Arduino libraries, made by Arduino & its community.
- **Debugger** - test and debug programs in real time.
- **Search** - search for keywords in your code.
- **Open Serial Monitor** - opens the Serial Monitor tool, as a new tab in the console.

2. Thingspeak

We will be utilizing ThingSpeak as the cloud platform for our system. This popular IoT platform offers a user-friendly interface for data storage, visualization, and integration with various tools. ThingSpeak's robust capabilities ensure secure data storage and facilitate seamless communication between the hardware components and the software applications.

- Data collection: Send data from devices, sensors, and instruments to the cloud.
- Data analysis: Use online analytical tools to explore and visualize data, calculate new data, and visualize it in plots, charts, and gauges.
- Data action: Trigger actions based on data, such as receiving a tweet when the temperature goes above 70° F.
- Data storage: Store data in channels, with each channel storing up to 8 fields of data.
- Data visualization: Create instant visualizations of live data.
- Data alerts: Send alerts using web services like Twitter and Twilio
- Event scheduling: Schedule actions to be performed at different times or on a specific schedule.

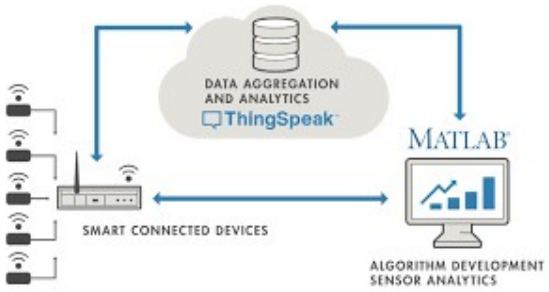

Figure 9: ThingSpeak Cloud

3. MIT App Inventor

MIT App Inventor is a web-based integrated development environment (IDE) created by Google and now maintained by MIT. It enables users to create Android apps through a visual, drag-and-drop interface, making it accessible for beginners, especially students and educators.

Key Features:

1. **Visual Programming Interface**: Users can build apps by connecting blocks that represent different functions and actions, simplifying the coding process.
2. **Component-Based Design**: The platform offers a variety of built-in components such as user interface elements, sensors, and connectivity options that can be easily added to apps.
3. **Real-Time Testing and Debugging**: Users can test their apps in real-time on an Android device or emulator, allowing immediate feedback and adjustments.
4. **Extensibility**: Advanced users can add custom features and functionality using App Inventor Extensions.

5. **Educational Focus**: It provides extensive resources, including tutorials and curriculum guides, to help users learn app development.
6. **Cross-Platform Development**: Supports both Android and iOS, enabling users to create apps for multiple platforms.
7. **Cloud-Based Storage**: Projects are stored in the cloud, facilitating easy access and collaboration from any computer.
8. **User-Friendly Documentation and Support**: Offers comprehensive guides, tutorials, and forums to assist users at all levels.

MIT App Inventor democratizes app development, making it easy for anyone to create functional and innovative mobile applications, with a strong emphasis on education and accessibility.

Figure 10: MIT App Inventor

Chapter 4: System Implementation

Having established the essential building blocks in the previous chapters, we now embark on the exciting journey of transforming these components into a cohesive and functional system. This chapter delves into the step-by-step process of system implementation. We'll explore how the hardware components are integrated and programmed to work in unison

Hardware Integration:

Now that the hardware components have been carefully chosen, it's time to bring them together to form the physical foundation of our system. This section will detail the process of integrating these components. As per the connections depicted in Figure 11 the hardware components are integrated.

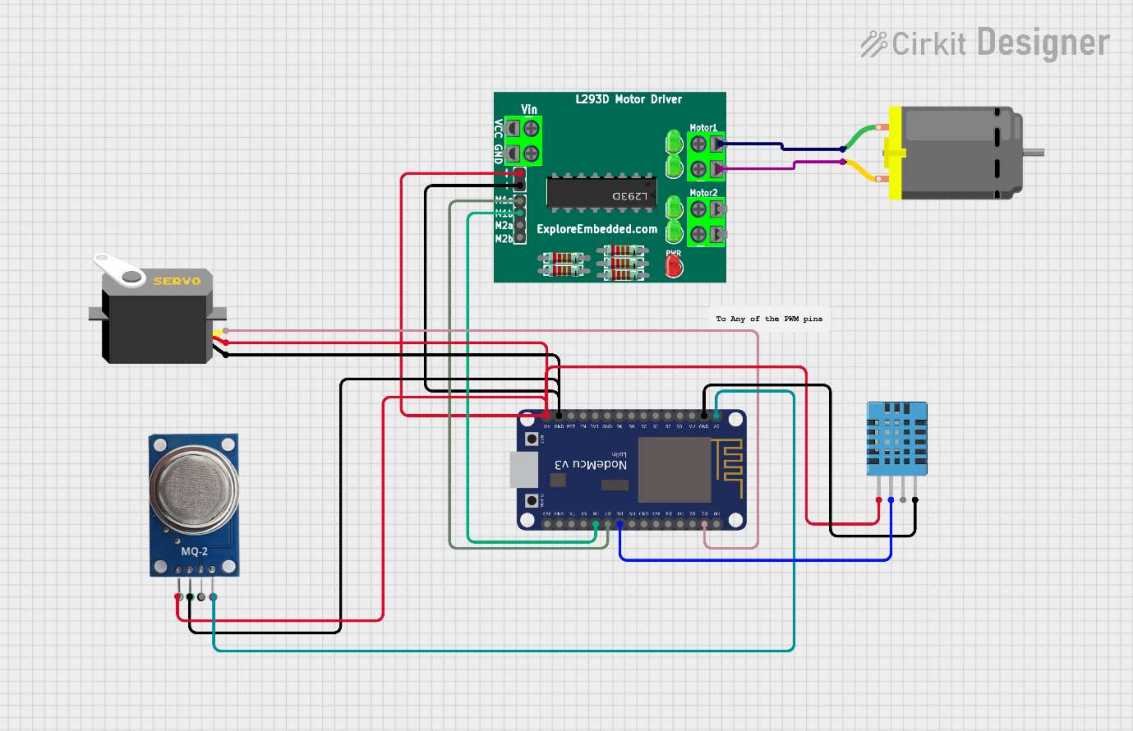

Figure 11: Hardware Integration

Hardware Programming:

The NodeMCU is programmed using the Arduino IDE, which provides a user-friendly environment for writing, compiling, and uploading code to the microcontroller. The code provided below demonstrates how to utilize the Arduino IDE to program the NodeMCU for specific tasks, such as connecting to Wi-Fi, reading sensor data, and controlling various

hardware components. This setup allows for seamless integration and efficient communication between the NodeMCU and other connected devices, ensuring smooth operation and functionality of the overall system. The detailed code example highlights the steps and commands necessary to achieve the desired functionality.

Code:

```
#include <ESP8266WiFi.h>
#include "secrets.h"
#include "ThingSpeak.h"


#include <Servo.h>
#include <DHT.h>
#include <DHT_U.h>
#define DHTPIN 14
DHT dht(DHTPIN,  DHT11);

const int servoPin = 5;
const int enablePin = 12;
const int motorPin1 = 13;
const int motorPin2 = 15;
int motorSpeed = 250; // Adjust motor speed (0-255)
Servo myservo;

// Wifi Credentials
char ssid[] = "motorola edge 20 fusion";
char pass[] = "vamsivamsi";
int keyIndex = 0;
WiFiClient  client;

// Thingspeak Channels Credentials

// Order channel
unsigned long channelid = 2588976;
const char* readapikey = "IGOM0DEHIZ32FCGR" ;
const char* myWriteAPIKey = "0CIJ3OZKXMW2R7KG";
unsigned int fieldnumber = 1;

// Gas Sensor channel details
unsigned long gaschannelid = 2552926;
const char* g_readapikey = "60L0Q03274DL72DD";
const char* g_writeapikey = "9OYXGBLB2LH1364I";

// T&H Channel Details
unsigned long tandhchannelid = 2538812;
const char* tandh_readapikey = "HP66GG7YLH8HC53U";
const char* tandh_writeapikey = "V1S1TBLQBK7RSOCW";
```

```
void setup() {

  Serial.begin(115200);  // Initialize serial monitor

  // Setting Pins
  pinMode(servoPin, OUTPUT);
  pinMode(enablePin, OUTPUT);
  pinMode(motorPin1, OUTPUT);
  pinMode(motorPin2, OUTPUT);

  // Initalizing Wifi and Thingspeak

  WiFi.mode(WIFI_STA);
  ThingSpeak.begin(client);  // Initialize ThingSpeak
   // Connecting or reconnecting to WiFi
  if(WiFi.status() != WL_CONNECTED){
    Serial.print("Attempting to connect to SSID: ");
    Serial.println(SECRET_SSID);
    while(WiFi.status() != WL_CONNECTED){
      WiFi.begin(ssid, pass);
      Serial.print(".");
      delay(5000);
    }
    Serial.println("\nConnected");
  }

  // Closing gate
  myservo.attach(servoPin);
  myservo.write(0);

}

void loop() {

  // Checking Storage's Temperature, Humidity and Spoilage
  int gas =analogRead(A0);
  Serial.print("Gas :");
  Serial.println(gas);
  ThingSpeak.writeField(gaschannelid, 1, gas, g_writeapikey);;

  int h = dht.readHumidity();
  int t = dht.readTemperature();
  int hic = dht.computeHeatIndex(t, h, false);

  Serial.print("Temperature : ");
  Serial.println(hic);
  ThingSpeak.writeField(tandhchannelid, 1, hic, tandh_writeapikey);;
  Serial.print("Humidity : ");
  Serial.println(h);
```

```
  ThingSpeak.writeField(tandhchannelid, 2, h, tandh_writeapikey);;

  // Reading order
  int DishInF = ThingSpeak.readFloatField(channelid, fieldnumber, readapikey);

  Serial.print("Order info: ");
  Serial.println(DishInF);


  if (DishInF == 1)
  {
    startSERVOmotor();
    startDCmotor(5000);
  }
  else if (DishInF == 2)
  {
    startSERVOmotor();
    startDCmotor(10000);
  }
  else if (DishInF == 3)
  {

    startSERVOmotor();
    startDCmotor(15000);
  }
  else {
    // Stop servo and motor
    myservo.write(0);
    digitalWrite(enablePin, LOW);
    Serial.println("wVIC State: Not chopping");
  }

  delay(500);

}

void startSERVOmotor()
{
  // Opening Storage Gate
  for (int pos = 0; pos <= 180; pos += 1) {
      myservo.write(pos);
      delay(15);
    }
  // Closing Storage Gate
    for (int pos = 180; pos >= 0; pos -= 1) {
      myservo.write(pos);
      delay(15);
    }
}
```

```
int startDCmotor(int d)
{

  digitalWrite(enablePin, HIGH);  // Enable motor driver
  digitalWrite(motorPin1, HIGH);  // Forward
  digitalWrite(motorPin2, LOW);
  Serial.println("wVIC: Chopping");
  delay(d);

  digitalWrite(enablePin, LOW);  // Stop motor
  return 0;
}
```

Apps Creation:

Two apps, WaiterWizz and KD (Kitchen Display), are developed using MIT App Inventor. WaiterWizz is designed for waiters to collect customer orders efficiently and update this information to the Thingspeak cloud in real time. In contrast, the KD app serves as a Kitchen Display system that provides a comprehensive view of all incoming orders. It displays order details, current status updates, and the status of the storage unit.

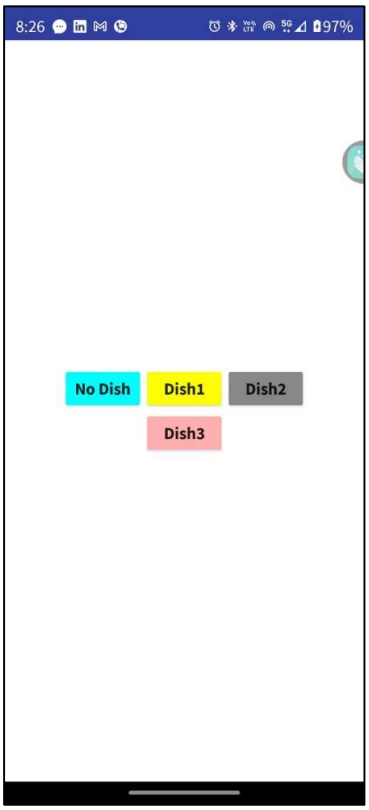

Figure 12: WaiterWizz App

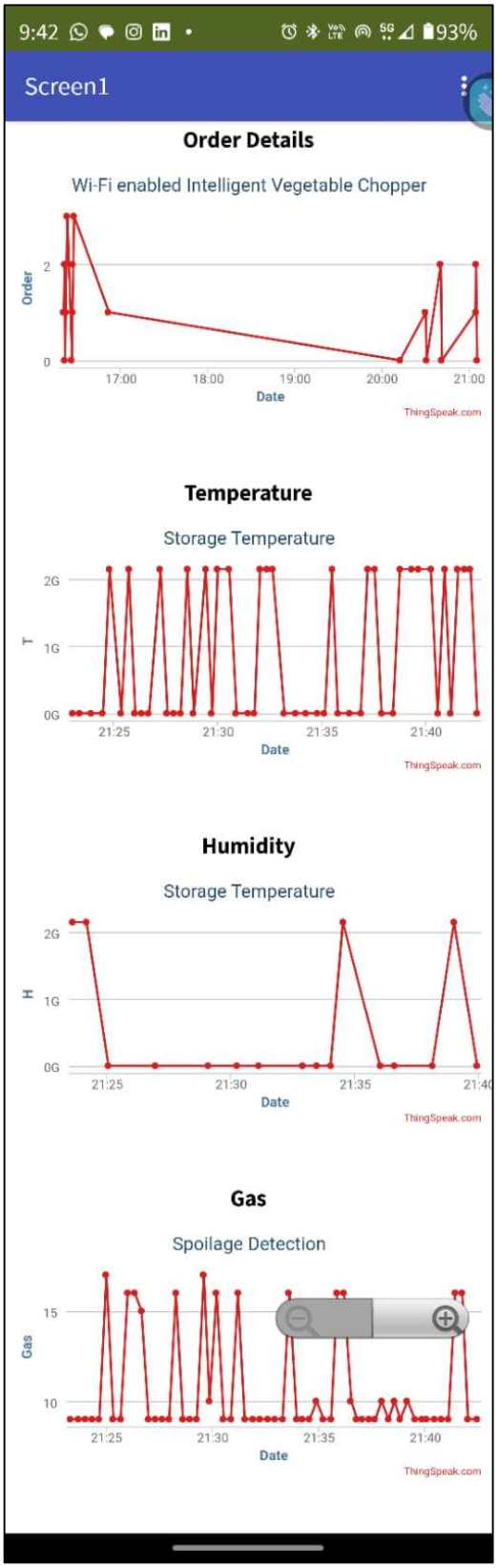

Figure 13: Kitchen Display App

Chapter 5: Results and Discussions

The Prototype of the wIVC is shown in Figure 14. Using the WaiterWizz app the Order data is sent to the Thingspeak cloud. The wIVC retrieves this data and acts according to this data i.e. Opens the gate and drives the chopper and finally we get the chopped vegetables.

```
09:26:20.909 -> Order info: 0
09:26:20.955 -> wVIC State: Not chopping
09:26:21.411 -> Gas :9
09:26:23.028 -> Temperature : 34
09:26:24.627 -> Humidity : 77
09:26:28.078 -> Order info: 0
09:26:28.120 -> wVIC State: Not chopping
09:26:28.580 -> Gas :9
09:26:30.194 -> Temperature : 34
09:26:31.589 -> Humidity : 77
09:26:34.633 -> Order info: 0
09:26:34.690 -> wVIC State: Not chopping
09:26:35.168 -> Gas :9
09:26:36.491 -> Temperature : 34
09:26:37.917 -> Humidity : 77
```

```
09:27:14.371 -> Order info: 2
09:27:21.626 -> wVIC: Chopping
09:27:32.089 -> Gas :16
09:27:34.096 -> Temperature : 34
09:27:35.477 -> Humidity : 77
```

```
09:28:49.621 -> Order info: 3
09:28:56.832 -> wVIC: Chopping
09:29:12.372 -> Gas :17
09:29:14.815 -> Temperature : 34
09:29:16.220 -> Humidity : 77
09:29:19.322 -> Order info: 3
```

Figure 14: Serial Monitor Output

Prototype

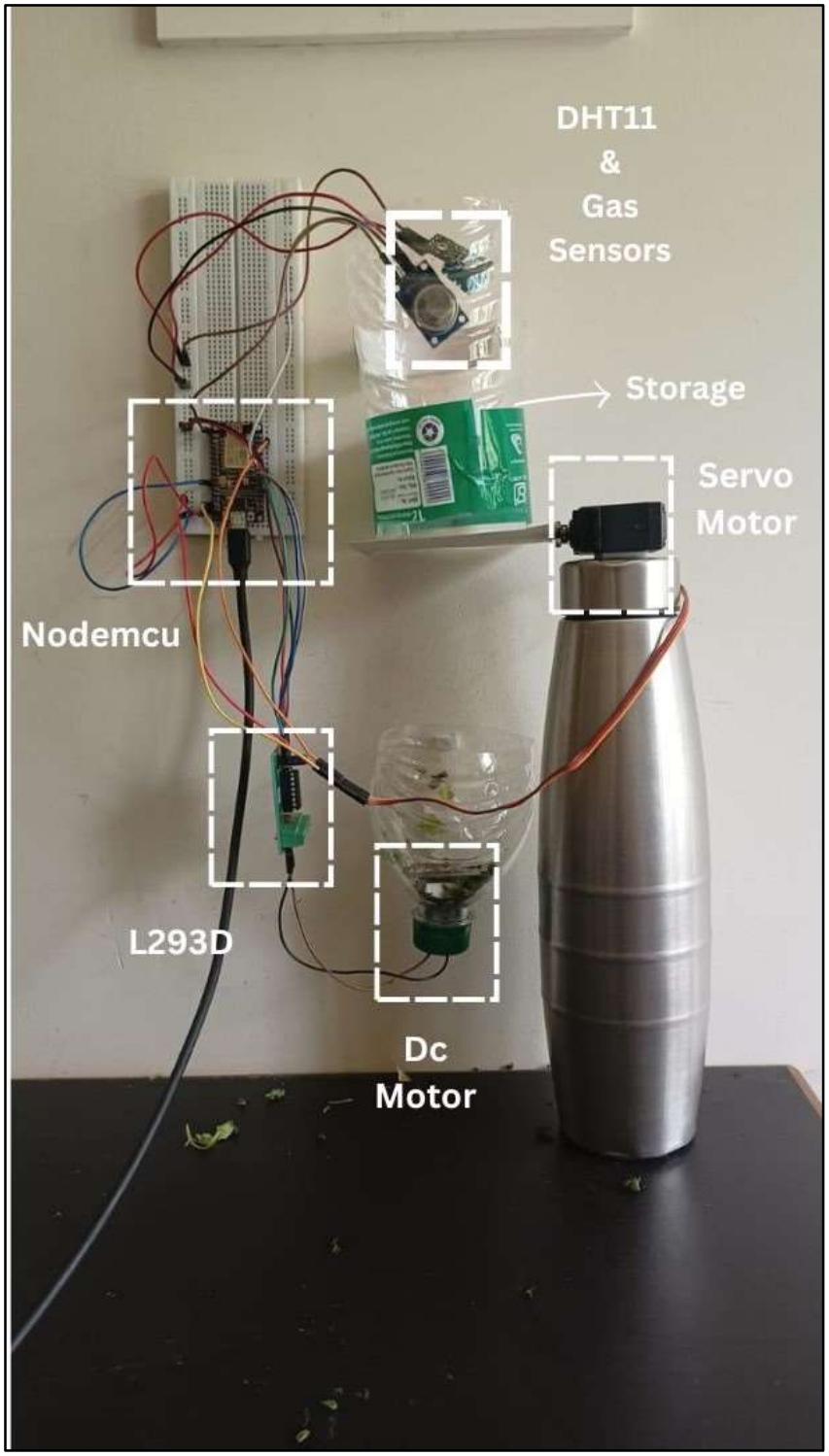

Figure 15: wIVC Prototype

Chapter 6: Conclusion, Future Scope and Refferences

Conclusion:

The Wi-Fi-enabled Intelligent Vegetable Chopper (wIVC) promises a brighter future for restaurant kitchens. Automating the time-consuming task of vegetable chopping unlocks a cascade of benefits. Faster chopping speeds and a streamlined workflow lead to quicker order fulfillment, while precise control over chop size ensures consistent presentation and textural harmony across dishes. The wIVC can potentially reduce the need for dedicated prep cooks, freeing valuable staff for higher-level culinary tasks, and eliminating the constant use of knives minimizes the risk of injuries. Ultimately, by freeing chefs from repetitive chopping, the wIVC elevates culinary potential, allowing them to focus on more creative endeavors and enrich the dining experience. This technology represents a significant step forward, and future iterations hold exciting possibilities, such as integration with recipe management software for automatic parameter adjustments, advanced spoilage detection for a wider range of ingredients, and even self-cleaning mechanisms for enhanced hygiene. By continuing to develop and refine the wIVC, we can contribute to a future where restaurant kitchens are characterized by efficiency, consistency, and a renewed focus on culinary creativity.

Future Scope:

The wIVC prototype paves the way for exciting advancements in automated vegetable chopping technology. Future iterations could see seamless recipe integration, where the wIVC automatically adjusts chopping parameters based on digital recipes. This not only streamlines workflow but also ensures consistent chop size across dishes. More sophisticated gas sensors could be incorporated to detect a wider range of spoilage markers in vegetables, promoting food safety and minimizing waste. For enhanced hygiene, self-cleaning mechanisms could be integrated for automatic sanitation of blades and the storage unit. Voice control integration would allow hands-free operation, while machine learning algorithms could analyze data to autonomously adjust settings for different vegetables, further optimizing efficiency and consistency. By exploring these possibilities, the wIVC can evolve into an intelligent kitchen assistant, transforming restaurant kitchens by prioritizing efficiency, safety, and hygiene while empowering chefs to unleash their creativity and deliver exceptional dining experiences.

References:

K. Aditya Pratap, S. Roji Marjorie, M. Saikumar, **Automatic vegetable chopper using image processing**, Materials Today: Proceedings, Volume 33, Part 7, 2020, Pages 4787-4789. https://doi.org/10.1016/j.matpr.2020.08.365