



Palestine Polytechnic University
College of Information Technology and Computer Engineering

Waiter Robot

Team Members:

Mahmoud Omar lahaseh

Marwan Omar Rayyn

Supervisor:

Eng.Elayan Abu-Gharbyeh

Hebron - Palestine

May , 2023

Acknowledgment

In the name of "Allah", the most beneficent and merciful who gave us strength, knowledge and helped us get through this project.

This project gave us valuable experience in machine learning. For that, we want to thank all the people who helped us through the project.

We would like to express our gratitude to our graduation project supervisors ENG. Elayan Abu-Gharbyeh, who oversaw our graduation project, for his leadership, support, and encouragement during the project.

Furthermore, we have to show our gratitude to Eng. Wael Takrouri for answering our questions and providing us with valuable suggestions and tips that helped us in our project, especially in the hardware problems. Moreover, we must thank our families for their generous encouragement, and continuous support throughout our life. For our friends, we are truly grateful for all of your support and help during the project and throughout the educational stage.

Finally, we would like to thank all the people who helped and supported us even with their feelings.

Abstract

During peak hours, large and popular restaurants often struggle to provide satisfactory service to their customers, leading to long waiting times for orders and deliveries due to overcrowding. To address this challenge, we have developed a specialized project aimed at enhancing customer service and meeting the unique needs of restaurants. Our solution involves the integration of a bot and a customized web page, working together to optimize in-restaurant operations.

At the heart of our system lies an advanced algorithm based on the Robot Operating System (ROS). This algorithm enables the robot to intelligently determine the most efficient path to fulfill customer orders. By carefully considering factors such as mapping and localization, the robot can select the optimal routes, minimizing the distance traveled and maximizing its speed. As a result, the system can provide faster and more efficient service to customers.

Our primary objective in this project is to design, implement, and train an intelligent system capable of effectively meeting the customer service requirements of restaurants. Through the utilization of robotics, path planning, ROS, mapping, and localization technologies, we aim to revolutionize the dining experience by reducing waiting times and improving overall service quality.

KEYWORDS: Robot, Path Planning, ROS,MAPING , Localization.

الملخص:

خلال ساعات الذروة ، غالبًا ما تكافح المطاعم الكبيرة والشهيرة لتقديم خدمة مرضية لعملائها ، مما يؤدي إلى فترات انتظار طويلة للطلبات والتسليم بسبب الاكتظاظ. لمواجهة هذا التحدي ، قمنا بتطوير مشروع متخصص يهدف إلى تعزيز خدمة العملاء وتلبية الاحتياجات الفريدة للمطاعم. يتضمن حلنا دمج روبوت وصفحة ويب مخصصة ، والعمل معًا لتحسين العمليات داخل المطعم.

في قلب نظامنا توجد خوارزمية متقدمة تعتمد على نظام تشغيل الروبوت (ROS). تمكن هذه الخوارزمية الروبوت من تحديد المسار الأكثر كفاءة بذكاء لتلبية طلبات العملاء. من خلال التفكير بعناية في عوامل مثل رسم الخرائط والتوطين ، يمكن للروبوت تحديد المسارات المثلى ، وتقليل المسافة المقطوعة وزيادة سرعتها إلى الحد الأقصى. نتيجة لذلك ، يمكن للنظام تقديم خدمة أسرع وأكثر كفاءة للعملاء.

هدفنا الأساسي في هذا المشروع هو تصميم وتنفيذ وتدريب نظام ذكي قادر على تلبية متطلبات خدمة العملاء في المطاعم بشكل فعال. من خلال استخدام الروبوتات ، وتخطيط المسار ، وتقنيات ROS ، ورسم الخرائط ، والتوطين ، نهدف إلى إحداث ثورة في تجربة تناول الطعام من خلال تقليل أوقات الانتظار وتحسين جودة الخدمة الشاملة.

الكلمات الرئيسية: الروبوت ، تخطيط المسار ، MAPPING ، ROS ، الترجمة.

Table of Contents

List of Acronyms	viii
1 Introduction	1
1.1 Overview	1
1.2 Aims and objectives	2
1.3 Problem Statement	2
1.4 Project Requirements	2
1.4.1 Functional Requirement	3
1.4.2 Nonfunctional Requirement	3
1.5 System Description	3
1.6 Project Limitations	4
1.7 Project Schedule	4
1.8 Report outline	5
2 Theoretical Background	6
2.1 Preface	6
2.2 Theories	7
2.2.1 ROS (Robot Operating System)	7
2.2.2 Simultaneous localization and mapping	9
2.2.3 Localization and Navigation	10
2.3 Literature Review	11
2.4 Summary	14
3 System Design	16

3.1	Preface	16
3.2	The system components and design option	16
3.2.1	Hardware Elements and Possibilities	17
3.2.2	System software components	27
3.3	Block Diagram	31
3.4	Flow Charts	33
3.5	Sequence Diagram	34
3.6	Schematic Diagram	35
3.7	Summary	35
4	Implementation	37
4.1	Overview	37
4.2	Hardware Implementation	37
4.2.1	Ultrasonic and Buzzer	37
4.2.2	Hardware elements and possibilities	38
4.3	Software Implementation	38
4.3.1	Operating System	39
4.3.2	Installing needed packages	39
4.3.3	linking	39
4.3.4	ROS implementation	39
4.3.5	ROS algorithms	40
4.4	Web site Implementation	43
4.4.1	User Interface	43
4.5	Implementation Issues	47
5	Validation and Results	49
5.1	Overview	49
5.2	Hardware Testing	49
5.2.1	start_ros	49
5.2.2	test_lieder	51
5.2.3	gmapping_basic	51

5.2.4	navigation	52
5.3	Software Testing	52
5.3.1	Mobile Robot MP-400	53
5.3.2	website	54
5.4	System Validation	55
6	Conclusion	56
6.1	Conclusion	56
6.2	Future work	56
	references	58

List of Figures

2.1	ROS Environment	8
2.2	Robot Serves	12
2.3	AUTOMATED WAITER ROBOT	13
3.1	ESP32 Core board	18
3.2	Arduino UNO	19
3.3	IR sensor	21
3.4	Ultrasonic sensor	22
3.5	URG-04LX-G01	23
3.6	mobile robot MP-400	26
3.7	block digram mobile robot MP-400	27
3.8	Block Diagram	31
3.9	Flow Charts to website	33
3.10	Flow Charts to robot	34
3.11	Sequence Diagram	35
3.12	Schematic Diagram	35
4.1	control hand of Mobile Robot	38
4.2	control hand of Mobile Robot	38
4.3	access point	39
4.4	map of raviz	41
4.5	move_base	43
4.6	orders customers	44
4.7	orders customers	45

4.8	orders customers	45
4.9	interface for the page of the second site	46
4.10	logo table	47
5.1	robot master in bashrc	50
5.2	rostopic list in Robot	50
5.3	start_ros	51
5.4	test_lieder	51
5.5	test_gmapping	52
5.6	test_navigation	53
5.7	test of mp-400	53
5.8	ordering process	54
5.9	information in database	55

List of Tables

1.1:schedule time.....	4
2.1 comparison of two project that are comparable to ours.....	20
3.1:difference between Arduino and ESP32 Core board.....	25
3.2:difference between sensors.....	30
3.3:difference between robot.....	31
3.4:difference between language software.....	34
3.5:comparison between c,c++,python,and java.....	36

List of Acronyms

ROS	Robot Operating System
SLAM	Simultaneous Localization and Mapping

Chapter 1

Introduction

1.1 Overview

Robot jobs in our world are increasing day by day and occupy an important place in our daily life. Robots have been used in medicine, surgery, engineering, architecture, aviation, industry and housekeeping; in this project the robot will be used in a restaurant, you order food through a screen or an app, and the robot comes to you with your order.

In our concept, the kitchen of the restaurant already has a ready-made selection of food and drinks for the customers; as a result, the robot will fetch food from the kitchen and deliver it to the customers' tables. The robot will use scheduling to determine its best course of action, allowing restaurants to attract customers who have a specific interest. (In a recent review of restaurant usage across the globe.

1.2 Aims and objectives

The objectives of the project are:

1. Use the Robot for serving food to customers.
2. Make a website for the restaurant that showcases the specialty foods and beverages that the establishment offers.
3. Manually providing the robot (through the manager) with the coordinates of the tables at their specified locations within the restaurant.
4. Generate map for the searching place.

1.3 Problem Statement

Robotics's Uses In today's world, robots are being used more and more. Numerous Android applications have made everyday life simpler and more productive.

Customers face many challenges in restaurants and hotels due to rush hour crowding, a lack of waiting for staff, and the manual processing of orders. Intelligent robots, meal ordering systems, and restaurant automation systems can be used as alternatives for the waiter to manage these challenges In this project, a robot will be used in the restaurant. You order food through a Web site , and after it arrives in the kitchen, your order is handed to a bot, who then delivers it to you.

1.4 Project Requirements

Some of the functional and non-functional requirements for our system are:

1.4.1 Functional Requirement

1. The robot must be able to create a map of the area.
2. The robot must be able to navigate space safely by avoiding obstacles.
3. The robot must be able to deliver the food order to the specified location (customers) in the best possible of his choice..
4. The web page should be developed to identify, receive and process orders and coordinate with the chef and the robot

1.4.2 Nonfunctional Requirement

1. The system should be user-friendly and easy to use.
2. The system must be Safe (things that are delivered to customers must not be damaged).

1.5 System Description

- our project consists of two phases, as follows: In the first stage, the request is taken from the customer in two ways:

Method 1: it is taken with a QR code located on the restaurant's countertop, which takes you to the Web site .

Method 2: The order is submitted through the application to the restaurant.

Note: We have selected the first approach the QR code in the tabel.

1.8 Report outline

The broad concept of a project and the problem that the project is intended to solve are also covered in this chapter, along with functional and non-functional needs and probable project hurdles. a strategy for the future that will enhance the task, as well as the theoretical underpinning required for the robot to accomplish the goals.

Chapter 2

Theoretical Background

The project in this section will be discussed in terms of the theories we want to use, as well as related and related projects that have been completed in the past. In addition, we will define the concept of each project, compare it with our project, and discuss how to expand our project on previous projects and support the basics of old projects in our project.

2.1 Preface

We are planning to build a robot for restaurants that takes food from the kitchen and brings it to the customers' tables because the restaurant's kitchen already has a ready set of food and drinks for the customers. By scheduling orders, the robot determines the optimal course of action, helping restaurants attract customers with specialized interests. The use of restaurants has recently undergone a re-evaluation, and in this project, a robot will be used. You can order food through a website, and once it reaches the kitchen, it will be delivered to you by a robot, which will also serve it to you.

The goal of the robot in our project is to take food from the kitchen and send it to customers' tables. This project is necessary to increase the efficiency of customer service[2]. Reduces waiting time during peak hours. The robot uses a microcontroller to work within the parameters of distributing specific food to specific customers, obtaining information from sensors, and providing a faster reaction. Having a robot in a restaurant will perform some of the tasks that humans do. This will significantly reduce the need to employ human resources, and this will be reflected in increased profits.

When customers sit at the restaurant table, the food is ordered through the restaurant's web page via the QR Code track on the restaurant table, and the requested food request is sent to the kitchen to prepare the order. When the food is ready from the kitchen, the robot's task now begins, which is to take the order and hand it over to customers. But before that, the robot locates it and knows its place in the room by the mapping algorithm and determines the number of tables in the room. room by algorithms and sensors.

When the robot loads customer orders, it schedules the orders and calculates using algorithms to choose the best path to follow the distribution of the orders it holds, knowing that the robot does not load all the orders together at the same time. Alternatively, some add to the robot's ability to avoid obstacles on its own, such as people and a wall.

2.2 Theories

2.2.1 ROS (Robot Operating System)

This section provides an overview of ROS, the reasons we picked it, and methodology.

- Overview

As a platform for software development, ROS is open-source and offers libraries and tools to assist programmers in building robotics applications. Hardware abstractions, device drivers, libraries, visualizers, message-passing, package management, and other features are available. 'Robot frameworks' and ROS share several similarities[3].

- **Methodology**

The tools and code in ROS enable you to execute your project code and complete the necessary tasks. Every node in the loosely linked system known as ROS is a process, and each node is expected to do a single job.

By delivering messages over logical channels known as topics, nodes exchange information with one another. The publish/subscribe approach allows each node to broadcast or receive data from the other nodes. Packages are how ROS organizes its software. A package might include ROS nodes, a library that isn't reliant on ROS, a dataset, configuration files, third-party software, or anything else that logically fits into the category of a useful module. These packages aim to offer this helpful functionality in a simple-to-use way so that software may be reused with ease. Generally speaking, ROS packages adhere to the "Goldilocks" principle: just the right amount of functionality to be practical but not so much that it makes the package bulky and challenging to integrate with other pieces of software[4].

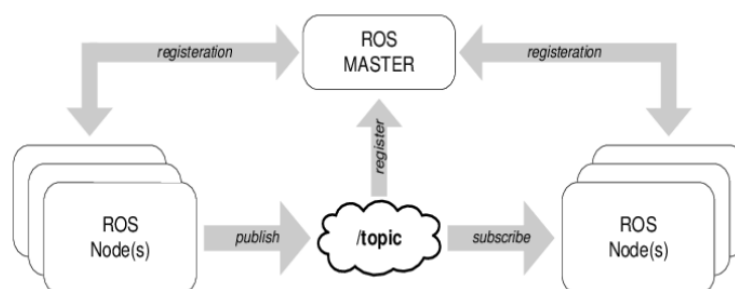


Figure 2.1: ROS Environment

[5]

ROS has multiple versions, currently there are two main versions that are supported[6]:

ROS Noetic Ninjemys has released in May 23rd, 2020.

ROS Melodic Morenia was released on May 23rd, 2018. The tools that are used in ROS, Rviz is a 3D visualization software application for robotics, sensors, and algorithms. Rviz is short for "ROS visualization." It allows you to observe how the robot sees the outside environment (real or simulated). Rviz's goal is to make it possible for you to view a robot's status. It makes an effort to accurately represent what is happening in the robot's surroundings using sensor data. A 3D robot simulator is Gazebo. Its goal is to imitate a robot so that you can get a good idea of how your robot will act in a practical setting. It can determine the effects of forces (such as gravity).[7]:

- **The following justifies the usage of ROS in the project:**

1. ROS is a language-neutral system. It makes it simple for a Python node to connect with a C++ node. It encourages reuse and coworking opportunities. Due to the fact that ROS has mostly focused on C++ and Python, it has several libraries that enable users to utilize different languages [8] .
2. ROS offers excellent simulation tools like Rviz and Gazebo that enable robots to run in an unreal environment.
3. It is able to manage several robots.
4. ROS uses a little amount of resources and storage.
5. The ROS project is open-source and has a permissive license.

2.2.2 Simultaneous localization and mapping

A map may be created, updated, and a robot's location can be estimated to map the environment using the mapping approach. You may utilize a variety of ROS programs, including Gmapping.

- **Gmapping**

Path Planing Techniques:

Gmapping is one method that can be used for path planning in robotics. In this approach, the robot constructs a map of its environment using sensors or other means, and then uses this map to plan a path to the desired destination. Gmapping algorithms can be used to generate paths that are optimal in some sense, such as being the shortest or fastest path, or paths that minimize the risk of collision with obstacles. Gmapping algorithms can also be used to plan paths that avoid certain areas or follow specific routes.

- **Mapping Methodology**

To create the map, start by creating the environment and importing Mobile Robot MP-400 into Gazebo. The gmapping package is used to carry out the mapping procedure. The robot model must give odometry data and be fitted with a horizontally placed, fixed, and laser rangefinder in order to utilize gmapping. The Kinect serves as a range finder and uses infrared sensors to acquire a depth image of the surroundings. As a color picture format, it uses 640 x 480 resolution, 16 bits per pixel infrared data. The depth image proc package in ROS is used to transform this depth image into a 3D point cloud [3].

2.2.3 Localization and Navigation

For mobile robots, localization and navigation are the two most crucial responsibilities. We need to be able to build a plan for how to go to a specific destination, thus we need to know where we are. Of fact, these two issues are interrelated rather than separate from

one another. A robot will have trouble getting there if it doesn't know its precise location at the beginning of a planned route.

After several algorithmic approaches to localization, navigation, and mapping were put out in the past, probabilistic techniques that reduce uncertainty are being applied to the entire issue complex at once (for example, SLAM, simultaneous localization and mapping).

for the purpose of localization and navigation A particle filter pairing algorithm will be used.[9]

2.3 Literature Review

This section will present a selection of earlier project that are comparable to ours and compare them.

1. Robot Serves Restaurant.Food in Restaurant

[10]

Author said (Mays Nassar):

In today's world, the use of the robot is going on increasing. Robots are able to carry out every work more effectively and efficiently than a human can do. Hence one of such application of robot could be the autonomous serving robot. There are many areas of research that could be done for a serving robot which will take the order and serve the food to the customer.

According to the report from International Federation of Robotics (IFR), "Since 2010, the demand for industrial robots has accelerated at an increasing rate. By 2020 more than 1.7 million new industrial robots will be installed in factories around the world"

.

With increasing population growth in recent times, changing lifestyles, working hours and rest, people are very late to sleep and start their evening after dinner, parents busy and the exit of women to work led to the increasing of eating habits in restaurants. I believe it's good to use the robot in restaurants especially in the recent reevaluation of using restaurants worldwide.[10]

In my project, I will build a robot that will bring food from the main counter to the customer's table in the restaurant. It gives the unique idea to give a boost to the business. It will help restaurants to fetch costumers' attention especially in the recent revaluation of using restaurants worldwide..show figure 2.11

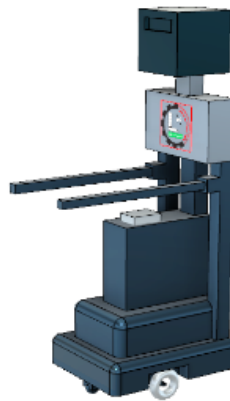


Figure 2.2: Robot Serves

[10]

2. AUTOMATED WAITER ROBOT WITH SMART ORDERING SYSTEM

[11]

International Research Journal of Modernization in Engineering Technology and Science I Volume:03/Issue:12/December-2021

several uses in robotics, including helping the old, sick, and injured. These robots may be employed in many ways depending on the circumstance since they are versatile .

In traditional restaurants, a real waiter takes orders and delivers the meal when it is ready. The consumer then pays the waiter or the payment counter for the ordered items.

For addressing clients, managing food orders, setting the request on the table, and assisting requests in remembering clients, this framework requires a lot of effort. Recent years have seen been a lot of focus on how to successfully improve customer service quality using technical advancements. One profession that individuals dislike is working as a waiter at a restaurant. Therefore, there is a propensity for restaurants to automate with robots. The difficulties of keeping people who are eager to start careers in the food and beverage business, as well as the high cost of recruiting, are frequently cited as causes.

Automating restaurants with waiter robots and intelligent ordering systems is the suggested answer to the aforementioned issue. This method will significantly lessen the need for labor and unskilled labor and will also give local and small food chains a full built-in system..show figure 2.12

There are two key components to this system. They are the robot and the mechanism for ordering meals (Navigation Unit). A web application serves as the foundation for the meal ordering system. The digital tablet (user interface) that is on the table may be used by the consumer to place their purchase.

Wi-Fi is used to transmit the order to the kitchen and cashier. The robotic waiter studies the line-following phenomena. To locate the appropriate table, the robot scans QR codes. This technique improves the restaurant's effectiveness.[11]

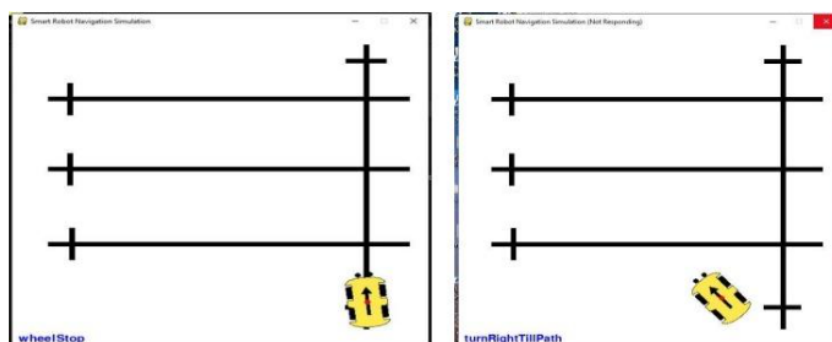


Figure 2.3: AUTOMATED WAITER ROBOT

Where the table shows a comparison between previous projects and our project

Table 2.1: Comparison of two projects that are comparable to ours

	Release year	USED microcontroller	USED sensors	Used algorithm	food delivery	Prepared by
Robot Serves Food in Restaurant.[1]	2017	PIC18F452	IR sensor	No algorithm	Line Follower	Mays Nassar
AUTOMATED WAITER ROBOT WITH SMART ORDERING SYSTEM [2]	2021	Raspberry pi	IR sensors	No algorithm	Line Follower	
Waiter Robot	2023	embedded systems	URG-04LX-UG01	yes algorithm	mapping	

2.4 Summary

Before taking the order and delivering the food to the customers, the robot chooses its location in the room and calculates the number of tables in the room using algorithms and sensors. When customers are seated at the restaurant table, the food is ordered through the restaurant's web page on the QR CODE path, and the ordered food order is sent to the kitchen for preparation. When the bot loads requests from clients, it schedules the requests and uses algorithms to determine the optimal path to take in order to keep track of the distribution of the requests it holds, even when it isn't loading all the requests at once. In

turn, some are working to improve the autonomy of the robot in avoiding obstacles such as humans and walls.

Chapter 3

System Design

This chapter gives a description of the system, detailed design, and important information about the design.

3.1 Preface

The project's methodology and implementation will be covered in this chapter, along with an explanation of the system's abstract block diagram. The comprehensive design of each component, including its schematic diagram, will be provided after the detailed explanation of the system. Finally, we'll describe the schematic of the system hardware components.

3.2 The system components and design option

This section explains the components of our project and why we picked them.

3.2.1 Hardware Elements and Possibilities

1. MicroController

To process the data that we will provide to a robot, we need a microcontroller.

One of the project's important components is the microcontroller. The type of processor that is chosen relies on a number of factors, including:

1. Cost.
2. memory size.
3. Speed and clock speed.
4. The quantity of I/O pins.

- **ESP32 Core board:** is a small development board built around an ESP-WROOM-32 module. From the I/O, this development board leads out 2.54mm-pitch headers to both sides. Users can connect peripheral devices based on their specific requirements. Standard headers on both sides can help you operate in a clearer and more practical manner when utilizing and debugging.

The ESP-WROOM-32 module, which has fewer than 10 external components, is the best Wi-Fi and Bluetooth solution. Filters, power amplifying devices, low noise amplifying devices, RF baluns, antenna switches, and power management modules are all included. Additionally, it utilizes TSMC's 40nm low-power technology. Radio frequency technology has an excellent economic performance, making it risk-free and dependable for expanding a variety of applications.

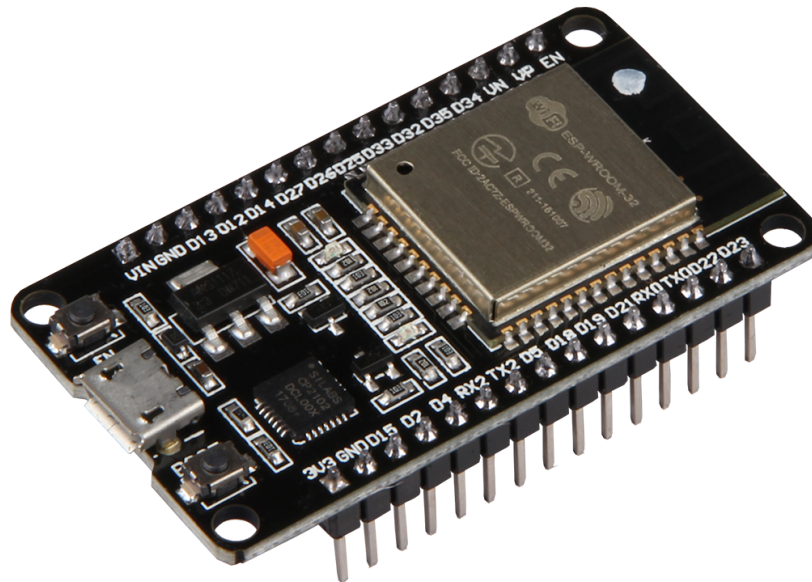


Figure 3.1: ESP32 Core board

ESP-NOW One of the main features of ESP8266[22] is ESP-NOW, a protocol developed by Espressif that enables multiple devices to communicate with one another without using Wi-Fi. The protocol is a lot like the 2.4 GHz low-power wireless connectivity that is frequently used in wireless mice. So, the pairing of devices is needed before their communication. After the pairing is done, the connection is secure and peer-to-peer, with no handshake being required.

Arduino:

An open-source electronics platform called Arduino is built on simple hardware and software. A motor can be started, an LED can be turned on, and anything may be published online by using an Arduino board to receive inputs like light on a sensor, a

finger on a button, or a tweet. Sending a set of instructions to the board's microcontroller will instruct your board what to do. You achieve this by using the Arduino Software (IDE), which is based on Processing, and the Wiring-based Arduino Programming Language[12].

Over the years, countless of projects, ranging from simple household items to intricate scientific equipment, have used Arduino as their brain. This open-source platform has attracted a global community of makers, including students, hobbyists, artists, programmers, and professionals. Their efforts have added up to an astounding quantity of accessible information that may be very helpful to both beginners and specialists.

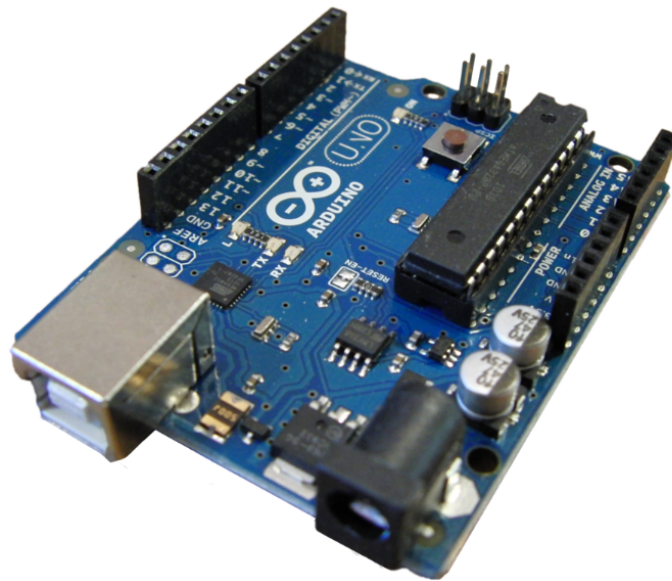


Figure 3.2: Arduino UNO
[13]

At the Ivrea Interaction Design Institute, Arduino was created as a simple tool for quick prototyping geared for students with no prior experience in electronics or programming. The Arduino board began altering as soon as it gained a larger audience, distinguishing its offerings from basic 8-bit boards to items for Internet of Things applications[12].

We examined and contrasted all of the potential choices before selecting the best one for our project. which are displayed in Table 3.1

Table 3.1: difference between arduino and ESP32 Core board .

<i>Feature</i>	<i>ESP32</i>	<i>Arduino Uno</i>
<i>Microcontroller</i>	Xtensa LX6, 32-bit	ATmega328P, 8-bit
<i>Clock Speed</i>	160 or 240 MHz	16 MHz
<i>Performance</i>	Higher processing power	Limited processing power
<i>Memory</i>	More program memory and RAM	Limited program memory and RAM
<i>Wireless Connectivity</i>	Built-in Wi-Fi and Bluetooth	No built-in wireless connectivity
<i>Digital I/O Pins</i>	More available	Limited number
<i>Analog Input Pins</i>	More available	Limited number
<i>Voltage</i>	3.3V	5V
<i>Development Environment</i>	Arduino IDE, PlatformIO	Arduino IDE, PlatformIO

After looking for different types of controllers, the ESP32 Core board is superior in many aspects such as: wifi and bluetooth, but our project does not need much wireless connection, so we chose Arduino Uno instead because it is the cheapest and fastest option available on the local market. If you're interested in learning more about this issue, see this more

technical explanation in Table 3.1:

2.sensor

- **IR distance sensor :**

An IR distance sensor, also known as an infrared distance sensor, is a device used to measure the distance between the sensor and an object without making physical contact. It utilizes infrared light to perform distance measurements. These sensors are widely used in various applications, including robotics, automation, industrial control, and consumer electronics.

IR distance sensors come in different types, such as analog output sensors, digital output sensors, and Time-of-Flight (ToF) sensors. Analog output sensors provide a continuous range of values proportional to the distance, while digital output sensors typically provide a binary output indicating whether an object is within a certain distance threshold or not. ToF sensors offer higher accuracy and can be used for more precise distance measurements.

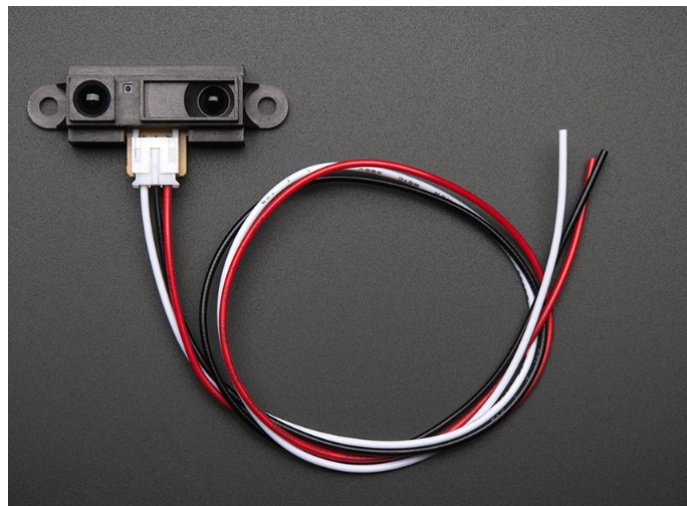


Figure 3.3: IR sensor

These sensors have various applications, including object detection, obstacle avoid-

ance, surface detection, and distance measurement in robotics and automation systems. They are relatively affordable and straightforward to use, making them popular choices in many electronic projects and commercial applications.

- **Ultrasonic sensor:**

Ultrasonic sensors are electronic devices that use the emission of ultrasonic sound waves to determine a target's distance before converting those waves into electrical signals. The speed of traveling ultrasonic waves is greater than the speed of audible sound. The transmitter and receiver are the two primary fundamental components. The transmitter produces sound using piezoelectric crystals, which it then sends to the target before returning to the receiving component. Ultrasonic sensor is shown in figure 3.5 Similar to sonar and radar, ultrasonic sensors analyze the characteristics of a target or item by deciphering the received echoes from sound or radio waves, respectively. These sensors generate high-frequency sound waves and examine the echoes they receive. The target's distance is determined by the sensors by timing the transmission and reception of echoes[14].

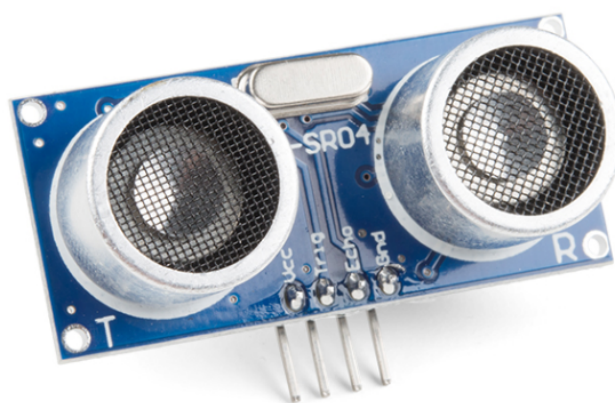


Figure 3.4: Ultrasonic sensor

URG-04LX-G01: The -04LX-UG01 is a straightforward, affordable 2D LiDAR sensor that is perfect for robotic applications in academia and RD start-ups. This device can communicate and receive power over the USB interface, and it can also gather measurement data up to a distance of 4 meters.

OPERATION BENEFITS:

1. indoor / outdoor - indoor
2. Range - 4 m
3. Scan window / angle - 240 degrees
4. Communication - USB2.0 [Mini B] (Full Speed)
5. Dimensions (L x W x H) - 50mm x 50mm x 70mm
6. Weight - 160 grams



Figure 3.5: URG-04LX-G01
[16]

It provides a quick, less accurate method for localizing autonomous robots and automated material handling systems in a given space. The URG-04LX-UG01 is a straightforward[17], affordable 2D LiDAR sensor that is perfect for robotic applications in academia and RD start-ups. Figure 3.6 shows URG-04 sensor.

This device can communicate and receive power over the USB interface, and it can also gather measurement data up to a distance of 4 meters. It provides a quick, less accurate method for localizing autonomous robots and automated material handling systems in a given space.

Table 3.2: difference between IR , URG-04LX-UG01 AND Ultrasonic sensor

Requirements	IR sensor	Ultrasonic sensor	LIDAR sensor
			
Rang	An infrared sensor (IR sensor) is an optoelectronic component that is radiation-sensitive and has a spectral sensitivity in the infrared wavelength range of 780 nm to 50 m.	The most used frequency range for ultrasonic sensing is 40 to 70 kHz. The lower the frequency, the greater the sensing range; frequency determines range and resolution. The measurement range is up to 11 meters and the resolution is one centimeter (cm) at 58 kHz, a widely used frequency.	the Hokuyo laser rangefinder entry-level model. 5.6 meters. ideal for detecting obstacles
Frequency	353thz	40 KZ	200THZ
Cost	7\$	10\$	400\$-1000\$
Working voltage	3.3/5v	5v	
Accuracy	Low	Low	high



We checked and compared all possible options before choosing the best one for our project. which are shown in Table 3.2 We chose the URG-04LX-UG01 LIDAR, which is

the most expensive of the ultrasonic and infrared sensors, but it is better at detecting small objects and suitable for detecting moving objects because it works as a 270-degree scan, so we only need one sensor. To protect the robot in case of failure of the LIDAR sensor, we additionally chose the ultrasonic sensor on the four sides of the robot (front, back, etc.).

3. Robot

The project must be designed by a robot, which will also connect and communicate with the various components. IN Table 3.3

Table 3.3: difference between TURTLEBOT 2 AND neobotix mp-400 Robot

Requirements	TURTLEBOT 2	neobotix mp-400
		
WEIGHT	6.3 kg	80 kg
Cost	\$1,049 USD	2500—3000\$
Base	Kobuki	-----
Battery	2200 mAh Li-Po Battery	AGM Batteries and LiFePO4
technology transfer	Hardware and Software and Open Source	Hardware and Software and Open Source
Charger	Fast Charger	Fast Charger
Use	Easy	Hard
Mounting Hardware	yes	yes
Speed	0.65 m/s	1.5 m/s
Size	590 x 559 x 411	590 x 559 x 411 (mm)

Although the Neobotix mp-400 Robot is a newer version, the Turlebot 2 and Neobotix are identical, so we chose the Neobotix mp-400 because our project required the robot to have a lot of space, and to be able to carry it around. Dishes (more than one plate of food), and are fast-moving. When it comes to working indoors, the MP-400 is remarkably unique.

- Mobile Mobile Robot MP-400:

The mobile robot MP-400 is an autonomous robot vehicle for a variety of uses. It also includes all variations and versions based on it. Where it its two large drive wheels and central differential drive allow for precise, efficient runs over longer distances and excellent maneuverability on a variety of terrain. The laser scanner's measurement data may be applied to localization, navigation, and collision avoidance. Additionally, the user-defined safety fields in front of the robot are monitored by this scanner. The robot is instantly put on emergency stop the moment an object is found inside the field that is currently operating[18].



Figure 3.6: mobile robot MP-400

[18]

With the use of these abilities, the robot can precisely detect where it is in relation to the working space, plan routes to arbitrary locations, and securely avoid colliding with pedestrians or other dynamic barriers.

The robot may accommodate the mounting and integration of additional parts and systems. They may be run off the robot's internal power source and managed by the on-board computer.

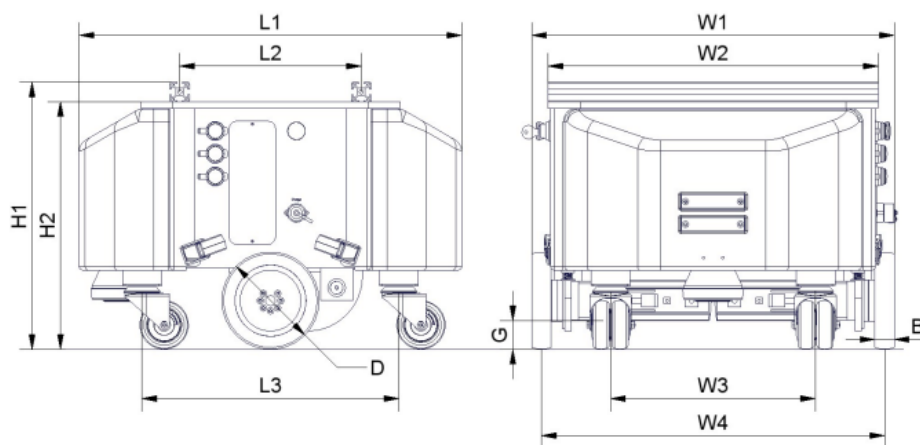


Figure 3.7: block digram mobile robot MP-400

[18]

3.2.2 System software components

The most appropriate linguistic frameworks for the production and acquisition of project components.

In the program section, the project's highest accuracy, outcomes, and performance were sought after. After looking for types and selecting the finest of them, we made the decision to use web developer as the language for developing a restaurant website as well as Python System programming language and link components.

Table 3.4: difference between Language software

<i>web Language</i>	<i>FLUTTER</i>	<i>Java</i>	<i>REACT NATIVE</i>
<i>User interface</i>	Easy-to-use interface	Flexibly extensible, and scalable	Impressive graphical user interface
<i>Performance</i>	Removed JS Bridging, enhanced App speed	Fewer bugs	Higher Performance that of native app
<i>Pricing</i>	Open-source platform	Paid updates for JDKcost	Paid updates for the JDKcost
<i>Language Stack</i>	Dart	java works on JVM	java works on JVM
<i>Supported Platforms</i>	Android Jelly Bean, v16,4.1.x and IOS 8+	Android apps	Android 4.0.3+ Versions and Ios 8+

Because of its high performance, usability and suitability for novice programmers, we chose REACT NATIVE as it is our hobby in web site development, as shown in Table 3.4,

- Python Programming Language:

We will use it to program the robot and connect components to each other

One of the most well-liked general-purpose programming languages is Python, an open-source, high-level, dynamically typed language. Compared to other programming languages with built-in data structures, it is faster.

Python has a simple structure that improves readability and lowers the cost of code maintenance and debugging thanks to the combination of dynamic typing and dynamic binding.

Programming in Python is simple, and other languages may rapidly learn Python. Additionally, Python language newbies like the clear syntax. and it's simple to master the indentation structure[19].

Additionally, it was backed by the OpenCV library, which gives researchers' projects access to objects identified by features.

- **C++ Programming Language :**

It is an object-oriented programming language capable of class and object identification. It is a versatile programming language with a wide range of possible uses. Among other things, it can create operating systems, browsers, and games. The programming paradigms it offers are diverse and include functional, procedural, object-oriented, etc. As a result, C++ is reliable and adaptable.

Though a legacy language, C++ is still functioning. Powerful programs and highly competent gaming software are regularly made using it[19]

.





- **Java Programming Language :**

is a well-liked programming language that was introduced in 1995. It is safe, secure, and object-oriented. Oracle owns the Java platform, which is used by more than 3 billion devices[19].

It may be used to create the safest software, including web applications (using the Spring Boot framework), embedded systems, desktop and mobile apps, and huge data processing.

Because Java contains a mechanism called Automatic Garbage Collection, unreferenced objects don't need to be deleted.

Table 3.5: Comparison Between C, C++, Python, and Java

c 	c++ 	java 	python 
Compiled Language	Compiled Programming language	Compiled Programming Language	Interpreted Programming Language
Operator overloading is not supported.	Operator overload is supported.	Operator overloading is not supported.	Operator overload is supported.
A small number of libraries are available.	Has a small number of library patrons	Many concepts, such as UX, are supported by the library.	It includes a sizable library collection that makes it possible to use it for applications in data science, AI, and other fields.
similar to C++ in speed	The programming language C++ compiles quickly.	Compared to the C++ compiler, the Java program compiler is a little slower.	When an interpreter is used, execution is delayed.
Platform-specific	Platform dependent	Platform-unaffected	Platform independent
Syntax rules are strictly followed.	Syntax rules are strictly followed.	Syntax rules are strictly followed.	It isn't necessary to use semicolon ';'.

The above table 3.5 helped us decide that the programming language Python would work best for our project. It will allow us to connect the project's many parts and program them to work together as a working system.

3.3 Block Diagram

The system's connections, how it functions as a unit, and the connections between its components are shown in the following diagram.

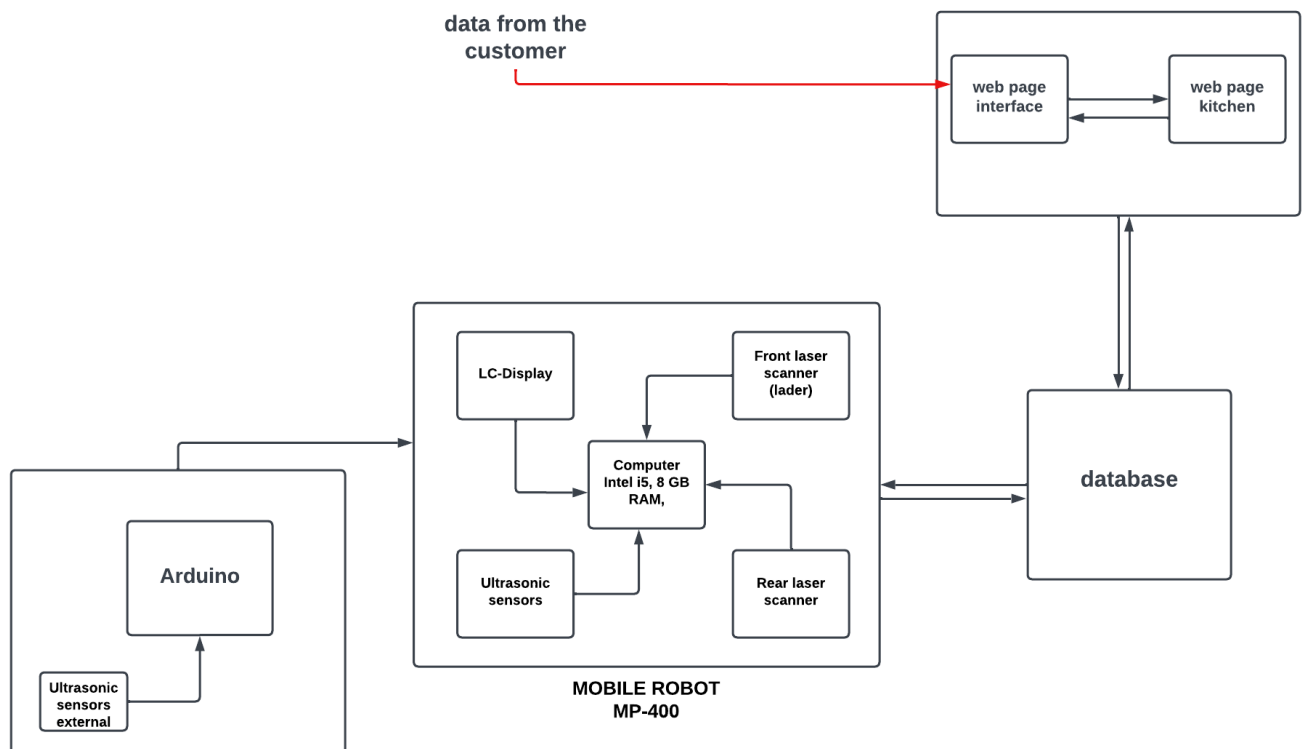


Figure 3.8: Block Diagram

As mentioned earlier, Figure 3.8 provides an overview of the system's general design, which comprises three interconnected systems. The first system is a website consisting of two electronic pages. The first page is dedicated to customers, allowing them to place food orders. This page then sends the orders to the second page, exclusively accessible by the restaurant staff. Customers do not have access to this section.

The second system is the robot itself, composed of interconnected components. It includes an LCD display that shows the robot's status and battery charge level. The robot is equipped with two types of sensors. The front laser scanner (lader) is positioned at the front of the robot, while ultrasonic sensors are present in all directions for protective purposes. These sensors are connected to a computer that functions as the brain of the

robot.

It is important to note that the first and second sections communicate with each other through shared databases.

The final section consists of additional external ultrasonic sensors connected to an Arduino board. These sensors are intended to provide extra protection to the robot in specific scenarios or special cases, for example when the robot approaches a staircase.

3.4 Flow Charts

A flowchart is a diagram that shows how a system, computer algorithm, or process operates. They are commonly used in a variety of disciplines to analyze, arrange, improve, and communicate frequently complex processes in clear, concise diagrams.

The flowchart of the site which is the interface for users to take orders, as shown in figure 3.9.

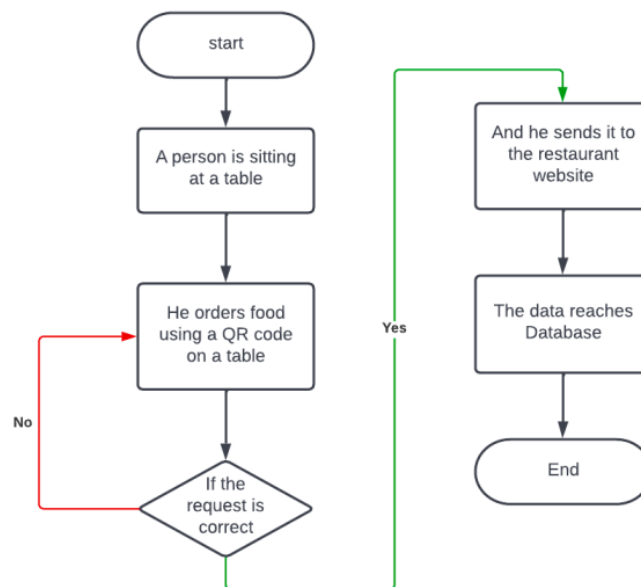


Figure 3.9: Flow Charts to website

Flowchart of the processes that occur in the robot in order to reach the goal.as a showing in figure 3.10.

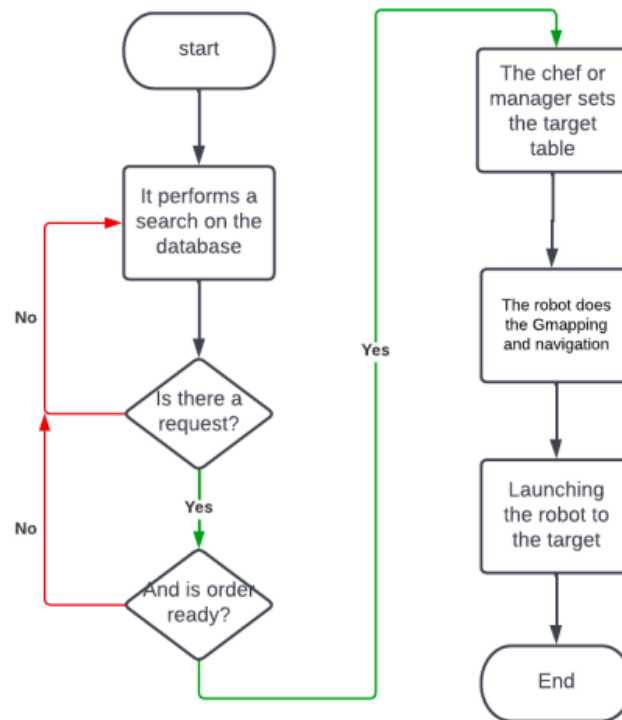


Figure 3.10: Flow Charts to robot

3.5 Sequence Diagram

Interaction diagrams called sequence diagrams display the processes that an operation must go through. They show how diverse objects interact when used in combination. Sequence diagrams, which graphically express the order of interaction by displaying when and how messages are sent, use the figure's vertical axis to represent time[20] .in FIGURE 3.11.

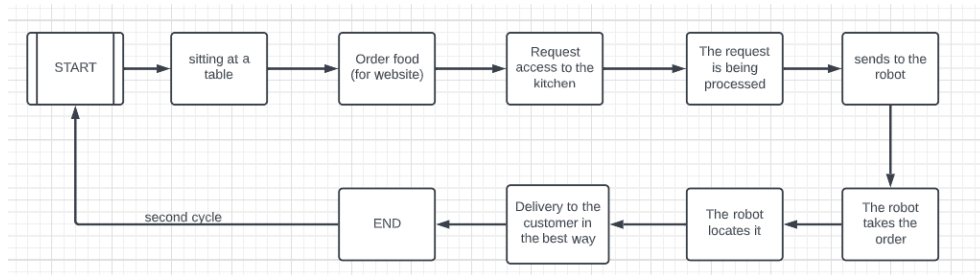


Figure 3.11: Sequence Diagram

3.6 Schematic Diagram

Schematic diagram of the ultrasonic connection with the Arduino Uno in Figure 3.12. We used four pieces of the sensor (ultrasonic sensor) to protect the robot if something happened to the main sensor (URG-04LX-G01), and we placed them in the four main directions (front, back, etc.).

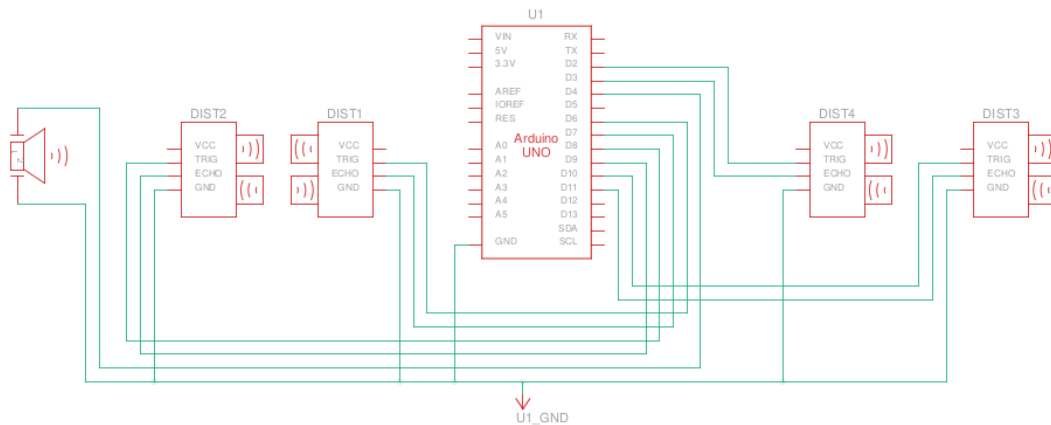


Figure 3.12: Schematic Diagram

3.7 Summary

At the conclusion of Chapter 3, we were able to describe how the device and procedure work. The decision on which portions to use and how to deliver them was made after considering requests from the restaurant. During the construction of this bot.

the following tasks were completed:

1. select the parts classes that are relevant to our project.
2. Drawing up the construction plan for the project.
3. Identify the ideal initial setup for the model circuit connections.

Chapter 4

Implementation

4.1 Overview

The implementation phase for the software and hardware components is explained in this chapter. The project's various hardware and software parts are covered in more detail overall.

4.2 Hardware Implementation

The hardware components used in our project will be described in this section.

4.2.1 Ultrasonic and Buzzer

1. We have gathered all the necessary components, including the Arduino board, ultrasonic sensor, buzzer, wires, breadboard, and power source.
2. After referring to the diagram provided in picture [4.1], we successfully established the necessary connections between the Arduino board, the ultrasonic sensor, and the buzzer. This wiring setup will enable the components to function as intended in our project.
3. We then connected the Arduino board to the robot Mobile Robot MP-400.
4. After ensuring the power supply was turned on, we verified the proper functioning of the Arduino board, ultrasonic sensor, and buzzer.

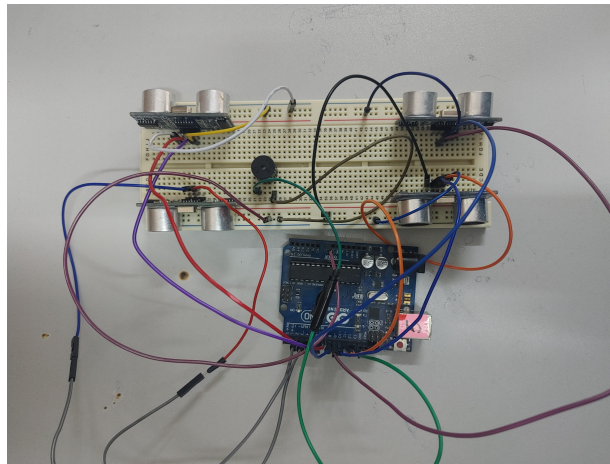


Figure 4.1: control hand of Mobile Robot

4.2.2 Hardware elements and possibilities

The main component is the computer connected to the other system. The ingredients are as follows: 1. We have connected the Mobile Robot MP-400 to a computer at the access point. 2. We use the control hand to move the Mobile Robot MP-400, showing in the figure 4.2.



Figure 4.2: control hand of Mobile Robot

4.3 Software Implementation

This section explains how to integrate a webpage with a robot and establish communication between them, as it describes Machine execution through code.

4.3.1 Operating System

We installed Ubuntu 20.04 OS on computer because the ROS .

4.3.2 Installing needed packages

Raspbian comes with many useful pre-installed packages such as JDK, Python, and others, but we need to install some needed packages for our project, like a ROS implementation.

4.3.3 linking

We create a connection between our computer and the robot using a local network in order to use the robot as a file. We enable remote desktop access through a Wi-Fi access point, as shown in Figure 4.3 the accompanying diagram, which highlights the access point in use.



Figure 4.3: access point

4.3.4 ROS implementation

ROS provides libraries and tools to help software developers create robot applications. It provides hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more. The researchers use Ros1 release Because its compatibility with the Ubuntu 20.04 OS We use the follow the instructions in to install ROS1.

4.3.5 ROS algorithms

The following algorithms are in charge of fully controlling the robot.

Gmapping:

This package contains the ROS wrapper for Gmapping for OpenSlam. The laser-based gmapping package provides SLAM (Synchronous Localization and Mapping), as a ROS node called slam-gmapping. With slam-gmapping you can create a 2D occupancy network map (like a building floor plan) from laser data and an image collected by an animated robot.

The map is drawn by the movement of the robot in a specific location, with distance sensors reading (laser sensors) the distances between the robot and nearby obstacles, and during the drawing the robot every thing around is being discovered.[3]

- **Hardware specifications:**

You require a mobile robot with odometry data and a stationary, horizontally mounted laser range-finder in order to use slam-gmapping. Every incoming scan will be converted into an odom (odometry) tf frame by the slam-gmapping node.

- **How to use Gmapping in Terminal(as commands):**

```
Bring an mp-400 to Create a Map.  
$ roslaunch mp-400_bringup minimal.launch.  
$ roslaunch mp-400_navigation gmapping_demo.launch.  
Launch the rviz program.  
$ roslaunch mp-400_rviz_launchers view_navigation.launch.  
To control the mp-400 by keyboard.  
$ roslaunch mp-400_teleop keyboard_teleop.launch.
```

- **Gmapping result:**

To obtain an image of the Gmapping result, you can follow these steps:

- **Launch** the required ROS nodes and launch files for Gmapping. This typically includes launching the TurtleBot or mp-400 base, as well as the necessary Gmapping launch files. For example:

```
$ roslaunch mp-400_bringup minimal.launch  
$ roslaunch mp-400_navigation gmapping_demo.launch
```

- Open the rviz program to visualize the mapping results:

```
$ roslaunch mp-400_rviz_launchers view_navigation.launch
```

- in rviz, you should see a 2D map visualization. If the map is not visible, you might need to add the "Map" display by clicking on the "Add" button on the bottom-left side of the rviz window and selecting the "Map" option.
- Once the mapping process is running and the map is visible in rviz, you can save an image of the Gmapping result. To do this, click on the "File" menu in rviz, select "Save Image," and choose a location to save the image file.
 - The map resulted using Rviz tool is shown in Figure 4.4



Figure 4.4: map of rviz

map_server

The utilities in this package produce maps, which are then stored in a pair of files. A YAML file in the first file, which also references an image file in the second file, provides the map metadata. The occupancy data of the map is encoded in the image file itself.[21]

move_base

In ROS (Robot Operating System), `move_base` is a commonly used package that provides a flexible and powerful navigation system for autonomous robots. It combines various components such as localization, mapping, and path planning to enable robots to navigate and move towards specific goals in a given environment.[22]

The `move_base` package utilizes the ROS navigation stack, which is a collection of software modules designed for robot navigation. Here's a brief overview of the key components within the `move_base` package:

- **Global Planner:** This component is responsible for long-term planning and generating a global path from the robot's current location to the desired goal. It typically uses a global map representation and a global planning algorithm (e.g., A* or Dijkstra's algorithm) to determine the optimal path.
- **Local Planner:** The local planner focuses on short-term planning and obstacle avoidance. It takes the global path provided by the global planner and generates velocity commands that guide the robot to follow the path while avoiding obstacles in real-time.
- **Costmap:** The costmap represents the occupancy and traversability of the environment. It provides information about obstacles, free space, and areas with varying costs based on their suitability for robot navigation.
- **Sensor Integration:** `move_base` integrates with various sensors, such as laser scanners or depth cameras, to perceive the surrounding environment and update the costmap accordingly. This sensor data is used for obstacle detection and collision avoidance.

By combining these components, `move_base` offers a robust navigation system that enables a robot to autonomously plan and execute its path towards a specified goal while avoiding obstacles along the way, shown in figure 4.5.

To use `move_base` in your ROS system, you need to configure the parameters for the planners, costmap, and sensor integration to suit your robot and environment. Addition-

ally, you should provide the necessary sensor data and goal information to the `move_base` node for it to plan and execute the navigation task.

Please note that `move_base` is a complex package with several configuration options and dependencies. It is often used in conjunction with other ROS packages and tools to achieve a complete robot navigation solution.[22]

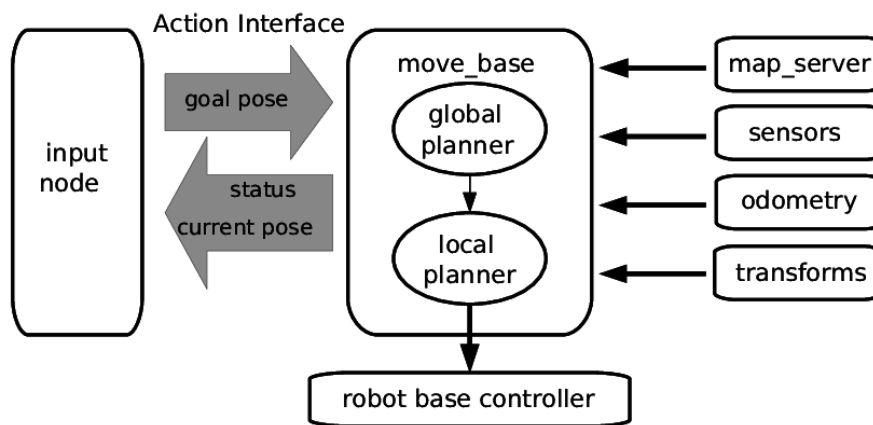


Figure 4.5: `move_base`
[22]

4.4 Web site Implementation

4.4.1 User Interface

In this section, we will provide an overview of the website developed using the React programming language. We will showcase the web pages and provide insights into their functionalities and features.

a basic description of how the website functions:

In the beginning, we have two web pages that we link to each other, which include:

- **Firstly**, the application includes a user interface that is accessible to all individuals. Users can access it by scanning a QR code placed on restaurant tables. Through this

interface, customers may place orders for a variety of dishes.

- **Secondly**, we have developed a dedicated page for the restaurant, which displays the table locations and the orders received from the first page. Through this page, manager of restaurant can send messages to a robot, instructing it to move to a specific location based on their preferences or requirements.

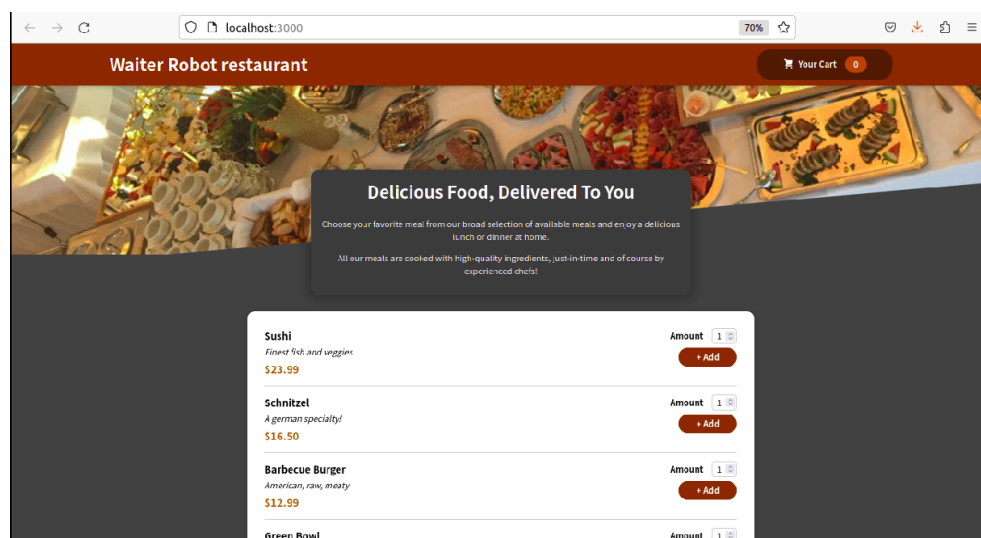


Figure 4.6: orders customers

This page works as a page to take orders from customers, where users can log in to it via the QR code located on the restaurant tables.

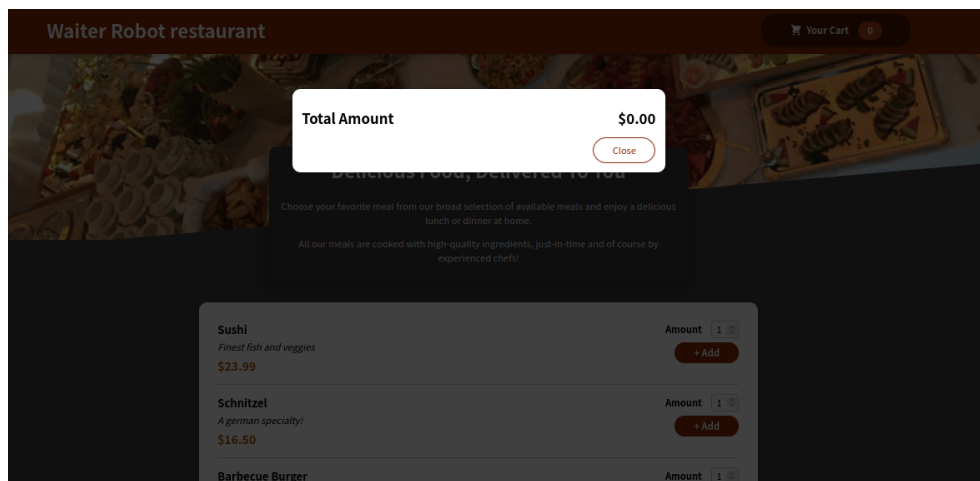


Figure 4.7: orders customers

This page is associated with the page that displays the orders consumers have placed and the cost computation for those orders.

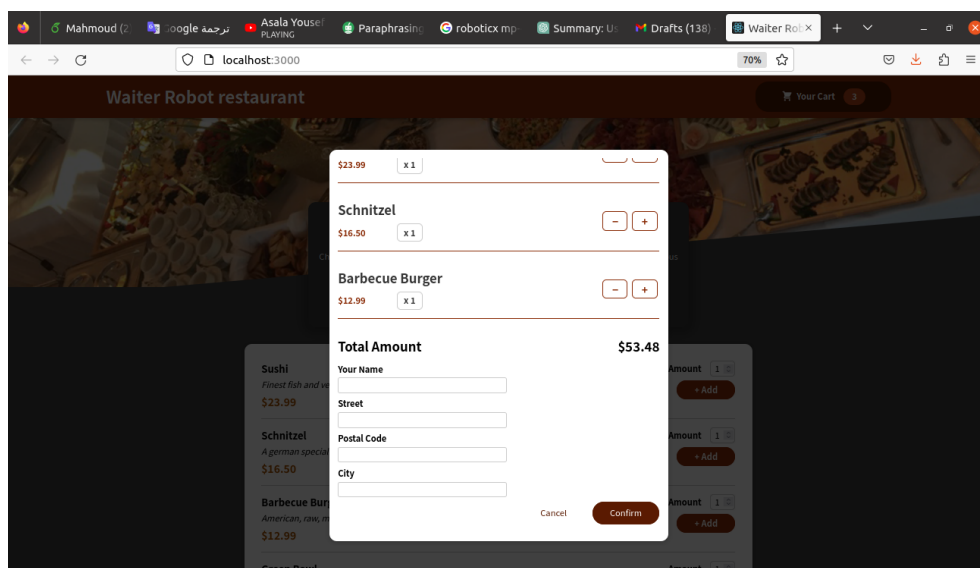
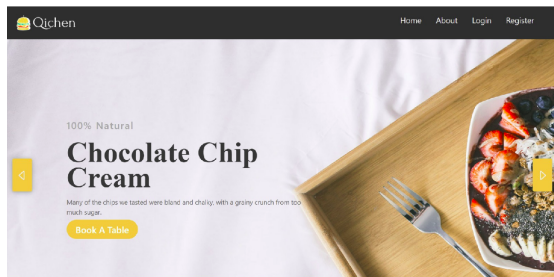
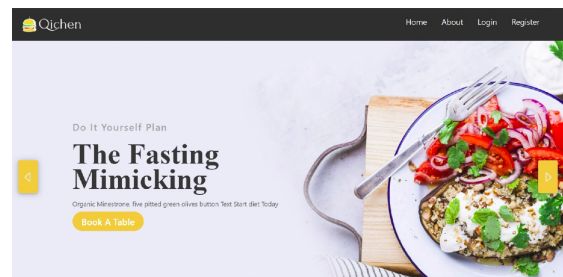


Figure 4.8: orders customers

On this page, the requests that the client requested appear and some intermediate information about him. This is the first page dedicated to all restaurant customers in order to take orders and calculate the cost of the order .



(a) First image



(b) Second image

Figure 4.9: interface for the page of the second site

This page is an interface for the page of the second site dedicated to a restaurant, which no one who is not allowed to enter the site can

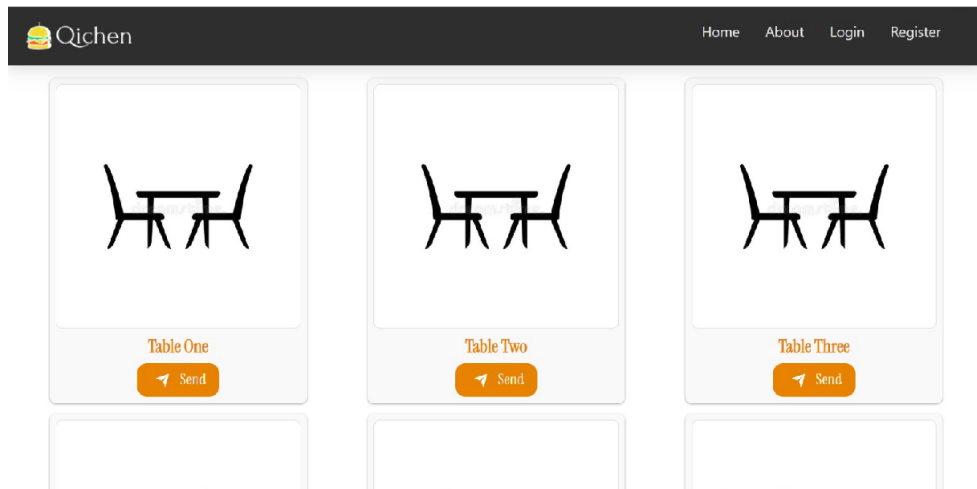


Figure 4.10: logo table

This page, as well as the last page of the second site, which contains a logo for the restaurant's tables, which allows the chef to send the robot to any table he chooses according to the requests prepared.

4.5 Implementation Issues

We encountered several problems when starting the practical part of the project, and we did our best to overcome these problems. Some of these problems include:

1. Installation and Setup: Setting up Ros in a private development environment was somewhat complicated, because they were running on specific versions or on a specific platform. Ensuring that the environment variables are correctly installed and configured correctly is critical.
2. Dealing with the ROS package: Managing dependencies between ROS packages can be challenging, and dealing with numerous sophisticated packages is tough for our device because the device's response is quite poor. To address this issue, we used university devices.
3. Debugging and troubleshooting: Due to the distributed nature of the framework, debugging ROS systems can be challenging. As we lacked the complete concepts to handle them, identifying and fixing problems with node connectivity, message

passing, or setup may call for a methodical approach and a deep comprehension of ROS principles. As an illustration, when we wanted to connect the robot to the restaurant website, we had to comprehend how the robot interacts with the cloud and how to handle it.

4. Real-time limitations: Real-time performance requirements can pose challenges in ROS systems. Achieving deterministic behavior and meeting strict timing constraints can be challenging.

As an illustration, when the robot arrives to the coordinates that it was provided through the website and reverses course, it is doing so since it is technically above the point.

Chapter 5

Validation and Results

5.1 Overview

This chapter illustrates the testing process of the overall system, its components, and software. We test all the parts to ensure that all the functions work as expected and without errors.

5.2 Hardware Testing

In this section, we will do all Testing robot inspections, beginning to end.

5.2.1 start_ros

1. First, we will verify that the robot is connected to the PC via the IP address (controlling the robot and declaring robot master).as a showing in figure 5.1

```

109 # Alias definitions.
100 # You may want to put all your additions into a separate file like
101 # ~/.bash_aliases, instead of adding them here directly.
102 # See /usr/share/doc/bash-doc/examples in the bash-doc package.
103
104 if [ -f ~/.bash_aliases ]; then
105     . ~/.bash_aliases
106 fi
107
108 # enable programmable completion features (you don't need to enable
109 # this, if it's already enabled in /etc/bash.bashrc and /etc/profile
110 # sources /etc/bash.bashrc).
111 if ! shopt -oq posix; then
112     if [ -f /usr/share/bash-completion/bash_completion ]; then
113         . /usr/share/bash-completion/bash_completion
114     elif [ -f /etc/bash_completion ]; then
115         . /etc/bash_completion
116     fi
117 fi
118 source /opt/ros/noetic/setup.bash
119 # source /home/ppu/calvin_ws/devel/setup.bash
120 source /home/ppu/ros_work/devel/setup.bash
121
122 export ROS_MASTER_URI=http://192.168.0.52:11311
123 export ROS_IP=192.168.0.100
124 export ROS_HOSTNAME=192.168.0.100
125
126
127
128
129
130

```

Figure 5.1: robot master in bashrc

We used the command **rostopic list** if showed appear in figure 5.2 to check the robot's functionality.

```

dauker@dauker-asus:~$ rostopic list
/cambot/joint_states
/camera/rgb/camera_info
/camera/rgb/image_raw
/camera/rgb/image_raw/compressed
/camera/rgb/image_raw/compressed/parameter_descriptions
/camera/rgb/image_raw/compressed/parameter_updates
/camera/rgb/image_raw/compressedDepth
/camera/rgb/image_raw/compressedDepth/parameter_descriptions
/camera/rgb/image_raw/compressedDepth/parameter_updates
/camera/rgb/image_raw/theora
/camera/rgb/image_raw/theora/parameter_descriptions
/camera/rgb/image_raw/theora/parameter_updates
/camera/rgb/parameter_descriptions
/camera/rgb/parameter_updates
/clicked_point
/clock
/cmd_vel
/gazebo/link_states
/gazebo/model_states
/gazebo/parameter_descriptions
/gazebo/parameter_updates
/gazebo/set_link_state
/gazebo/set_model_state
/initialpose
/joint_states
/move_base_simple/goal
/odom
/rosout
/rosout_agg
/scan
/simple_controller/absolute_position
/tf
/tf_static

```

Figure 5.2: rostopic list in Robot

When the **rostopic list** menu appears, we have contacted the robot.

2. Second, we enter the robot device for a program called VNS {it is aremotr desktop applicahion} to confirm the robot start_ros. as a show in figer 5.3

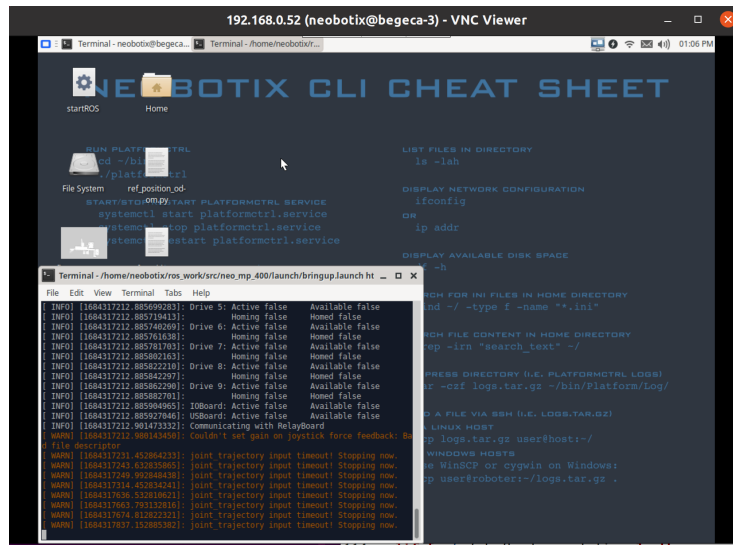
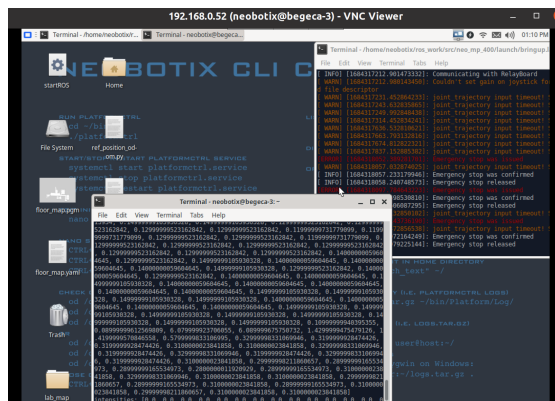


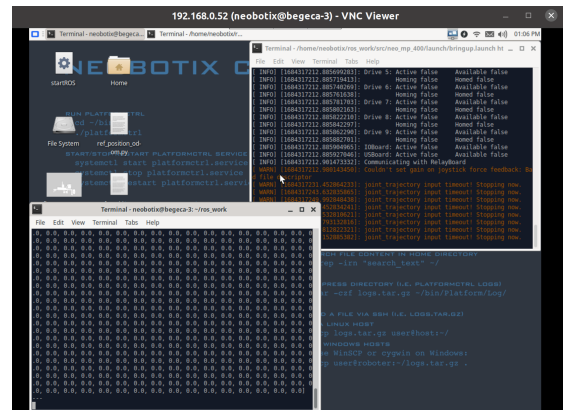
Figure 5.3: start_ros

5.2.2 test_lieder

At this point, we will test the cover sensor in an open place without any obstacles in front of it. And in the second case, that is with obstacles, as a showing figure 5.4 .



(a) test_lieder If a wall is in his path



(b) test_lieder if there is no firewall in its path

Figure 5.4: test_lieder

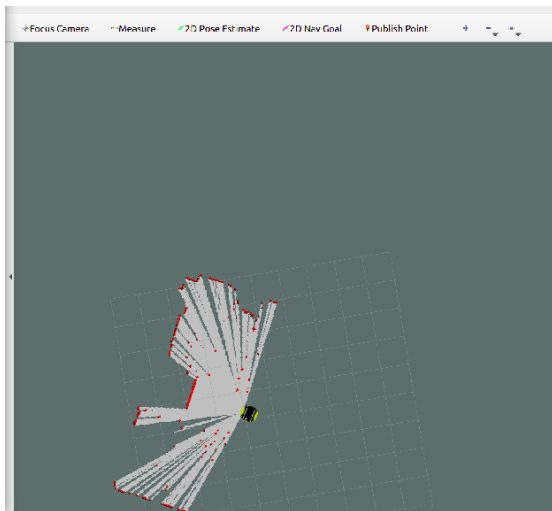
5.2.3 gmapping_basic

We need to concentrate on a few factors when testing GMapping in order to create a fantastic map. For instance, we used the leader sensor for that as we were aware of the

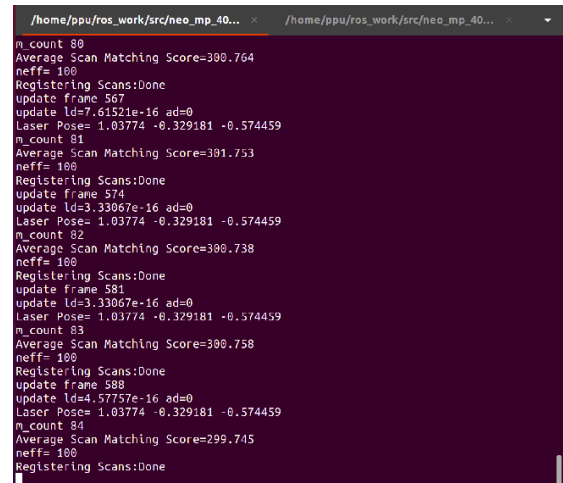
type of sensor that handles scanning. However, a thorough examination of the robot's surroundings is required, good for an excellent map.as a show in Figure 5.5

we used the command in cmd to gmapping :

```
$ roslaunch neo_mp_400 gmapping_basic.launch
```



(a) gmapping in Rviz



(b) gmapping in cmd

Figure 5.5: test_gmapping

5.2.4 navigation

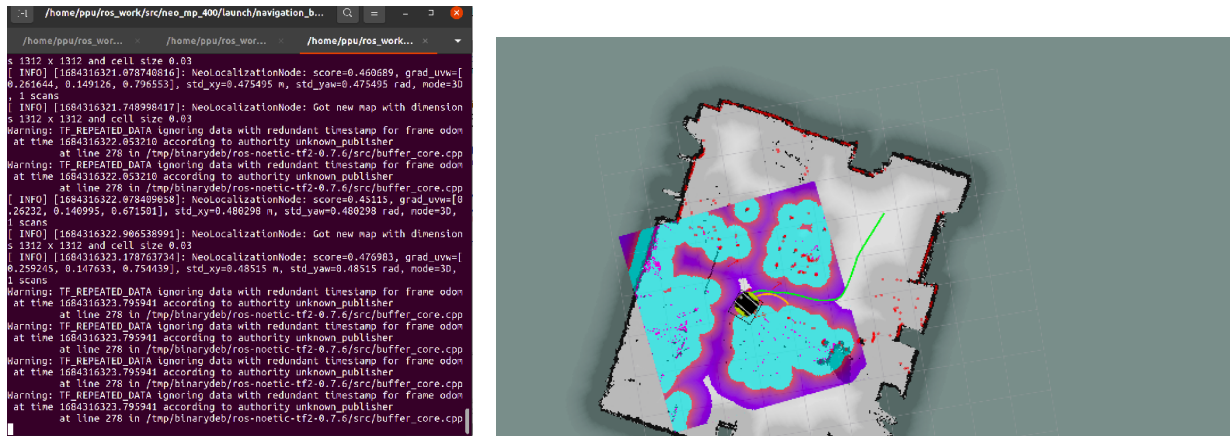
In order to ensure that a robot can safely and successfully navigate its surroundings, testing navigation systems is a critical step. The robot is equipped with the ability to respond to simple directions, choose the best course of action when navigating and, if necessary, change course to prevent negative outcomes along its path.as a show in Figure 5.6

we used the command in cmd to navigation :

```
$ roslaunch neo_mp_400 navigation_basic_neo.launch
```

5.3 Software Testing

In this section, we conducted tests on the website, and the Mobile Robot MP-400.



(a) navigation in Rviz

(b) navigation in cmd

Figure 5.6: test_navigation

5.3.1 Mobile Robot MP-400

We tested the Mobile Robot MP-400 using the Ultrasonic sensor and lidar. His response to impediments in his path was recorded as a show in Figure 5.7.

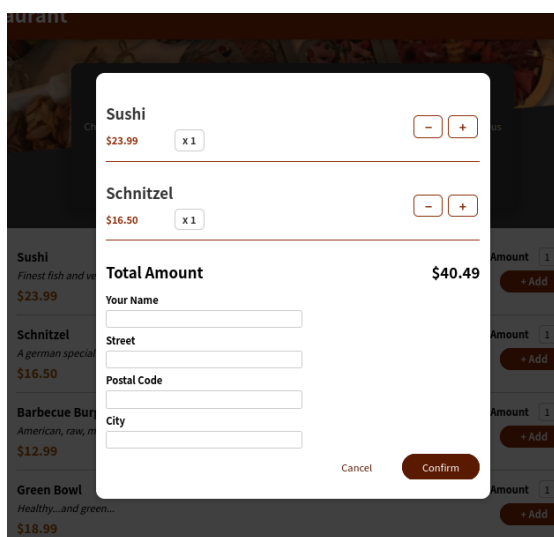
```
[ INFO] [1684317212.885904965]: I0Board: Active false Available false
[ INFO] [1684317212.885927046]: USBBoard: Active false Available false
[ INFO] [1684317212.901473332]: Communicating with RelayBoard
[ WARN] [1684317212.980143450]: Couldn't set gain on joystick force feedback: Bad file descriptor
[ WARN] [1684317231.452864233]: joint trajectory input timeout! Stopping now.
[ WARN] [1684317243.632835865]: joint trajectory input timeout! Stopping now.
[ WARN] [1684317249.992848438]: joint trajectory input timeout! Stopping now.
[ WARN] [1684317314.452834241]: joint trajectory input timeout! Stopping now.
[ WARN] [1684317636.532810621]: joint trajectory input timeout! Stopping now.
[ WARN] [1684317663.793132816]: joint trajectory input timeout! Stopping now.
[ WARN] [1684317674.81202321]: joint trajectory input timeout! Stopping now.
[ WARN] [1684317837.152885382]: joint trajectory input timeout! Stopping now.
[ ERROR] [1684318057.889301931]: Emergency stop was issued
[ WARN] [1684318057.832874025]: joint trajectory input timeout! Stopping now.
[ INFO] [1684318057.233179946]: Emergency stop was confirmed
[ INFO] [1684318058.240748573]: Emergency stop released
[ ERROR] [1684318097.784843223]: Emergency stop was issued
[ INFO] [1684318102.598530810]: Emergency stop was confirmed
[ INFO] [1684318103.606087295]: Emergency stop released
[ WARN] [1684318114.132850102]: joint trajectory input timeout! Stopping now.
[ ERROR] [1684318134.143730100]: Emergency stop was issued
[ WARN] [1684318136.472056530]: joint trajectory input timeout! Stopping now.
[ INFO] [1684318138.972164249]: Emergency stop was confirmed
[ INFO] [1684318139.979225144]: Emergency stop released
```

Figure 5.7: test of mp-400

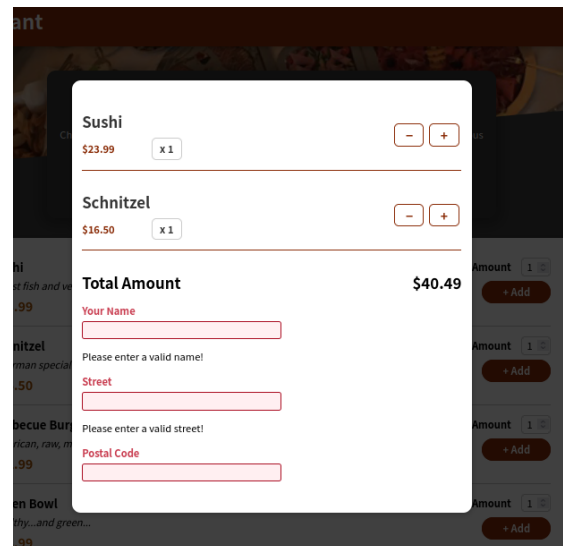
5.3.2 website

We ran a test to verify the accuracy of the readings obtained from the site after ordering and to make sure they are correct. Further, we checked the correct format for displaying information in the file.as a show in figure 5.8.a

As depicted in Figure 5.8.b, we can determine whether an error occurred during the request intake or if the information provided was incomplete.



(a) the order



(b) error in order

Figure 5.8: ordering process

In the subsequent figer 5.9, it is evident that if the process is executed correctly, the information will be transmitted to the Database, which will be prepared for sending to the dedicated chef's page for order processing.

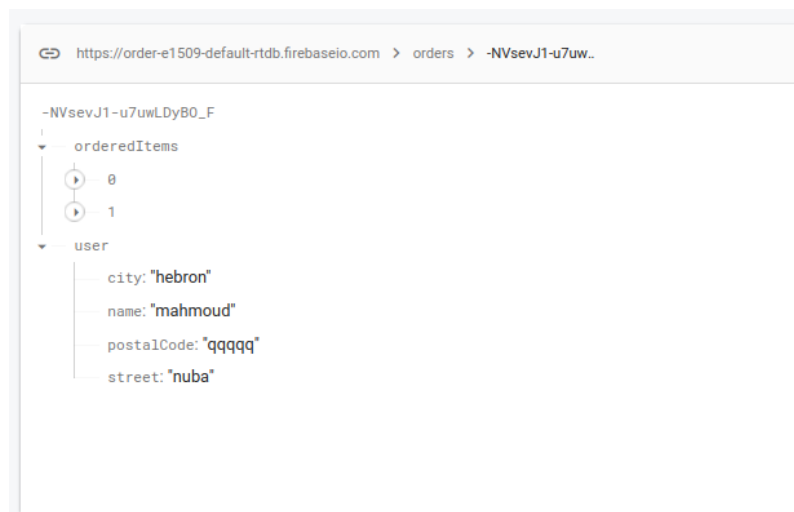


Figure 5.9: information in database

5.4 System Validation

Once each component has been individually connected to the computer and tested, we proceed to connect all the parts of the robot. We then execute the Python code to ensure that the system functions properly. The robot is expected to accurately map its location, receive data from the site correctly, move to the desired location, and return to its original starting point.

Chapter 6

Conclusion

6.1 Conclusion

In this project, we have presented an approach to build a smart prototype to create a Waiter Robot, The site mapping is drawn by moving the robot and using sensors.

We established a relationship between The web page and the computer () By using a common data base between them, the robot can also avoid the consequences that it faces and can, in the worst case, change the path it was walking on to a path that is less of obstacles in order to achieve speed in reaching the goal, which reduces the time it needs.

6.2 Future work

There are several features that can be added to this project to improve and enhance the machine. The following are some examples:

1. Adding several requests that it makes together.
2. Adding the ability to submit requests for other requests, according to the priority of the restaurant
3. The use of additional algorithms in order to increase the freedom of the router, especially in narrow areas

4. Converting the human or chef responsible for sending the robot into an automatic intelligent system
5. Adding an arm hand to the robot in order to submit requests
6. Added self-charging feature using auto-docking to recharge when The battery is low on its own

references

- [1] D. Akerele and K. Gidado, “The risks and constraints in the implementation of pfi/ppp in nigeria,” in *Proceedings of 19th Annual ARCOM Conference*, vol. 1, pp. 379–391, 2003.
- [2] “President’s report,<https://ifr.org/ifr-press-releases/news/presidents-report-09>.”
- [3] “gmapping,<http://wiki.ros.org/gmapping>.”
- [4] “All about circuits.”
- [5] F. Author and S. Author, “Nodes communication model in the ros environment,” Year. Accessed: May 20, 2023.
- [6] “Ros wiki,<http://wiki.ros.org>.”
- [7] “Ros command line tools,<http://wiki.ros.org/ros/commandlinetools>.”
- [8] “8 reasons to use ros in your robotics projects,<https://service.niryo.com/en/blog/8reasons-use-ros-robotics-projects>.”
- [9] F. Author and S. Author, *Title of the Chapter*, p. Page numbers. Publisher’s Location: Springer, Year.
- [10] M. Nassa, “Robot serves restaurant.food in restau.,” *Optometry and Vision Science: project in Palestine Polytechnic Universi*, vol. 66, no. 3, 2017.
- [11] “International research journal of management, economics, and social sciences.”
- [12] “What is arduino?,<https://docs.arduino.cc/learn/starting-guide/whats-arduino>.”
- [13] “Arduino-uno-perspective-transparent.png,<https://commons.wikimedia.org/wiki/file:arduino-uno-perspective-transparent.png>.”

-
- [14] “Ultrasonic sensor,<https://www.watelectronics.com/ultrasonic-sensor/>.”
- [15] “Ultrasonic sensor - hc-sr04,<https://www.sparkfun.com/products/15569>.”
- [16] “Urg-04lx lidar obstacle detection,<https://hokuyo-usa.com/products/lidar-obstacle-detection/urg-04lx>.”
- [17] “Urg-04lx-ug01 lidar obstacle detection,<https://hokuyo-usa.com/products/lidar-obstacle-detection/urg-04lx-ug01>.”
- [18] “Mobile robot mp-400,<https://www.neobotix-robots.com/products/mobile-robots/mobile-robot-mp-400>.”
- [19] “Javatpoint,<https://www.javatpoint.com/>.”
- [20] “What is a sequence diagram?,<https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-sequence-diagram/>.”
- [21] “ROS Wiki: map_server.” http://wiki.ros.org/map_server. Accessed: May 20, 2023.
- [22] “ROS Wiki: move_base.” http://wiki.ros.org/move_base. Accessed: May 20, 2023.