ACKNOWLEDGEMENT

The successful completion of the project is indeed practically incomplete without the mention of all those people who greatly supported and encouraged us throughout the project. We feel grateful to our guide DR. M. THACHAYANI, Professor, Department of Electronics andCommunication Engineering, Puducherry Technological University, for her encouragement and support. She has been a source of valuable guidance, suggestions, and kindness during the course ofproject work.

We would like to express our sincere thanks to our VICE CHANCELLOR Dr. S.MOHAN, Puducherry Technological University for providing the college facilities for the completion of this project.

We would like to express our heartfelt gratitude to Dr. V. SAMINADAN, Professor and Head of the Department of Electronics and Communication Engineering, Puducherry Technological University for providing all department facilities and guidance which enabled us to complete the project.

We feel obliged to thank the review by the panel members Dr. S. BATMAVADY Professor, Dr. V. VIJAYALAKSHMI Professor, and Dr. G. NAGARAJAN Professor, of the Electronics and Communication Engineering Department, for their support, encouragement, and guidance.

We express our deep sense of gratitude to all teaching and non-teaching staff of our department, and to our friends for their support and encouragement during the entire course of this dissertation work. We also express our sincere thanks to our parents who motivated us in all means

ABSTRACT

Our innovative Intelligent Supermarket Service Robot represents a transformative leap forward in retail technology, poised to revolutionize the shopping experience and optimize supermarket operations. At its core, our system is engineered to deliver personalized assistance and seamless functionality through an array of advanced features, including human following, product recognition, customer interaction, and automated billing.

Central to the system's architecture is the utilization of cutting-edge hardware components, strategically integrated for maximum efficiency and performance. Leveraging a Raspberry Pi as the central processing unit, equipped with natural language processing capabilities, our robot engages customers intelligently, responding to verbal commands and enhancing user experience through intuitive interactions. The Raspberry Pi, coupled with Edge Impulse for product identification, enables swift and accurate recognition of merchandise, facilitating streamlined checkout processes and reducing wait times for patrons.

Complementing the Raspberry Pi's capabilities, an Arduino Uno serves as the backbone of our human tracking and following functionalities. By employing precise distance maintenance techniques, the robot navigates through the supermarket environment autonomously, tailoring its movements to accommodate customers' pace and preferences. This integration of hardware platforms underscores our commitment to delivering a seamless and intuitive shopping journey for every customer.

Moreover, our system extends beyond immediate operational efficiencies to embrace predictive analytics powered by machine learning. By leveraging historical datasets of customer purchases, machine learning algorithms forecast demand patterns for individual products, empowering supermarkets to optimize inventory management proactively. This predictive capability Minimizes instances of out-of-stock scenarios, ensuring a consistent supply of sought-after items and enhancing overall customer satisfaction.

In essence, our Intelligent Supermarket Service Robot represents a convergence of cutting-edge technologies aimed at elevating the retail experience. By combining machine learning algorithms for object detection we have been able to identify products and bill them automatically as well as the smart shopping cart provides personalized assistance and efficient product management.

Through data-driven insights, our this system redefines the parameters of modern retail, by using previous sales data to forecast future demand and hence empowering supermarkets to operate with agility, foresight, and an unwavering commitment to customer satisfaction.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATION

ML	Machine Learning
GPIO	General Purpose Input/Output
YOLO	You Only Look Once
SSD	Single Shot Detector
ARIMA	Auto Regressive Integrated Moving Average
MAE	Mean Absolute Error
RMSE	Root Mean Square Error
IR	Infra Red
USB	Universal Serial Bus
DIY	Do It Yourself
DC	Direct Access Memory
RAM	Random Access Memory
HDMI	High Definition
OS	Operating System
IP	Internet Protocol
EDA	Exploratory Data Analysis
IDE	Integrated Development Environment
GNU	GNU's not Unix
FSP	Free Software Foundation
ARM	Advanced RISC Machine
RISC	Reduced Instruction Set Computer
LPDDR	Low Power Double Data Rate
BLE	Bluetooth Low Energy
AVR	Alf and Vegard's Risc Processor
PWM	Pulse Width Modulation
UART	Universal Asynchronous Receiver Transmitter

HC	High Conductance
EMF	Electromotive Force
CSV	Comma Separated Values

CHAPTER 1

INTRODUCTION

1.1 PRELUDE

The rapid evolution of automation and robotics has catalysed a new era of intelligent machines capable of seamlessly integrating into everyday life. In this paper, we introduce a pioneering concept: the Intelligent Supermarket Robot, leveraging the sophisticated capabilities of Raspberry Pi[1][2] and Arduino Uno[3] to revolutionize the retail landscape.

Our visionary approach to designing an Intelligent Supermarket Robot extends far beyond conventional robotics. At its core, this robot embodies accessibility and convenience, offering invaluable assistance to individuals with mobility restrictions, including the elderly and disabled. By autonomously following users and carrying their groceries, our robot transforms the shopping experience into a more inclusive and streamlined process, eliminating barriers and enhancing accessibility for all.

Moreover, our Intelligent Supermarket Robot is engineered to circumvent common shopping frustrations, such as congested aisles and lengthy checkout queues. By intelligently navigating through store environments, the robot optimizes time efficiency, allowing shoppers to focus on selecting items without undue delays.

The significance of our innovation extends beyond individual convenience to encompass strategic insights that drive operational excellence for store owners and managers. Through continuous data collection and analysis, our robot compiles valuable information on customer preferences, purchasing habits, and traffic patterns within the supermarket[4].This wealth of data empowers decision-makers to optimize inventory management strategies, refine product placement, and tailor marketing initiatives to align with evolving consumer behaviours.

By harnessing the synergistic capabilities of Raspberry Pi, Arduino Uno, and state-of-the-art sensors and algorithms, our platform represents a paradigm shift in retail automation. Beyond assisting individuals with mobility challenges, our Intelligent Supermarket Robot

serves as a catalyst for transformative change, elevating operational efficiency and delivering personalized experiences that resonate with today's discerning shoppers.

As the retail landscape evolves in the digital age, innovations like ours stand poised to redefine the supermarket experience. By embracing cutting-edge technology and human-centric design principles, we envision a future where intelligent robots enrich daily life, empower businesses, and inspire a new era of inclusive and personalized retail interactions. Through collaboration and innovation, we are shaping the future of retail, one supermarket aisle at a time.

1.2 MOTIVATION

Enhanced Customer Experience with Automated Checkout:

Automated checkout systems offer a transformative solution to reduce long queues and enhance customer service within supermarkets and retail stores. By expediting the checkout process through automated transactions[2], customers benefit from reduced waiting times and increased convenience, leading to improved overall satisfaction.

Optimized Inventory Management with Automation:

The constant manual monitoring of inventory levels by store staff often results in out-of-stock situations and heightened workload. Introducing automated inventory management systems enables real-time monitoring and predictive analytics, mitigating stock shortages and alleviating the burden on store personnel.

Streamlined Operations through Automation:

Automated solutions play a pivotal role in streamlining checkout processes and optimizing inventory management by leveraging demand prediction algorithms. By accurately forecasting product demands based on historical data and market trends, supermarkets can efficiently allocate resources and minimize operational inefficiencies.

Alleviating Wait Times and Staff Workload:

Long waiting times at billing sections not only inconvenience customers but also increase the workload on store staff. Implementing automated checkout systems reduces queues,

expedites transactions, and empowers employees to focus on delivering exceptional service rather than routine tasks

Enhanced Customer Service with Automation:

Poor customer service in stores can be significantly improved with the adoption of automad solutions. By automating repetitive tasks such as checkout and inventory monitoring, staff members can dedicate more time to addressing customer inquiries, providing personalized assistance, and cultivating positive interactions.

Minimized Out-of-Stock Incidents with Automated Inventory:

Automated inventory management techniques play a pivotal role in minimizing out-of-stock situations. By implementing real-time monitoring, automated replenishmentsystems, and predictive analytics, supermarkets can optimize stock levels, ensuring product availability and enhancing customer satisfaction.

1.3 LITERATURE SURVEY

TABLE 1.1

TITLE OF THE PAPER	AUTHORS	JOURNAL NAME, MONTH AND YEAR	TECHNIQUES PROPOSED IN THE PAPER
Design of Intelligent Supermarket Service Robot Based on Raspberry Pi[1]	Xue Han, Wei Wang, Fei Gao	2023 IEEE 3rd International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA 2023)	Hardware Configuration Raspberry Pi
Object Detection and Separation Using Raspberry Pi[2]	Sumeet Sanjay Walam, Bilal Sikandar Thakur, Siddhi Prakash Teli, Ravindra Ramchandra Nevarekar	2nd IEEE International Conference on Inventive Communication and Computational Technologies (ICICCT 2018)	Used Raspberry Pi and open CV for product detection and billing
Design and Control of Two-wheeled Self-Balancing Robot using Arduino[3]	Anmol Singh Shekawat,Yogesh Rohilla	2020 International Conference on SmartElectronics and Communication	Used Arduino Uno for Robot locomotion.
Effective Demand Forecasting Model using Business Intelligence..empowered with Machine Learning[4]	Muhammad Adnan Khan, Shaiza Saqib, Tahir Alyas, Anees Ur Rehman	IEEE Access Journal 2020	Concept of demand prediction with machine learning

Intelligent Shopping Cart Design Based on the Multi-Sensor Information Fusion Technology and Vision Servo Technology[5]	Qingqing Yuan, Zhiyong Liu, Fengnian Yang, and Ting Ma	IEEE Sensors Journal, Vol. 21, No. 22, November 15, 2021	Use of ultrasonic sensor and monocular camera for human tracking

1.4 LITERATURE REVIEW

The research paper titled "**Intelligent Shopping Cart Design Based on the Multi-Sensor Information Fusion Technology and Vision Servo Technology**" gave valuable insight on Intelligent Shopping Cart Systems, Multi-Sensor Information Fusion, Vision Servo Technology, Human-Robot Interaction (HRI), Applications and Case Studies and Challenges and Future Directions for the current system.

The research paper titled "**Design of Intelligent Supermarket Service Robo Based on Raspberry Pi**" was used to obtain information regarding Service Robotics in Retail, Raspberry Pi Integration, Raspberry Pi as a Robotic Platform, Computer Vision with Raspberry Pi, Real-world Deployments and Case Studies and Challenges faced while using Raspberry Pi module.

The research paper "**Object Detection and Separation Using Raspberry Pi**"has elucidate key aspects regrading Object Detection and Raspberry Pi such as Introduction to Object Detection and Separation, Raspberry Pi as a Platform, Object Detection Algorithms, Object Separation Techniques, Integration of Raspberry Pi with Object Detection Systems and Future direction and Trend in this particular field.

Specific information obtained from the research paper "**Effective Demand Forecasting Model using Business Intelligence empowered with Machine Learning**" such as Introduction to Demand Forecasting, Role of Business Intelligence in Demand Forecasting, Machine Learning for Demand Forecasting and Evaluation Metrics and Performance Benchmarking have been instrumental in shaping the project.

The paper on "**Design and Control of Two-wheeled Self-Balancing Robot using Arduino**" covers the robot's physical structure, sensor setup, and the control algorithm

implemented with Arduino. It discusses challenges faced during design, like sensor accuracy and motor responsiveness.

1.5 TECHNOLOGY USED AND ITS INTEGRATION

Multi-Sensor Information Fusion Technology:

The intelligent shopping cart exemplifies a sophisticated fusion of sensory technologies, utilizing an array of sensors including high-resolution cameras, ultrasonic sensors for distance measurement, and infrared sensors for environmental scanning. Through real-time data collection from these diverse sources, the cart gains a holistic perception of its surroundings. Advanced algorithms meticulously integrate and process this multi-dimensional sensory data, enabling the cart to make informed decisions autonomously. This fusion technology ensures heightened situational awareness, facilitating seamless navigation through dynamic store environments and enhancing user experience.

Hardware Configuration of Raspberry Pi:

The foundational hardware architecture of the supermarket service robot is centered around the Raspberry Pi, a versatile and cost-effective single-board computer renowned for its adaptability. This core platform is enriched with a suite of complementary hardware components meticulously integrated to amplify functionality. Equipped with high-definition cameras for visual input, ultrasonic sensors[5] for precise proximity detection, and responsive actuators facilitating agile movement, the robot embodies a sophisticated amalgamation of cutting-edge hardware technologies. Wireless communication modules such as Wi-Fi and Bluetooth establish seamless connectivity, enabling real-time interaction and control within complex environments.

Software Architecture of Raspberry Pi:

The software ecosystem of the Raspberry Pi-based robot is intricately crafted to harness the computational prowess of this platform. Rooted in a Linux-based operating system like

Raspbian, the software architecture encompasses a constellation of purpose-built modules tailored for diverse functionalities. These include modules for environment perception, leveraging image processing and computer vision techniques, as well as sophisticated navigation algorithms enabling precise maneuverability. Human interaction capabilities are augmented through intuitive software components enabling responsive and adaptive behavior. Furthermore, robust data processing modules ensure efficient handling of sensory inputs, driving the robot's intelligent decision-making processes.

Functionality and Features of Raspberry Pi:

The supermarket service robot is architected to deliver a spectrum of sophisticated functionalities tailored for the dynamic retail landscape. From autonomously navigating congested store aisles with precision to employing advanced object recognition algorithms for product identification, the robot epitomizes intelligent automation. Capable of fulfilling customer requests by swiftly locating and retrieving items, the robot enhances operational efficiency while enriching the shopping experience. Moreover, the robot's hazard detection capabilities ensure safe traversal through crowded environments, epitomizing a fusion of innovation and practicality.

Integration of Machine Learning:

Central to the robot's cognitive capabilities is the strategic integration of machine learning (ML) techniques, enabling adaptive and contextually aware behavior. Through ML-driven object recognition, the robot seamlessly identifies products and interacts with customers, fostering personalized experiences. This integration empowers the robot with the ability to learn and adapt over time, optimizing performance based on real-world interactions and feedback. By leveraging ML, the robot embodies a paradigm shift towards intelligent automation, enhancing versatility and responsiveness in retail settings

.**Raspberry Pi Setup and Camera Module:**

The project harnesses the computational prowess of the Raspberry Pi as the cornerstone of its architecture, facilitating robust object detection and separation capabilities.

Complemented by a dedicated camera module, the system captures high-resolution images and video feeds essential for precise object detection. Leveraging the Raspberry Pi's GPIO pins, the project seamlessly interfaces with an array of sensors and actuators, enabling real-time data acquisition and responsive control mechanisms.

Object Detection Algorithm and Separation Mechanism:

Critical to the project's success is the implementation of sophisticated object detection algorithms deployed on the Raspberry Pi platform. Utilizing techniques ranging from classical Haar cascades to state-of-the-art deep learning models like YOLO or SSD, the system achieves unparalleled accuracy in identifying and localizing objects within captured images or video frames. The integration of a robust separation mechanism, driven by motorized actuators and control logic, enables the physical manipulation of detected objects, epitomizing a convergence of hardware and software prowess.

Software Development and Practical Applications:

The project's software development lifecycle entails meticulous programming in Python, leveraging the Raspberry Pi's computational capabilities to interface with camera modules and process captured data. This comprehensive approach enables seamless integration of object detection algorithms with external components, fostering a synergistic fusion of software-driven automation. Realizing practical applications across diverse domains such as automated industrial sorting, smart waste management, and interactive installations, the project exemplifies the transformative potential of embedded systems and computer vision technologies.

Machine Learning (ML) Algorithms and Demand Forecasting:

In the domain of demand forecasting, ML algorithms represent a cornerstone in building predictive models that transcend traditional statistical methods. Leveraging techniques spanning regression models (e.g., linear regression, logistic regression) and time series analysis (e.g., ARIMA, exponential smoothing), demand forecasting models are imbued with unparalleled accuracy and adaptability. Advanced ML methodologies including neural networks and ensemble learning further enhance forecasting capabilities, enabling dynamic adaptation to evolving market dynamics and consumer behavior patterns.

Data Preprocessing, Feature Engineering, and Model Evaluation:

Central to the efficacy of ML-driven demand forecasting is the meticulous process of data preprocessing and feature engineering. This entails handling complex datasets through techniques like imputation of missing values, normalization, and dimensionality reduction to optimize model performance. Model evaluation and validation methodologies encompass robust techniques such as cross-validation and metrics like Mean Absolute Error (MAE) or Root Mean Square Error (RMSE), ensuring the reliability and generalizability of forecasting models across diverse scenarios.

Integration with Business Processes and Benefits:

The integration of ML-driven demand forecasting models within organizational business processes engenders a transformative impact on operational efficiency and strategic decision-making. By translating model outputs into actionable insights, organizations optimize inventory management, enhance production planning, and formulate targeted marketing strategies. Moreover, the benefits extend beyond operational realms, encompassing improved resource allocation, heightened customer satisfaction, and sustained competitive advantage in dynamic market landscapes. This holistic integration epitomizes a symbiotic relationship between data-driven intelligence and strategic foresight, fostering sustainable growth and resilience in contemporary business ecosystems.

1.6 PROBLEM STATEMENT

Supermarkets and retail stores encounter significant operational challenges characterized by persistent issues such as prolonged queues and suboptimal customer management, particularly within the billing sections. The resultant long waiting times not only inconvenience patrons but also impose additional burdens on store personnel, exacerbating overall operational inefficiencies. Concurrently, the absence of robust inventory management techniques necessitates continual manual oversight of product sales and stock levels, leading to frequent out-of-stock scenarios and further amplifying the workload on store employees.

In response to these multifaceted challenges, a compelling imperative emerges for the deployment of automated solutions capable of ameliorating congestion, optimizing checkout

processes, and revolutionizing inventory management through sophisticated demand prediction capabilities. By integrating advanced technologies such as machine learning and computer vision, these solutions have the potential to enhance operational fluidity, elevate customer experiences, and liberate store staff from mundane tasks, thereby fostering a more agile and competitive retail ecosystem.

The convergence of cutting-edge automation with strategic data-driven insights heralds a transformative era in retail operations, promising to mitigate operational bottlenecks, optimize resource allocation, and fortify supply chain resilience. Through proactive forecasting and real-time decision support, automated solutions stand poised to usher in a paradigm shift, empowering retailers to anticipate and respond dynamically to market fluctuations while nurturing enduring customer loyalty.

In summary, the imperatives driving the adoption of automated retail solutions underscore not only the exigency of operational optimization but also the strategic imperative of harnessing technology to realize unprecedented efficiencies and elevate the overall shopping experience.

CHAPTER 2

CIRCUIT DESCRIPTION

2.1 BLOCK DIAGRAM

The block diagram in Fig 2.1 given below is of a smart shopping cart utilizing an IR sensor, Arduino Uno, ultrasonic sensor, and two motors. The block diagram provides an outline on how each component interacts within the system.

SYSTEM MODEL – SMART CAR

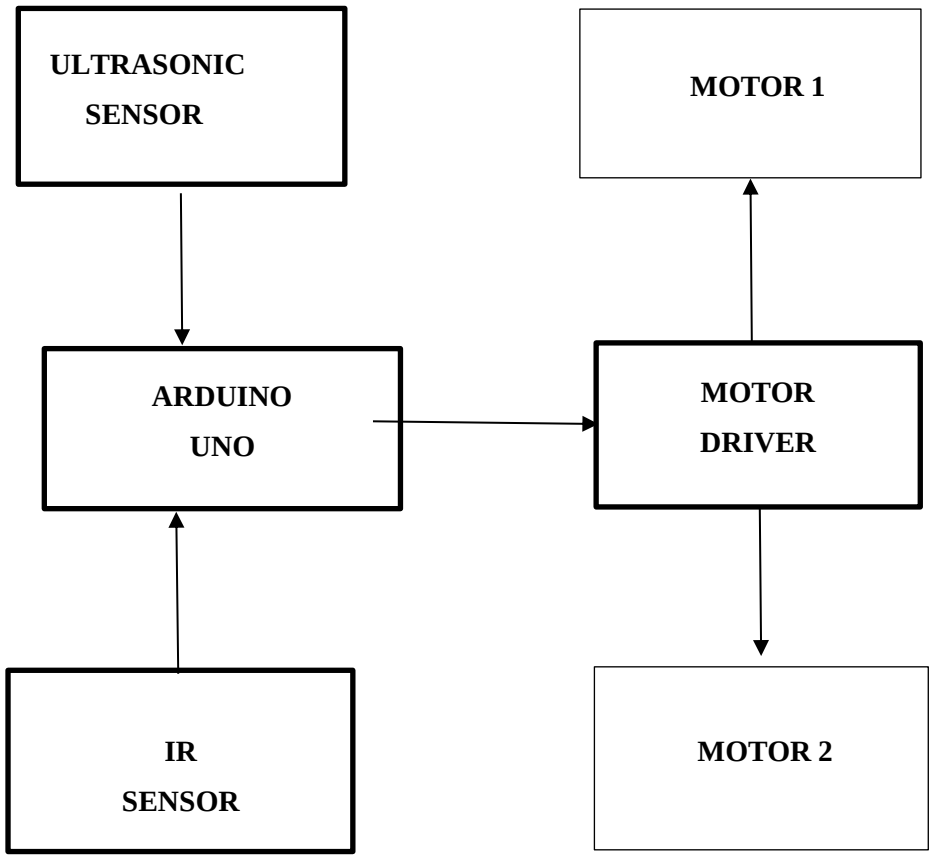

Fig 2.1 Smart cart-system model

The block diagram given below in Fig 2.2 gives an outline of the various components of the production detection module comprising of USB Camera connected to Raspberry PI which runs the detection software.

SYSTEM MODEL – PRODUCT DETECTION

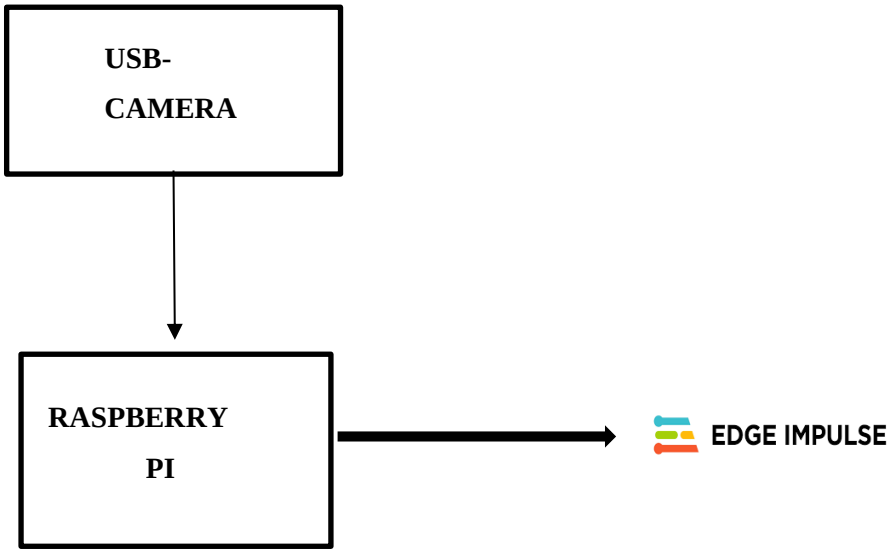

Fig 2.2 System model – product detection

2.2 BLOCK DIAGRAM DESCRIPTION

Ultrasonic sensor: An ultrasonic sensor is a device that emits high-frequency sound waves and detects their reflections off objects. It measures the time it takes for the sound waves to return, enabling it to calculate the distance to the object. This is used in shopping cart we are using for navigation purposes.

Arduino Uno: The Arduino Uno is a popular microcontroller board based on the ATmega328P chip, widely used for prototyping and DIY electronics projects. It features digital and analog input/output pins that can be programmed to interact with various sensors, actuators, and other electronic components. In this project we are using it to control the shopping cart movement.

IR sensor: An IR sensor detects infrared radiation emitted or reflected by objects using a transmitter and receiver pair. It's commonly used for proximity sensing and object detection in applications like security systems and consumer electronics, providing a non-contact method for detecting presence or motion.In this project we are using navigation purpose of robot.

Motor Driver: A motor driver is an electronic device that controls the speed and direction of electric motors. It typically takes low-power signals from a microcontroller or other control circuitry and converts them into higher-power signals suitable for driving motors. Motor drivers are essential components in robotics, automation, and electric vehicle systems, enabling precise control over motor movements. They come in various types, such as H-bridge and stepper motor drivers, each tailored to specific motor types and applications. In this application we use it for control the motors which are connected to the tyres of robot.

Motor: A 6V DC motor is a type of electric motor designed to operate on a 6-volt direct current power supply. These motors are commonly used in a wide range of applications, including hobby projects, small-scale robotics, and electronic devices. They come in various sizes and configurations, offering different torque and speed characteristics to suit specific needs. In this project we are using this for the movement of cart.

USB Cam: A USB camera, also known as a webcam, is a compact camera device that connects to a computer or other compatible device via a USB port. It is typically used for video conferencing, live streaming, video recording, and other multimedia applications. USB cameras come in various resolutions and features, offering options such as autofocus, built-in microphones, and wide-angle lenses. We are using this here for to give input images that is needed for object detection.

Raspberry Pi: The Raspberry Pi is a series of affordable single-board computers developed by the Raspberry Pi Foundation. Featuring a Broadcom system-on-chip, RAM, USB ports, HDMI output, and GPIO pins, these compact devices run various operating systems like Raspberry Pi OS and are popular for education, DIY projects, and small-scale computing tasks. In this project we are using as a computing source that is needed for demand prediction and object detection.

CHAPTER 3

SIMULATION/SOFTWARE TOOLS

3.1 EDGE IMPULSE

Edge Impulse provides powerful automations and low-code capabilities to make it easier to build valuable datasets and develop advanced AI for edge devices. Used by makers of health-wearable devices like Oura, Know Labs, and NOWATCH, industrial organizations like NASA, as well as top silicon vendors and over 100,000 developers on over 250,000 ML projects, Edge Impulse has become the trusted platform for enterprises and developers alike. We will use Edge Impulse to train an ML model and integrate it with a computing machine (Raspberry Pi) and acquire live video and images through camera interfacing

For data acquisition, take at least 100 photos of the different objects that you want to classify and use for training and testing the ML model.Put an object in front of the camera and the video of object can be seen in Live Classification Section of Edge Impulse without any label of the classified and detected object, such as the bottle, cup, or a person.

If you want to see it with IP address, run the command

edge-impulse-linux-runner

This command builds and downloads the model in Raspberry Pi and gives an IP address to see live classification. When you open this IP address in browser you can see the output http://192.168.1.19:4912

Edge Impulse stands out among similar software platforms for its user-friendly interface, seamless integration with popular microcontroller platforms like Arduino and Raspberry Pi, pre-built signal processing blocks for easy feature extraction, optimization techniques tailored for edge devices, and flexibility in deployment to various edge devices.Edge Impulse follows a simplified work cycle(Fig 3.1) .With a focus on simplicity, efficiency, and

flexibility, Edge Impulse offers developers of all skill levels an accessible solution for developing, training, and deploying machine learning models on edge devices.

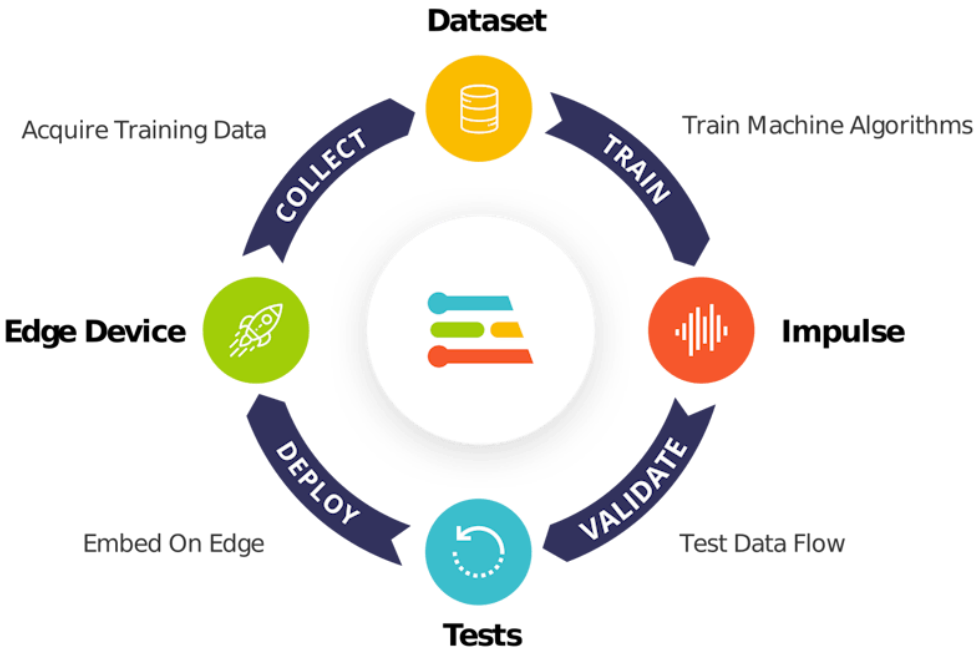

Fig 3.1 Edge Impulse work cycle

3.2 JUPYTER NOTEBOOK IN ANACONDA

JupyterLab is the latest web-based interactive development environment for notebooks, code, and data. Its flexible interface allows users to configure and arrange workflows in data science, scientific computing, computational journalism, and machine learning. A modular design invites extensions to expand and enrich functionality.

.Using Jupyter Notebook, we conducted exploratory data analysis (EDA) to gain insights into historical sales data and customer behavior patterns. We utilized Python libraries such as Pandas, NumPy, and Matplotlib within Jupyter Notebook to preprocess the data, handle missing values, and visualize trends over time shown in Fig 3.2.

Jupyter notebooks are used for all sorts of data science tasks such as exploratory data analysis (EDA), data cleaning and transformation, data visualization, statistical modeling, machine learning, and deep learning.

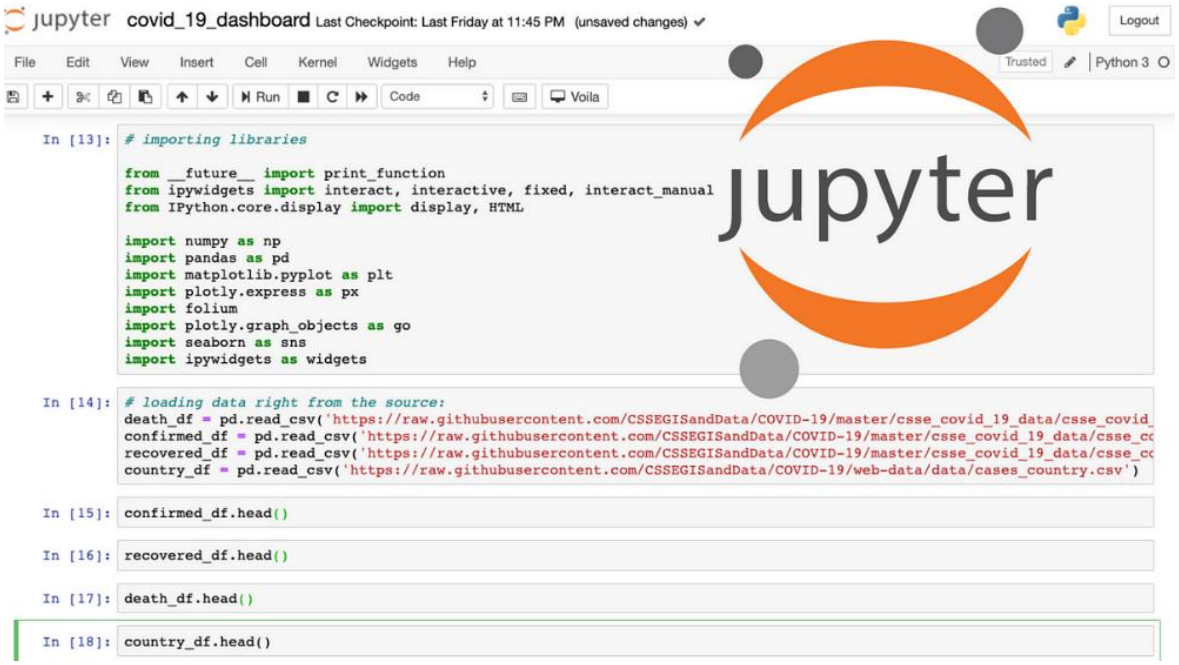

Fig 3.2 Executing Jupyter Notebook

Anaconda is favored in data science and machine learning for its robust package management system, encompassing essential libraries like NumPy, pandas, and scikit-learn. It simplifies environment setup and maintenance while offering tools like Jupyter Notebook for interactive computing, enabling efficient data analysis and model development.

Anaconda software helps you create an environment for many different versions of Python and package versions. Anaconda is also used to install, remove, and upgrade packages in your project environments. Furthermore, you may use Anaconda to deploy any required project with a few mouse clicks.

By integrating demand prediction capabilities into Jupyter Notebook, we aimed to optimize inventory management, enhance operational efficiency, and ultimately improve customer satisfaction in our smart cart project.

3.3 ARDUINO UNO IDE

The Arduino IDE(Fig 3.3) is a freely available software that enables users to write and upload code to Arduino boards. This application is compatible with various operating systems including Windows, Mac OS X, and Linux, and supports programming languages such as C and C++. In this context, IDE refers to Integrated Development Environment.

The process of writing code in the Arduino IDE is commonly referred to as sketching. To upload the sketch created in the Arduino IDE software, it is necessary to connect the Genuino or Arduino board to the IDE. The sketch is typically saved with the file extension '.ino.'.

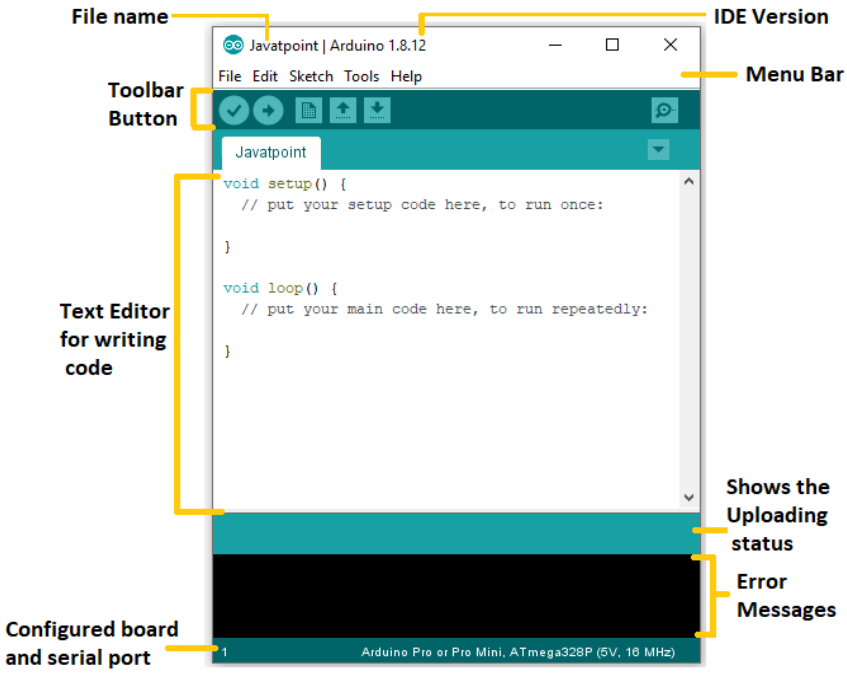

Fig 3.3 Arduino Uno IDE Interface

Once the code is written and verified for syntax errors, it is compiled into machine code, which is then uploaded to the Arduino board's memory. Upon uploading, the Arduino board executes the code, carrying out the specified tasks based on the programmed instructions such as movement of the robot in this project. This process enables users to create custom

applications and devices using Arduino microcontrollers, leveraging the platform's simplicity and versatility for a wide range of projects.

The Arduino IDE (Integrated Development Environment) is used to write the computer code and upload this code to the physical board. The Arduino IDE is very simple and this simplicity is probably one of the main reason Arduino became so popular. We can certainly state that being compatible with the Arduino IDE is now one of the main requirements for a new microcontroller board. Over the years, many useful features have been added to the Arduino IDE and you can now manage third-party libraries and boards from the IDE, and still keep the simplicity of programming the board.

3.4 DEBIAN LEGACY 64 BIT OS FOR RASPBERRY PI

Debian, established as one of the earliest operating systems built on the Linux kernel, remains the second oldest Linux distribution currently under active development, following Slackware. The project is managed online by a group of volunteers under the guidance of the Debian Project Leader and three key documents: the Debian Social Contract, the Debian Constitution.

Debian has historically been developed transparently and shared freely in alignment with the principles of the GNU Project and Free Software. As a result, the Free Software Foundation supported the project from November 1994 to November 1995. However, Debian is no longer endorsed by GNU and the FSF due to its practice of hosting non-free software repositories and, since 2022, including non-free firmware in its installation media by default.

To run code on a Raspberry Pi using a Debian-based OS like Raspbian or Debian itself, start by installing the OS and setting up the development environment. We wrote our code using a text editor save it with the appropriate file extension, and compiled. Then, we executed our code from the terminal by navigating to its directory and running the appropriate command. Debug any issues that arise and iterate as necessary to ensure our code runs as expected

3.5 CODE:

ARDUINO UNO CODE:

This code is for controlling a smart shopping cart prototype using an Arduino Uno board. It utilizes an ultrasonic sensor to detect person in front of the cart and two IR sensors to follow a line on the ground. When a person is detected within a specified range, the cart stops. Otherwise, it follows the line using the IR sensors. The motors are controlled accordingly to move the cart forward, turn left, turn right, or stop. Overall, the code enables the cart to navigate autonomously following a designated path.

```
// Define pins

const int trigPin = 9;

const int echoPin = 10;

// Define pins

const int leftIRPin = 2;   // Digital pin for left IR sensor

const int rightIRPin = 3;  // Digital pin for right IR sensor

const int leftMotorPin1 = 4;

const int leftMotorPin2 = 5;

const int rightMotorPin1 = 6;

const int rightMotorPin2 = 7;

// Define variables

long duration;

int distance;

void setup() {
```

```
  // Initialize serial communication
  Serial.begin(9600);
  // Set the trigPin as an OUTPUT
  pinMode(trigPin, OUTPUT);
  // Set the echoPin as an INPUT
  pinMode(echoPin, INPUT);
  // Set the motor control pins as OUTPUT
  pinMode(leftMotorPin1, OUTPUT);
  pinMode(leftMotorPin2, OUTPUT);
  pinMode(rightMotorPin1, OUTPUT);
  pinMode(rightMotorPin2, OUTPUT);
  // Set IR sensor pins as INPUT
  pinMode(leftIRPin, INPUT);
  pinMode(rightIRPin, INPUT);
  // Set motor control pins as OUTPUT
  pinMode(leftMotorPin1, OUTPUT);
  pinMode(leftMotorPin2, OUTPUT);
  pinMode(rightMotorPin1, OUTPUT);
  pinMode(rightMotorPin2, OUTPUT);
}
void loop() {
  // Clear the trigPin
```

```
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
// Set the trigPin HIGH for 10 microseconds
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
// Read the echoPin, calculate the distance in centimeters
duration = pulseIn(echoPin, HIGH);
distance = duration * 0.034 / 2;
// Print the distance on the serial monitor
Serial.print("Distance: ");
Serial.print(distance);
Serial.println(" cm");
// Check if the distance is within the specified range
if (distance >= 5 && distance <= 10) {
  // If the distance is within the range, run the line follower
  lineFollower();
}
else {
  // If the distance is not within the range, stop the motors
  stopMotors();
}
```

```
  // Delay before the next reading
}
void lineFollower() {
  // Write your line follower logic here
  // Read IR sensor values
  int leftIRValue = digitalRead(leftIRPin);
  int rightIRValue = digitalRead(rightIRPin);
  // Print sensor values (for debugging)
  Serial.print("Left IR Value: ");
  Serial.println(leftIRValue);
  Serial.print("Right IR Value: ");
  Serial.println(rightIRValue);
  // Check sensor readings and control motors accordingly
  if (leftIRValue == HIGH && rightIRValue == HIGH) {
    // Both sensors detect the line, move forward
    moveForward();
  } else if (leftIRValue == HIGH && rightIRValue == LOW) {
    // Only left sensor detects the line, turn right
    turnRight();
  } else if (leftIRValue == LOW && rightIRValue == HIGH) {
    // Only right sensor detects the line, turn left
    turnLeft();
```

```
  }
  else {
    // Both sensors do not detect the line, stop
    turnLeft();
    delay(100);
    turnRight();
  }
}
void moveForward() {
  digitalWrite(leftMotorPin1, HIGH);
  digitalWrite(leftMotorPin2, LOW);
  digitalWrite(rightMotorPin1, HIGH);
  digitalWrite(rightMotorPin2, LOW);
}
void turnLeft() {
  digitalWrite(leftMotorPin1, HIGH);
  digitalWrite(leftMotorPin2, LOW);
  digitalWrite(rightMotorPin1, LOW);
  digitalWrite(rightMotorPin2, LOW);
}
void turnRight() {
  digitalWrite(leftMotorPin1, LOW);
```

```
  digitalWrite(leftMotorPin2, LOW);
  digitalWrite(rightMotorPin1, HIGH);
  digitalWrite(rightMotorPin2, LOW);
}
void stopMotors() {
  digitalWrite(leftMotorPin1, LOW);
  digitalWrite(leftMotorPin2, LOW);
  digitalWrite(rightMotorPin1, LOW);
  digitalWrite(rightMotorPin2, LOW);
}
```

DEMAND PREDICTION CODE:

The given code performs demand prediction for a supermarket using linear regression. It utilizes pandas for data manipulation, scikit-learn for machine learning, and matplotlib for visualization. The accuracy of predictions is evaluated against real demand data, providing insights into model performance and potential improvements in forecasting accuracy.

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.impute import SimpleImputer
```

```
# Function to calculate demand

def calculate_demand(unit_price, quantity, rating):

    # Assuming a linear relationship between demand, unit price, quantity, and rating

    demand = 2 * unit_price + 3 * quantity - 4 * rating

    return demand



# Function to plot daily demand

def plot_daily_demand(df, month):

    # Filter data for the specified month

    month_df = df[df['Date'].dt.month == month].copy()



    # Group data by 'Product line' and 'Date'

    grouped = month_df.groupby(['Product line', 'Date'])



    # Initialize a dictionary to store daily demand for each product line

    daily_demand_dict = {}

        # Calculate demand for each group

    for (product_line, date), group_data in grouped:

        demand = calculate_demand(group_data['Unit price'], group_data['Quantity'], group_data['Rating']).sum()

        if product_line not in daily_demand_dict:
```

```
            daily_demand_dict[product_line] = {}
        daily_demand_dict[product_line][date] = demand

    # Plot daily demand for each product line
    plt.figure(figsize=(12, 8))
    for product_line, demand_data in daily_demand_dict.items():
        dates = list(demand_data.keys())
        demand_values = list(demand_data.values())
        plt.plot(dates, demand_values, label=product_line)

    plt.title(f'Daily Demand for Each Product Line in {month}')
    plt.xlabel('Date')
    plt.ylabel('Demand')
    plt.legend()
    plt.grid(True)
    plt.xticks(rotation=45)
    plt.tight_layout()
    plt.show()
# Function to calculate hypothetical daily demand for March
def calculate_hypothetical_demand(df, month):
    # Filter data for January and February
    jan_feb_df = df[df['Date'].dt.month.isin([1, 2])].copy()
```

```
    # Group data by 'Product line'
    grouped = jan_feb_df.groupby('Product line')

    # Calculate the average demand for each product line
    avg_demand_dict = {}
    for product_line, group_data in grouped:
        avg_demand=calculate_demand(group_data['Unitprice'], group_data['Quantity'], group_data['Rating']).mean()
avg_demand_dict[product_line] = avg_demand
    # Create a hypothetical daily demand DataFrame for March
    march_dates = pd.date_range(start='2023-03-01', end='2023-03-31', freq='D')
    march_demand_data=pd.DataFrame(index=march_dates, columns=avg_demand_dict.keys())

    # Fill in the hypothetical demand values
    for product_line in march_demand_data.columns:
        # Generate random variation around the average demand value
        variation = np.random.normal(loc=avg_demand_dict[product_line], scale=9, size=len(march_dates))
        march_demand_data[product_line] = avg_demand_dict[product_line] + variation

    return march_demand_data
```

```
# Function to prepare data for linear regression

def prepare_data_for_regression(df, march_demand_data):

    # Merge March demand data with original DataFrame

    df_regression = pd.merge(df, march_demand_data, how='left', left_on='Date', right_index=True)

    return df_regression



# Function to train linear regression model

def train_linear_regression_model(df):

    # Features for training the model

    X = df[['Unit price', 'Quantity', 'Rating']]



    # Target variable

    y = df['Demand']



    # Split data into training and testing sets


X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=4)

        # Impute missing values in the features using mean imputation

    imputer = SimpleImputer(strategy='mean')

    X_train_imputed = imputer.fit_transform(X_train)

    X_test_imputed = imputer.transform(X_test)
```

```
    # Initialize and train linear regression model
    model = LinearRegression()
    model.fit(X_train_imputed, y_train)

    return model, X_test_imputed, y_test

# Function to predict demand for March
def predict_demand_for_march(df, model):
    # Features for prediction
    features = df[['Unit price', 'Quantity', 'Rating']]

    # Predict demand using the trained model
    demand = model.predict(features)

    # Store the predicted demand in the DataFrame
    df['Demand'] = demand

    return df

# Path to the CSV file
csv_file_path = '/Users/cardoz/Desktop/Demand Prediction/supermarket.csv'
```

```
# Read CSV file
df = pd.read_csv(csv_file_path)

# Convert 'Date' column to datetime type
df['Date'] = pd.to_datetime(df['Date'])

# Plot daily demand for January
plot_daily_demand(df, month=1)

# Plot daily demand for February
plot_daily_demand(df, month=2)

# Calculate hypothetical daily demand for March
march_demand_data = calculate_hypothetical_demand(df, month=3)

# Print total predicted demand for each product line in March
print("\nTotal Predicted Daily Demand for Each Product Line in March:")
print(march_demand_data.sum())

# Plot hypothetical daily demand for March
plt.figure(figsize=(12, 8))
```

```
for product_line in march_demand_data.columns:
    plt.plot(march_demand_data.index,march_demand_data[product_line], label=product_line)

plt.title('Predicted Daily Demand for Each Product Line in March')
plt.xlabel('Date')
plt.ylabel('Demand')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()

# Plot total hypothetical daily demand for each product line in March as a bar graph
plt.figure(figsize=(10, 6))
total_hypothetical_demand_march = march_demand_data.sum()
total_hypothetical_demand_march.plot(kind='bar', color='skyblue')
plt.title('Total Predicted Daily Demand for Each Product Line in March')
plt.xlabel('Product Line')
plt.ylabel('Total Demand')
plt.xticks(rotation=45)
plt.grid(axis='y')
plt.tight_layout()
```

```
plt.show()


# Real demand data for March
real_demand = {
    'Electronic accessories': 5742.64,
    'Fashion accessories': 5141.02,
    'Food and beverages': 5561.54,
    'Health and beauty': 5454.66,
    'Home and lifestyle': 6192.68,
    'Sports and travel': 5658.30
}


# Calculate absolute difference between predicted and real demand for each product line
absolute_difference={product_line: abs(march_demand_data[product_line].sum() - real_demand[product_line]) for product_line in real_demand}


# Calculate accuracy for each product line
accuracy={product_line:(1-(absolute_difference[product_line] / real_demand[product_line])) * 10  for product_line in real_demand}
# Print accuracy for each product line
print("\nAccuracy for Each Product Line:")
for product_line, acc in accuracy.items():
```

```
    print(f"{product_line}: {acc:.2f}%")


# Calculate total absolute difference and total demand for all product lines
total_absolute_difference = sum(absolute_difference.values())
total_real_demand = sum(real_demand.values())


# Calculate Mean Absolute Percentage Error (MAPE)
MAPE = (total_absolute_difference / total_real_demand) * 100


# Calculate Mean Absolute Error (MAE)
MAE = total_absolute_difference / len(real_demand)


# Print results
print("\nMean Absolute Percentage Error (MAPE): {:.2f}%".format(MAPE))
print("Mean Absolute Error (MAE): {:.2f}".format(MAE))
```

BILLING CODE:

The C program given below is used to manage product records, allowing creation, loading, and saving of product data. It supports adding products from a CSV file, by extracting product IDs and quantities. Additionally, it can create a new CSV file with aggregated product data as well as print a bill.

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
```

```
#define MAX_PRODUCTS 100
#define FILENAME "Records.dat"
#define INVOICE_BASE_FILENAME "Invoice"
#define INVOICE_FILE_EXTENSION ".txt"
#define CSV_FILENAME "output.csv"
#define NEW_CSV_FILENAME "new_output.csv"
struct item {
    int productno;
    char productname[20];
    int quantity;
    int price;
};
struct customer {
    int productno;
    char productname[20];
    int quantity;
    int price;
    int amount;
};
struct item items[MAX_PRODUCTS];
struct customer cart[MAX_PRODUCTS];
int cart_index = 0;
```

```
int num_items = 0;
int invoice_number = 0;
void loadProducts() {
    FILE *fp = fopen(FILENAME, "rb");
    if (fp == NULL) {
        printf("No existing products.\n");
        return;
    }
    fread(&num_items, sizeof(int), 1, fp);
    fread(items, sizeof(struct item), num_items, fp);
    fclose(fp);
}
void saveProducts() {
    FILE *fp = fopen(FILENAME, "wb");
    if (fp == NULL) {
        printf("Error saving products.\n");
        return;
    }
    fwrite(&num_items, sizeof(int), 1, fp);
    fwrite(items, sizeof(struct item), num_items, fp);
    fclose(fp);
}
```

```
void create() {
    int i;
    printf("Enter the Number of Records: ");
    scanf("%d", &num_items);
    printf("\n");
    for (i = 0; i < num_items; i++) {
        printf("Enter Product Code: ");
        scanf("%d", &items[i].productno);
        printf("Enter Product Name: ");
        scanf("%19s", items[i].productname);
        printf("Enter Quantity: ");
        scanf("%d", &items[i].quantity);
        printf("Enter Price: ");
        scanf("%d", &items[i].price);
    }
    saveProducts();
    printf("Records are Created\n\n");
}
void addProductsFromCSV() {
    FILE *fp = fopen(CSV_FILENAME, "r");
    if (fp == NULL) {
        printf("Error opening CSV file.\n");
```

```
        return;
    }
int product_ids[MAX_PRODUCTS] = {0};
    int product_counts[MAX_PRODUCTS] = {0};
    int num_unique_products = 0;
    char line[1024];
    while (fgets(line, sizeof(line), fp)) {
        char *token;
        const char *delim = ",";
        int product_id;
        token = strtok(line, delim);
        if (token != NULL) {
            product_id = atoi(token);
            int i;
            for (i = 0; i < num_unique_products; i++) {
                if (product_ids[i] == product_id) {
                    product_counts[i]++;
                    break;
                }}
            if (i == num_unique_products) {
                product_ids[num_unique_products] = product_id;
                product_counts[num_unique_products] = 1;
```

```
            num_unique_products++;
        }}}
    fclose(fp);
    fp = fopen(NEW_CSV_FILENAME, "w");
    if (fp == NULL) {
        printf("Error creating new CSV file.\n");
        return;}
       for (int i = 0; i < num_unique_products; i++) {
        fprintf(fp, "%d,%d\n", product_ids[i], product_counts[i]);}
    fclose(fp);
fp = fopen(NEW_CSV_FILENAME, "r");
    if (fp == NULL) {
        printf("Error opening new CSV file.\n");
        return;
    }
```

CHAPTER 4

HARDWARE REQUIREMENTS

4.1 Raspberry Pi 4b

The Raspberry Pi shown in Fig 4.1 is used to carry out product detection. It carries out product detection by running a python code which utilises a Edge Impulse model file.

The Raspberry Pi 4 Model B is a versatile single-board computer designed for hobbyists, educators, and professionals. It features a powerful quad-core ARM Cortex-A72 processor, offering enhanced performance compared to previous models. With options for up to 8GB of RAM and support for dual 4K displays, the Raspberry Pi 4B is suitable for a wide range of projects, including media centers, retro gaming consoles, IoT devices, and more. Its GPIO pins allow for easy interfacing with sensors, actuators, and other hardware components, making it an ideal platform for learning and experimentation in the field of embedded computing.

RAM:

The Raspberry Pi 4 Model B is equipped with 4GB of LPDDR4-3200 SDRAM, providing substantial memory capacity for multitasking and running various applications simultaneously. This enhanced RAM capacity compared to earlier models allows for smoother performance and improved responsiveness**.**

Storage Options:

Raspberry Pi primarily utilizes microSD cards for storage, offering users the flexibility to choose storage capacities according to their needs. The microSD card slot allows for easy expansion of storage, accommodating operating system files, application data, and user files efficiently.

Networking:

With built-in Ethernet (RJ45) support, Raspberry Pi enables reliable wired network connections, ideal for scenarios where stable network connectivity is crucial. Additionally, dual-band Wi-Fi 802.11ac (2.4GHz and 5GHz) on newer models offers high-speed wireless networking, enhancing flexibility in network connectivity options.

Bluetooth:

The inclusion of Bluetooth 4.2 (BLE) on Raspberry Pi 4 and 3 facilitates seamless connections to Bluetooth-enabled peripherals such as keyboards, mice, speakers, and game controllers, expanding the device's versatility**.**

USB Ports:

Raspberry Pi 4 features USB 2.0/3.0 ports shown in Fig 4.1, providing ample connectivity options for connecting peripherals such as keyboards, mice, external storage devices, and more. These ports offer high-speed data transfer capabilities, enhancing overall user experience.

Video Output:

The HDMI port supporting up to 4K resolution delivers crisp and clear video output, making Raspberry Pi suitable for multimedia applications, digital signage, and high-definition displays.

Audio Output:

Raspberry Pi includes a 3.5mm analog audio-video jack, allowing users to connect speakers, headphones, or audio systems for audio playback. This feature enables multimedia applications and enhances the device's multimedia capabilities.

GPIO Pins:

General Purpose Input/Output pins provide Raspberry Pi with extensibility and versatility, allowing users to interface with a wide range of sensors, actuators, and other external devices. These pins enable users to create custom electronics projects and experiment with hardware interfacing.

Power Input:

Raspberry Pi boards feature a USB-C power input, offering a convenient and standardized power supply solution. The USB-C connector provides reliable power delivery, ensuring stable operation of the device in various usage scenarios.

Operating Systems:

Raspberry Pi supports a diverse range of operating systems tailored for different applications. Raspberry Pi OS (formerly Raspbian), based on the Debian Linux distribution, serves as the default operating system optimized for Raspberry Pi hardware. This versatile OS supports a wide array of applications, from educational programming to multimedia entertainment and IoT (Internet of Things) projects. Additionally, users have the flexibility to explore alternative operating systems, expanding the device's capabilities and compatibility with various software ecosystems.

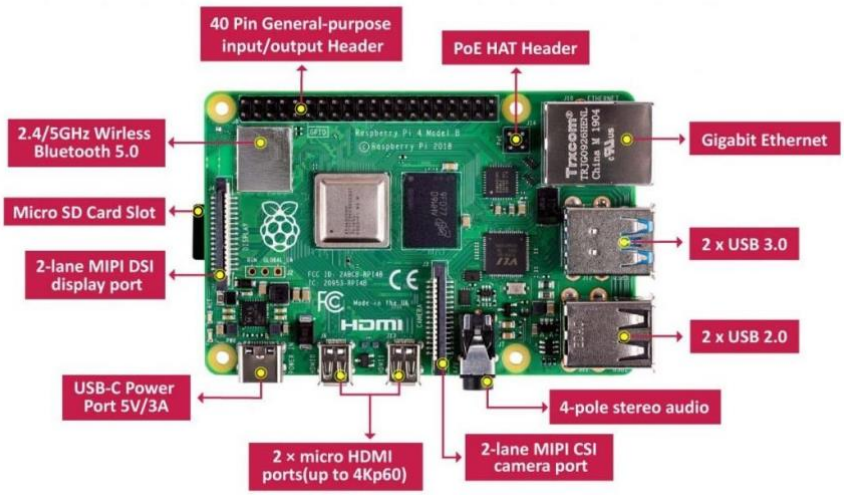

Fig 4.1 Raspberry Pi pins

4.2 Arduino Uno

The Arduino Uno shown in Fig 4.2 is used to control the smart shopping cart. It is interfaced with a IR sensor , Ultrasonic sensor and two motors. The ultrasonic sensor help keep a constant distant from the customer and the IR sensor is used to follow a designated path .

The Arduino Uno is a popular open-source microcontroller board based on the ATmega328P microcontroller chip. It features a straightforward design with various input and output pins, making it ideal for beginners and hobbyists interested in electronics and programming. The Uno board offers a user-friendly development environment, allowing users to write and upload code easily using the Arduino Integrated Development Environment (IDE). With its versatile capabilities, the Arduino Uno is widely used for prototyping projects, controlling electronic devices, and creating interactive gadgets and artworks. It serves as an accessible platform for learning electronics, programming, and building innovative projects across a wide range of applications, from home automation to robotics and beyond.

Memory and Voltage Specifications

With 32 KB of flash memory (0.5 KB of which is utilized by the bootloader), 2 KB of SRAM, and 1 KB of EEPROM, the Uno offers sufficient storage for program code, variables, and non-volatile data, respectively. Its operating voltage of 5V ensures compatibility with a wide range of electronic components and peripherals.

Input/Output Capabilities

In terms of input/output capabilities, the Arduino Uno boasts 14 digital I/O pins, including 6 pins capable of providing PWM output for generating analog-like signals(Fig 4.4). Additionally, it offers 6 analog input pins, enabling the board to read analog sensors and interface with analog devices. These I/O capabilities make the Uno versatile for a variety of projects, from simple LED blinking to complex sensor interfacing and motor control tasks.

Connectivity

Connectivity is another strength of the Arduino Uno, featuring a USB interface for programming and serial communication. This interface allows users to upload sketches to the board and communicate with it via a serial terminal. Furthermore, the Uno includes UART (Universal Asynchronous Receiver-Transmitter) pins for additional serial communication capabilities, enabling communication with other serial devices such as GPS modules, Bluetooth modules, and other microcontrollers.

Power Specifications

When it comes to power, the Arduino Uno can be powered via a DC power jack or through a USB connection, with an input voltage range of 7-12V (recommended). Additionally, it can be powered directly from the USB port, which supplies 5V. Understanding power requirements is crucial for designing Arduino projects, and the Uno's low current consumption in both active mode (~50 mA) and sleep mode (~8 mA) makes it suitable for battery-powered applications.

Timing and Clock Management

In the realm of timing and clock management, the Uno's 16 MHz clock speed provides precise timing for various applications. It features three timers (Timer0, Timer1, and Timer2), allowing for time-sensitive tasks such as generating PWM signals, measuring pulse widths, and creating precise delays.

Lastly, the Arduino Uno is supported by a robust development environment, consisting of the Arduino IDE (Integrated Development Environment) and an extensive collection of libraries. Programmed using the C/C++ language, the Uno's development ecosystem simplifies the process of writing code and interfacing with external hardware, making it accessible to beginners and experienced users alike.

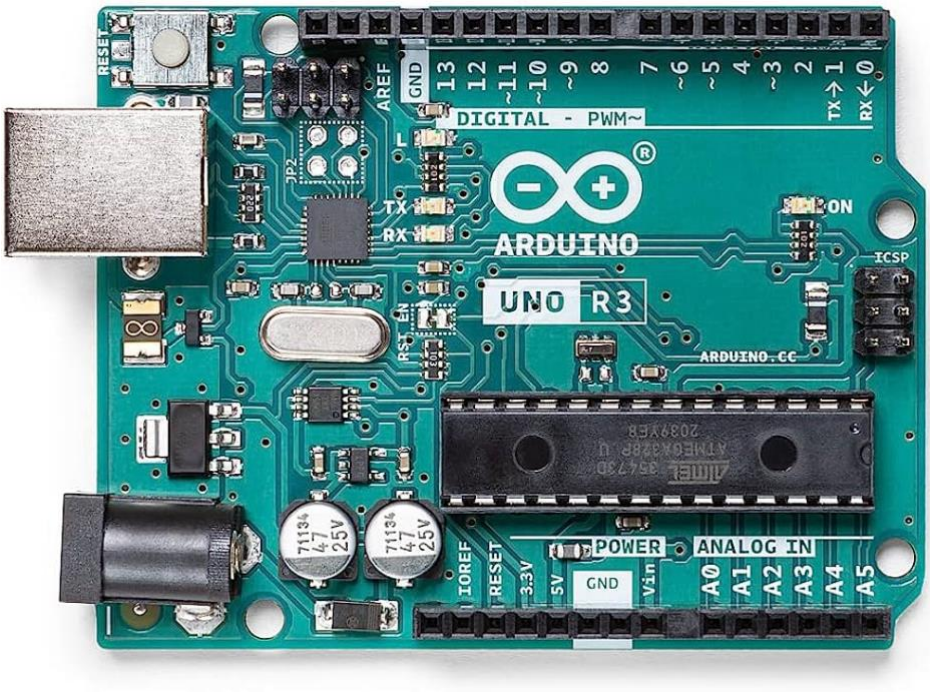

Fig 4.2 Arduino UNO

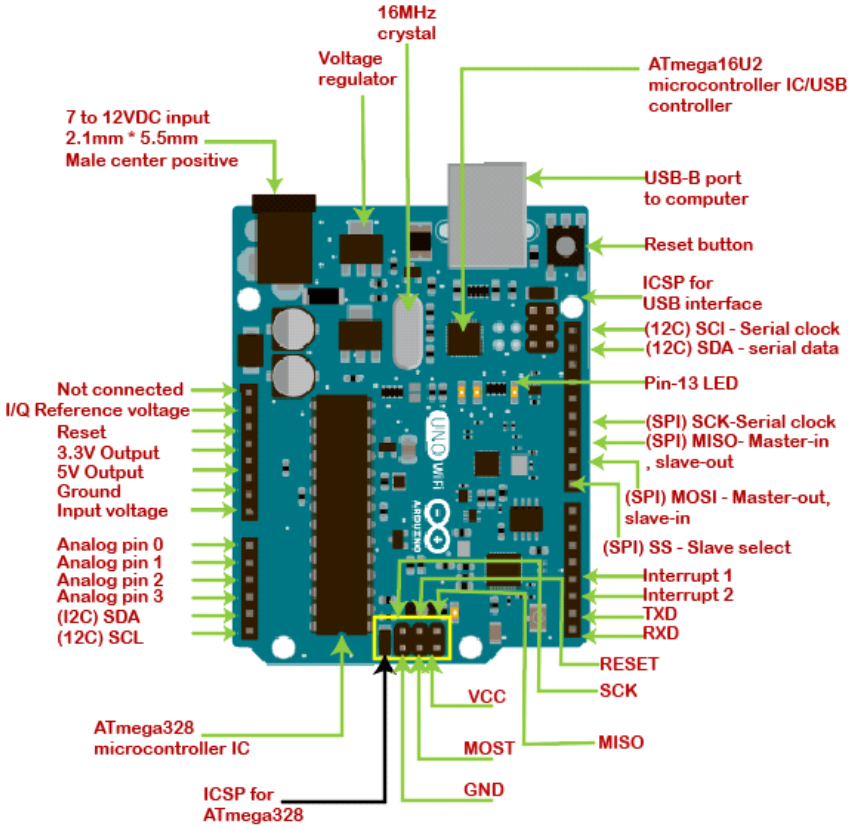

Fig 4.3 Arduino UNO pins

4.3 Ultrasonic Sensor:

The HC-SR04 ultrasonic sensor shown in Fig 4.4 is a popular model known for its simplicity and reliability in distance measurement applications. An ultrasonic sensor works by emitting high-frequency sound waves (ultrasonic waves) from a transmitter. These waves travel through the air and bounce off an object in front of the sensor. The sensor then receives the reflected waves using a receiver. By measuring the time it takes for the waves to travel to the object and back, the sensor can calculate the distance to the object based on the speed of sound in the air.In this project Ultrasonic sensor is used to measure the distance between person and the cart.

Detection Range:

The HC-SR04 has a detection range of approximately 2 cm to 400 cm (or 1 inch to 13 feet), making it suitable for a wide range of distance sensing applications.

Minimum Detection Distance:

The minimum detection distance of the HC-SR04 is around 2 cm, meaning it can reliably detect objects as close as 2 centimetres away.

Operating Voltage:

The HC-SR04 operates at a voltage range of 5V DC, making it compatible with most microcontroller systems, including Arduino and Raspberry Pi.

Beam Angle:

The ultrasonic beam emitted by the sensor has a narrow beam angle, typically around 15 degrees, allowing for precise distance measurements

Fig 4.4 Ultrasonic sensor

4.4 IR Sensor:

The REES52 IR sensor module shown in Fig 4.5, often based on the LM393 comparator IC, offers features suitable for various proximity sensing applications. In this project IR sensor is used to follow the designated path along with the customer. Its features typically include:

Detection Range:

The IR sensor module can detect objects within a specified range, typically a few millimetres to several centimetres, depending on the model and configuration..

Operating Voltage:

The IR sensor module operates within a wide voltage range, typically compatible with both 3.3V and 5V DC systems, making it versatile for various microcontroller platforms.

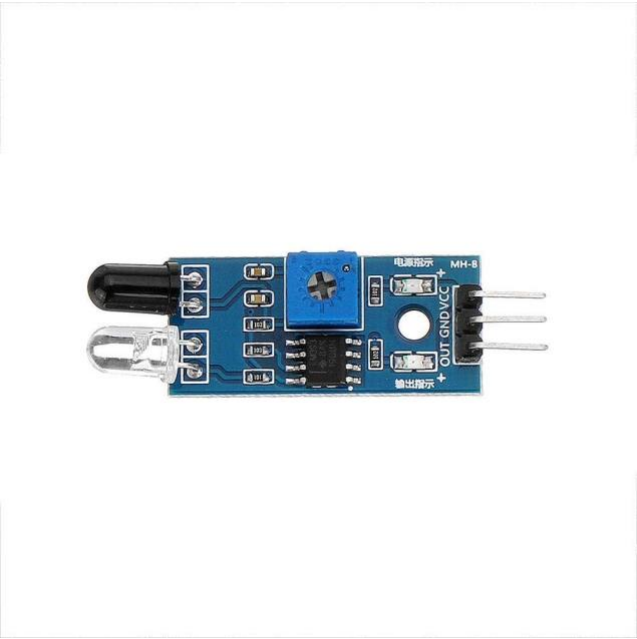

Fig 4.5 IR Sensor

4.5 Motor Driver L298N

The motor driver shown in Fig 4.6 is used to control the speed, direction, and operation of motors . Motor drivers provide the necessary electrical power and control signals to drive motors efficiently and safely as well as to provide the required power. The L298N motor driver module offers several features:

Dual H-Bridge Configuration:

The L298N module contains two H-bridge circuits, allowing it to control the direction and speed of two DC motors or one stepper motor.

High Current Capability:

It can handle high currents, typically up to 2A per channel and a peak current of 3A per channel, making it suitable for driving a wide range of motors.

Wide Voltage Range:

The L298N module operates over a wide voltage range, typically from 5V to 35V, allowing it to be compatible with various power sources and motor specifications.

PWM Speed Control:

The module supports pulse-width modulation (PWM) for speed control, enabling smooth and precise motor speed adjustments.

Built-in Diodes:

The L298N module includes built-in diodes (flyback diodes) for protection against back electromotive force (EMF), reducing the risk of damaging the module or other components.

Current Sensing Pins:

It features current sensing pins that allow monitoring of motor current, which can be useful for applications requiring feedback or protection mechanisms.

Built-in Thermal Protection:

Some versions of the L298N module include built-in thermal protection circuits to prevent overheating and ensure the module's safe operation under high load conditions.

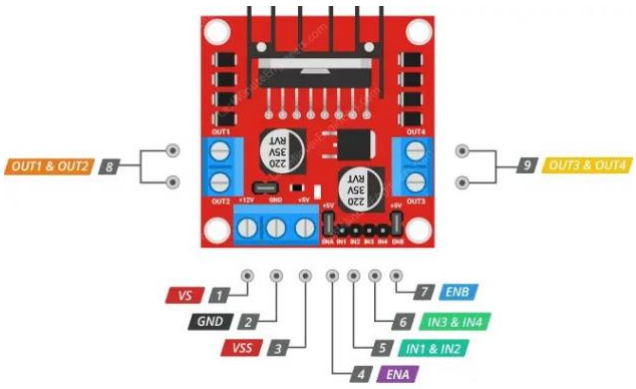

Fig 4.6 Motor Driver L298N

4.6 6V DC Motor:

6V DC Motor is being used in this project for locomotion of Smart Cart, Some features of 6V DC motors are:

Voltage Rating:

6V DC motors are specifically designed to operate with a 6-volt power supply, ensuring optimal performance and longevity.

Compact Size:

These motors are typically small and lightweight, making them suitable for integration into small-scale robotic systems without adding excessive weight or bulk.

High Torque:

Despite their small size, 6V DC motors often provide high torque output, allowing them to drive small robotic platforms and perform tasks requiring significant force.

Low Power Consumption:

Small-scale robotics often require energy-efficient components to prolong battery life. 6V DC motors are designed to operate efficiently, minimizing power consumption while delivering adequate performance.

Ease of Control:

These motors are compatible with a variety of motor driver circuits and microcontrollers commonly used in robotics projects, facilitating easy control and integration into robotic systems.

Durable Construction:

6V DC motors are typically built with durable materials and construction techniques, ensuring reliable operation in demanding robotic applications.

Affordability:

Due to their widespread use and mass production, 6V DC motors are often cost-effective, making them accessible for hobbyists, students, and small-scale robotics projects.

Versatility:

These motors come in various types, including brushed and brushless, geared and ungeared, allowing users to choose the type that best suits their specific application requirements.

CHAPTER 5

RESULTS AND DISCUSSION

5.1 PERFORMANCE METRICS

The main idea behind the project is to identify an object accurately ,get its Product ID and print a bill according to the type and number of objects identified. The smart shopping cart has to follow the customer accurately within a fixed distance along a destined path. Another stand alone part of the project is to take in the quantity price and rating of the items sold for a time period and predict the demand for another period from those data using machine learning. The performance metrics for the project are given as:

1.Model Testing Accuracy

Model accuracy is a measure of how well a machine learning model is performing. It quantifies the percentage of correct classifications made by the model.

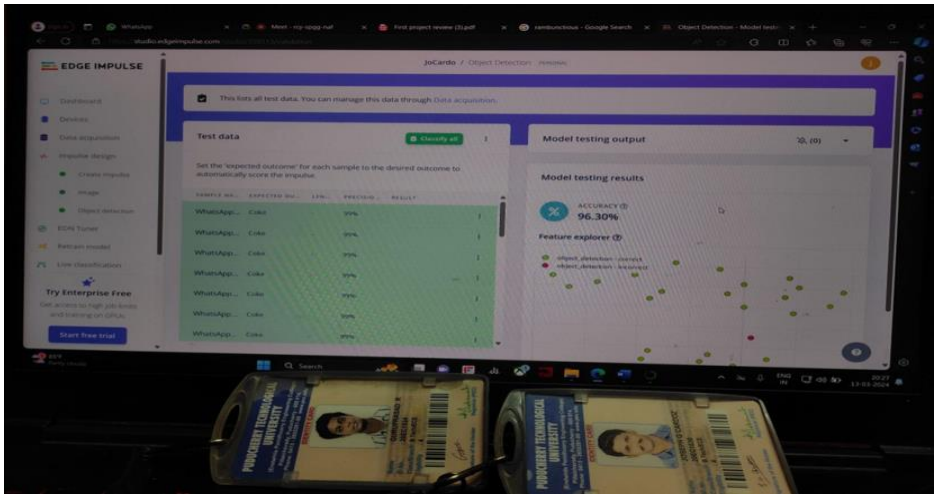

Fig 5.1 Model testing results

It is typically measured using evaluation metrics such as Intersection over Union (IoU) which measures the overlap between the predicted bounding box and the ground truth bounding box of an object, and Average Precision (AP) which computes the precision-recall

curve for different confidence thresholds of the model predictions and calculates the average precision value. Higher AP values indicate better object detection performance. As shown as in FIGURE 5.1 the Model accuracy of this model was found out to be 96.30%.

2. Object Detection Accuracy

Object detection using bounding boxes involves not only detecting the presence of objects but also precisely localizing them within the image using bounding boxes is one of evaluation metric that can be considered for object detection accuracy.

Intersection over Union (IoU) measures the overlap between the predicted bounding boxes and the ground truth bounding boxes. It's calculated as the ratio of the area of intersection to the area of union between the predicted and ground truth bounding boxes.

Higher IoU values indicate better localization accuracy. Typically, a threshold IoU value (e.g., 0.5 or 0.75) is used to determine whether a predicted bounding box is considered a true positive. In Fig 5.2 value of Item 1 is found to be 0.99,so that object is predicted to be the same. Similar situation can be seen in Fig 5.3 where score of Item 2 is found to be 0.95 so it is also correctly predicted .

Fig 5.2 Object detection accuracy of item 1

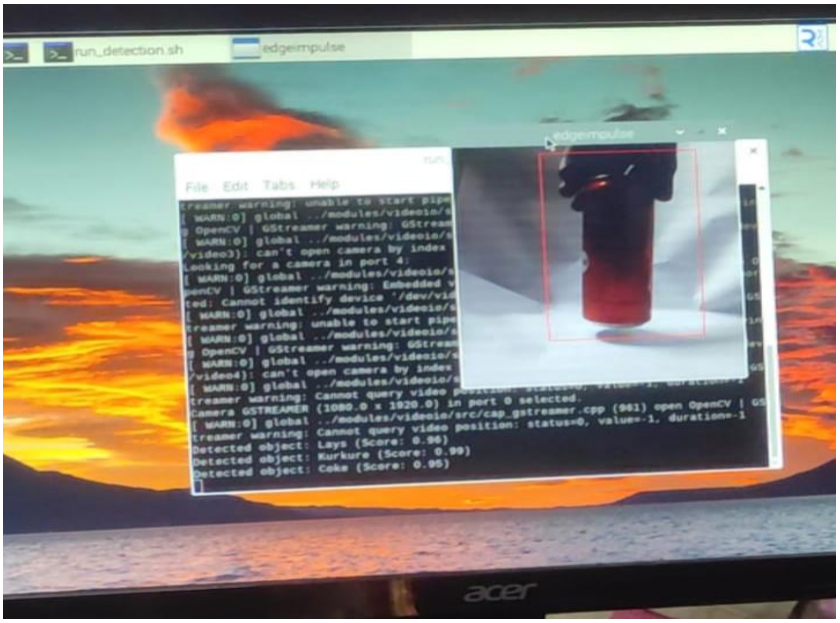

Fig 5.3 Object detection accuracy of item 2

The accuracy of an object detection model depends on the quality and number of training samples, the input imagery, the model parameters, and the requirement threshold for accuracy.

3. Demand Prediction Accuracy

Linear regression model was used for the Demand Prediction Task due to its simplicity making it suitable for demand forecast. We trained the model using a subset of the data and validated its performance on a separate validation set to ensure robustness.

The accuracy of our demand prediction model was evaluated using metrics such as Root Mean Squared Error (RMSE), which quantify the difference between predicted and actual demand values. In Fig 5.4 Demand Prediction Accuracy of each item is displayed.

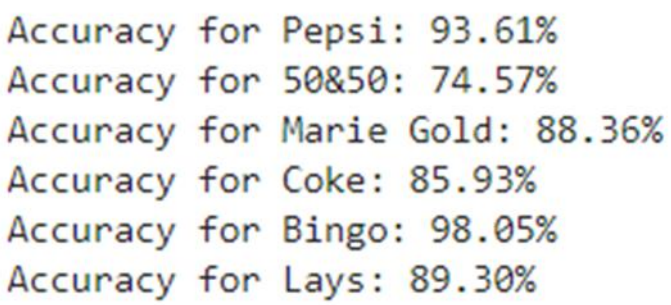

Fig 5.4 Demand prediction accuracy of each item

By continuously refining the model and incorporating new data, we aim to further improve the accuracy of our demand predictions and enhance decision-making processes.

4.Following Distance

The smart shopping cart uses an Ultrasonic sensor to calculate the distance between the customer and the cart. If the customer is within a distance in the range of 40 cm to 70 cm it follows the customer along the designated path, If the distance between the shopping cart is less than 40 cm the cart comes to a halt.

5.2 OUTPUTS OBTAINED

Console Based Application for Billing

We integrated a Billing Module with our system, so as every object is shown before USB Camera the details of that object such as its product ID ,price ,and quantity is added to a CSV file and after a record is created as show in Fig 5.5

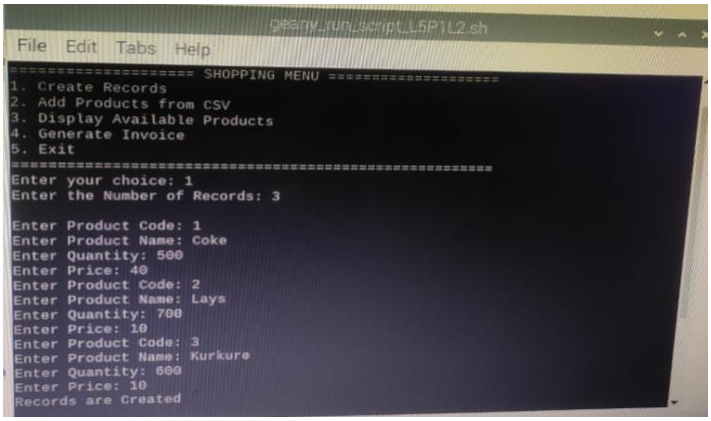

Fig 5.5 Product record creation

The final generated Bill is given in below in fig 5.6. All the products bought are displayed along with their Product number, Product name, Quantity bought and Amount to be paid.

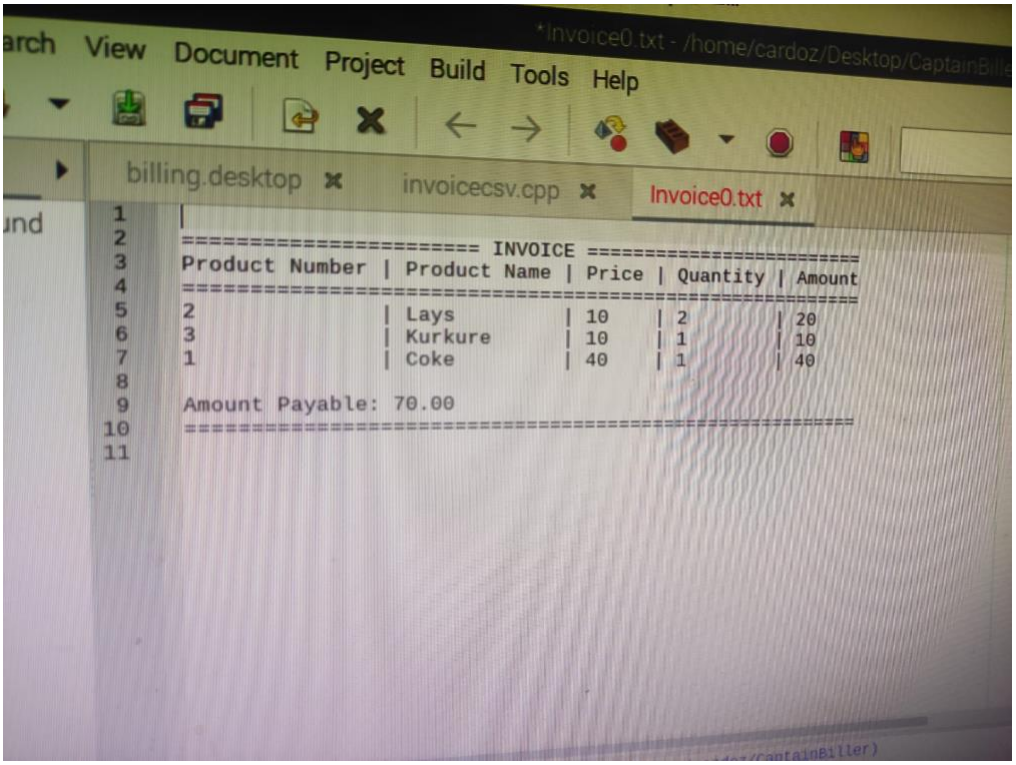

Fig 5.6 Final bill generation

Demand Prediction using Jupyter Notebooks

We used Jupyter Notebooks in anaconda for demand prediction and we obtained a graphical representation of both Actual demand and Predicted demand as a output, We also get details about Prediction accuracy which depicts how well our system is performing.

Actual Demand

The sales data for the month of March is already preent in the gien csv file.The Actual demand data for the month of March is then calculated from the csv file and given below in Fig 5.7.

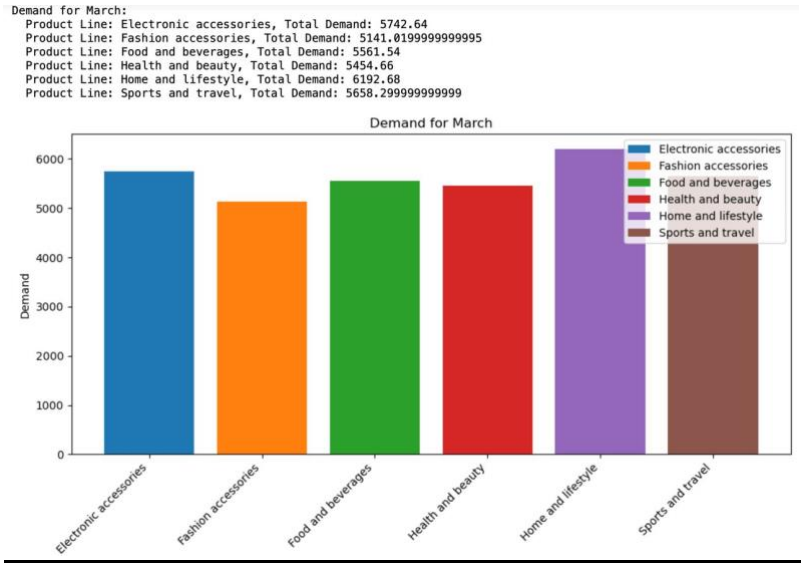

Fig 5.7 Actual demand for March

Predicted Demand

By analyzing historical sales for the month of January and February, data and relevant features such as date, pricing, quantity sold , the linear regression model predicts future demand levels. We trained the model using a subset of the data and validated its performance

with respect to original data ensure robustness**.** The predicted demand for the month of March based on sales data of the previous two months is given belwo in Fig 5.8

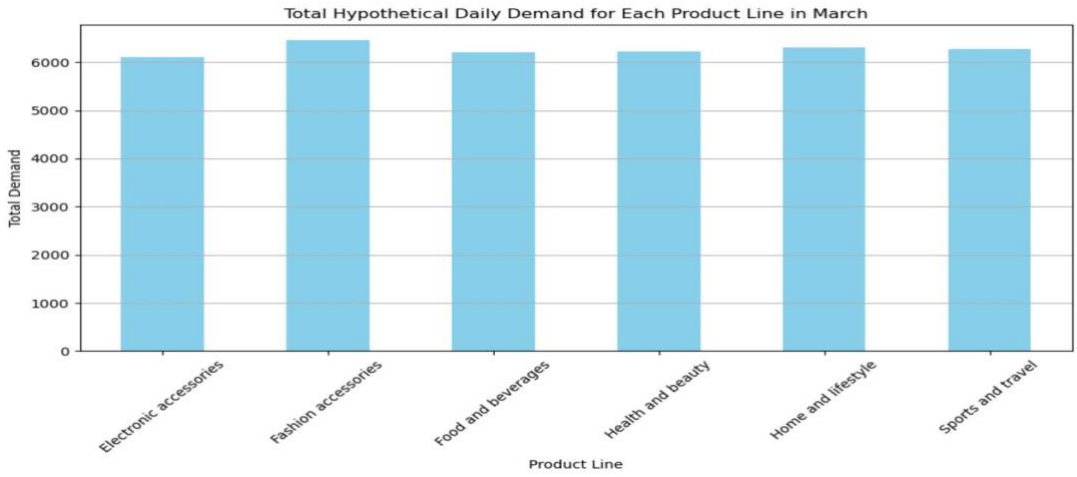

Fig 5.8 Predicted demand

So, in our project the product places in front of the USB camera is identified as shown in Fig 5.1. Then along with its Product ID, price and name and the output of object detection is written into a csv file and records are generated as in Fig 5.6 . This csv file is then used to generate a bill and the generated bill is displayed to the suer. The sales data hence stored in csv file for a period of time can be then used to calculate the demand for each product in the store as well as predict future demand for the given product as in Fig 5.9.

CHAPTER 6
CONCLUSION AND FUTURE SCOPE

6.1 CONCLUSION:

This project involves development of an automated shopping cart robot integrating Arduino Uno,5V DC Motors,Motor drivers,USB Camera,Ultrasonic senor,IR sensor and Raspberry PI which also harness the power of machine learning at the edge impulse, coupled with demand predictive analytics, aims to revolutionize the shopping experience, making it more convenient and efficient for both customers and retailers alike. This automated shopping cart robot represents a paradigm shift in retail automation, combining cutting-edge technologies to create a seamless and personalized shopping experience. By leveraging machine learning and predictive analytics, this innovative solution has the potential to revolutionize the retail industry, driving efficiency, enhancing customer satisfaction, and paving the way for the future of retail shopping.

6.2 FUTURE SCOPE

The future scope of the automated shopping cart robot project is promising, with opportunities for advancement in various areas. These include enhancing object detection accuracy, implementing real-time inventory management, refining navigation algorithms for smoother movement in crowded environments, offering personalized shopping experiences through recommendation systems, integrating multi-modal interaction capabilities, connecting with e-commerce platforms for seamless online-offline integration, providing retailers with actionable analytics and insights, and ensuring scalability and adaptability to evolving technology and market trends. By embracing these possibilities, the project can continue to innovate and redefine the retail shopping experience, driving efficiency, convenience, and customer satisfaction to new heights.

REFERENCES:

[1] Han, X., Wang, W., & Gao, F. (2023). Design of Intelligent Supermarket Service Robot Based on Raspberry Pi. In 2023 IEEE 3rd International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA) (pp 609612).Chongqing,China.https://doi.org/10.1109/ICIBA56860.2023.10165533

[2] B. Rasidi, N. Ilya, O. I. Al-Sanjary, M. Y. Kashmola, and K. L. T. Aik, "Development on Autonomous Object Tracker Robot using Raspberry Pi," in IEEE 10th Conference on Systems, Process & Control (ICSPC), 2022, pp. 29-33.

[3] S. S. Walam, S. P. Teli, B. S. Thakur, R. R. Nevarekar and S. M. Patil, (2018). Object Detection and Separation Using Raspberry Pi. In 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT pp 214-217 Coimbatore, India. https://doi.org/10.1109/ICICCT.2018.8473068

[4] A. S. Shekhawat and Y. Rohilla, (2020) .Design and Control of Two-wheeled Self-Balancing Robot using Arduino In 2020 International Conference on Smart Electronics and Communication (ICOSEC), Trichy, India, 2020, pp. 1025-1030, doi: 10.1109/ICOSEC49089.2020.9215421.

[5] M. A. Khan et al., (2020). Effective Demand Forecasting Model Using Business Intelligence Empowered with Machine Learning. IEEE Access,8,116013-116023. https://doi.org/10.1109/ACCESS.2020.3003790

[6] Q. Yuan, Z. Liu, F. Yang and T. Ma (2021). Intelligent Shopping Cart Design Based on the Multi-Sensor Information Fusion Technology and Vision Servo Technology. IEEE Sensors Journal, 21(22), 26033-26041. https://doi.org/10.1109/JSEN.2021.3116956

[7] S. Poonkuntran, R. K. Dhanraj, and B. Balusamy, "Object Detection with Deep Learning Models: Principles and Applications," 1st Edition, Chapman and Hall, 2022.

[8] Edge Impulse, "Raspberry Pi 4," Edge Impulse Documentation, https://edge-impulse.gitbook.io/docs/edge-ai-hardware/cpu/raspberry-pi-4. Accessed: April 30, 2024.

[9] D. H. Maulud and A. M. Abdulazeez, "A Review on Linear Regression Comprehensive in Machine Learning," Journal of Advanced Studies in Topology and Topological Algebras, vol. 13, no. 3, pp. 32-41, Mar. 2021. DOI: 10.38094/jastt1457.

[10]"Raspberry Pi Hardware Monitoring," Linux Kernel Documentation, https://www.kernel.org/doc/html/v6.6/hwmon/raspberrypi-hwmon.html. Accessed: April 30, 2024.