

SMART SHOPPING TROLLEY WITH TO CONTROL USING TOUCH PAD

Submitted in partial fulfillment of the requirements for the award of Bachelor of
Engineering Degree in Electrical and Electronics Engineering

by

SUDARSAN T (Register No. 40140704)



**DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING
SCHOOL OF ELECTRICAL AND ELECTRONICS**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with Grade "A++" by NAAC

**JEPPIAAR NAGAR, RAJIV GANDHI
SALAI, CHENNAI - 600119, TAMILNADU**

APRIL-2024



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

[DEEMED TO BE UNIVERSITY]

CATEGORY-I UNIVERSITY BY UGC

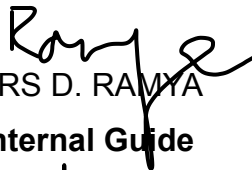
Accredited "A++" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **SUDARSAN T** (Reg. No. 40140704) who carried out the project entitled "**PARKING MANAGEMENT SYSTEM USING IOT TECHNOLOGY**" under my supervision from November 2023 to April 2024.



MRS D. RAMYA

Internal Guide



Dr. S. RADHIKA

Head of the Department

Submitted for Viva voce Examination held on _____

Name:

Signature:



Internal Examiner



External Examiner

DECLARATION

We, **SUDARSAN T** (Reg.No.40140704) hereby declare that the Project Report entitled “**SMART SHOPPING TROLLEY WITH TO CONTROL USING TOUCH PAD**” done by us under the guidance of **MRS D. RAMYA** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Electrical and Electronics Engineering.

1.

2.



SIGNATURE OF THE CANDIDATE

DATE:

PLACE: CHENNAI

ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to Board of Management of **Sathyabama Institute of Science And Technology** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

We convey my thanks to **Dr. N.M. Nandhitha, M.E., Ph.D., Dean, School of Electrical and Electronics**, and **Dr. S. Radhika, Head of the Department, Dept. of Electrical and Electronics Engineering** for providing us necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to our Project Guide **MRS D. RAMYA**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of our project work.

We wish to express our thanks to all Teaching and Non-teaching staff members of the Department of Electrical and Electronics who were helpful in many ways for the completion of the project.

ABSTRACT

This paper presents the design and implementation of an automated smart shopping trolley system utilizing a Raspberry Pi 4 Model B and LM298N motor driver. The system aims to streamline the shopping experience by integrating automatic billing, product scanning via a website and mobile app interface, and reducing waiting times at billing counters. The methodology involves detailed hardware setup, software development, and integration with existing technologies. A comprehensive literature review is provided to contextualize the project within the existing body of research. The system's effectiveness in improving shopping efficiency and customer satisfaction is evaluated, leading to valuable insights and conclusions.

TABLE OF CONTENT

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	v
	TABLE OF CONTENT	vi
	LIST OF FIGURES	viii
1	INTRODUCTION	
	1.1 General Introduction	1
	1.2 Literature Survey	3
	1.3 Scope Of The Project	6
	1.4 Advantages Of Proposed Technique	7
	1.5 Applications Of Proposed Technique	7
2	PROJECT DESCRIPTION	
	2.1 Block Diagram Of shopping trolley	8
	2.2 Components	9
	2.3 Modules Name	13
3	METHODOLOGY	
	3.1 Proposed Methodology	22
	3.2 Implementation	23
	3.3 Application Frame Work	
4	SIMULATION THEORY	
	4.1 Proteus Description	46
	4.2 Raspberry Simulation	47
5	SOFTWARE PROGRAM DEVELOPED FOR PROPOSED PROJECT	
	5.1 Automatic speed control Arduino Code	49

	5.2 Automatic speed control based Raspberry Pi code	51
6	SIMULATION RESULT AND HARDWARE RESULT	
	6.1 Simulation result	58
	6.2 Application result	59
	6.3 Hardware result	
7	CONCLUSION	
	7.1 Conclusion	60
	7.2 Future Scope	

References

LIST OF FIGURES

FIGURE NO	TITLE	PAGE NO
2.1	Block Diagram of proposed system	15
2.2	Visual representation of RPI 3B	17
2.3	pictorial view of dc motor	18
2.4	schematic view of motor driver lm298n	19
2.5	pictorial view of 12 volt DC supply	20
2.6	5 volt DC supply for raspberry pi 3b+	20
6.1	simulation result	54
6.2	application results	56

CHAPTER 1 INTRODUCTION

1.1 GENERAL INTRODUCTION

Traditional shopping experiences often involve tedious processes such as manual product selection, queuing at billing counters, and manual billing, leading to inefficiencies and customer dissatisfaction. To address these challenges, we present an automated smart shopping trolley system that leverages modern technology to enhance the shopping experience. By utilizing a Raspberry Pi 4 Model B and LM298N motor driver, along with web and mobile app interfaces for billing and product scanning, the system aims to revolutionize the shopping process. This paper outlines the methodology employed to design, develop, and implement the system, offering a novel approach to improving retail operations.

1.2 LITERATURE SURVEY

Prior research in the field of automated shopping systems has explored various technologies and methodologies to enhance the retail experience. Studies have investigated the use of RFID technology, computer vision, and mobile applications to automate product identification and billing processes. Additionally, research has focused on improving navigation within stores using localization techniques such as Bluetooth beacons and Wi-Fi positioning systems. By synthesizing findings from existing literature, this project builds upon previous work to create a comprehensive solution that addresses the shortcomings of traditional shopping methods.

Development of Smart Shopping Trolley using Arduino and Raspberry Pi" by A. C. Jadhav, S. D. Biradar, and S. A. Patil:

This paper discusses the development of a smart shopping trolley system using Arduino and Raspberry Pi. It covers the hardware and software architecture, including sensors for obstacle detection, RFID for product identification, and a camera module for image processing. The system enables autonomous navigation within a store environment, making shopping more convenient for customers.

"Design and Implementation of an Autonomous Shopping Cart" by H. Patel and S. Ranjan:

This project focuses on the design and implementation of an autonomous shopping cart using Arduino and Raspberry Pi. It includes features such as collision avoidance using ultrasonic sensors, object detection using RFID, and navigation using motor control. The paper discusses the integration of these components and presents experimental results demonstrating the effectiveness of the system.

"Autonomous Shopping Cart System with Raspberry Pi" by M. S. Chavan, S. D. Chougule, and P. S. Magdum:

This paper presents an autonomous shopping cart system built with Raspberry Pi. It describes the hardware components, including motor drivers, sensors, and a camera module. The system utilizes image processing techniques for obstacle detection and navigation. The authors discuss the challenges faced during development and propose solutions to improve the performance of the system.

"Smart Trolley System for Shopping Malls" by K. S. Saini, S. Yadav, and S. Kumar:

This project explores the development of a smart trolley system for shopping malls using Arduino and Raspberry Pi. It describes the integration of sensors for obstacle detection, RFID for product identification, and Wi-Fi communication for data exchange. The paper discusses the advantages of the proposed system, including enhanced shopping experience and reduced manual effort for customers.

1.3 SCOPE OF THE PROJECT

The scope of study for this project is to study and apply the suitable method for human tracking, obstacles avoid and interface the whole system with microcontroller. Automation is basically the delegation of human control function to technical equipment. Designing a user-friendly interface for customers to interact with the smart shopping trolley. Integrating RFID technology for automatic item recognition and inventory management. Implementing sensors to detect product weight, shelf location, and expiry dates. Developing a backend system for data processing, analytics, and personalized recommendations. Ensuring compatibility with existing

checkout systems and mobile devices for seamless communication. Conducting thorough testing to validate the trolley's performance and reliability in various environments. Incorporating security measures to protect customer data and prevent unauthorized access. Collaborating with retail partners to customize features and meet specific store requirements. Providing technical support and maintenance services to ensure uninterrupted operation.

Documenting the project scope, requirements, and outcomes for future reference and improvements.

1.4 ADVANTAGES OF PROPOSED TECHNIQUE

- ☐ Efficiency
- ☐ Personalization
- ☐ Inventory Management
- ☐ Payment Options
- ☐ Data Insights
- ☐ Accessibility
- ☐ Integration

1.5 APPLICATIONS OF PROPOSED TECHNIQUE

- ☐ Super Market.
- ☐ Automation Shopping Malls.

CHAPTER 2 PROJECT DESCRIPTION

2.1 BLOCK DIAGRAM OF SHOPPING TROLLEY

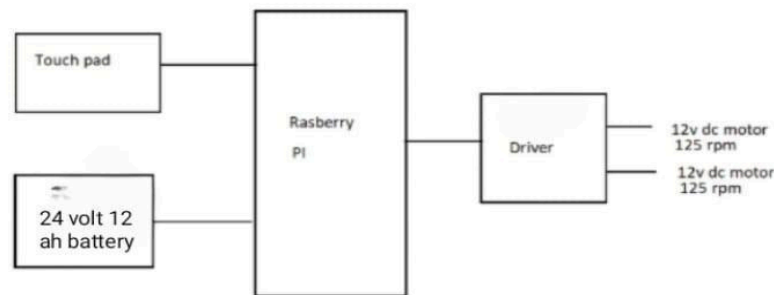


Fig 2.1 Block diagram of Shopping Trolley

The above figure 2.1 shows the general block diagram of *shopping trolley* respectively.

A block diagram is a visual representation of a system that shows its components as blocks or rectangles, connected by lines or arrows representing the flow of data or signals between them. In the context of a smart shopping trolley, a block diagram would illustrate the various components and their interconnections to create a functional system. Here's an explanation of the key components typically found in a block diagram for a smart shopping trolley:

Physical Components: These are the tangible parts of the smart shopping trolley, such as the cart itself, the display screen, input interfaces (like buttons or touchscreens), sensors, actuators, and any other hardware components.

Microcontroller or Processor: This is the central processing unit of the smart trolley system. It controls the operation of various components based on input from sensors and user interactions. The microcontroller processes data, executes algorithms, and communicates with other parts of the system.

User Interface: The user interface is the part of the system that allows interaction between the customer and the smart shopping trolley. It could include a display

screen for showing product information, buttons or touchscreens for input, and speakers for audio feedback.

Communication Modules: These modules enable communication between the smart shopping trolley and external systems, such as the store's inventory management system or mobile devices. Communication could occur via Wi-Fi, Bluetooth, or other wireless protocols.

Data Processing and Storage: This component handles the processing of data collected by sensors and user interactions. It may include a local database or memory for storing information about items in the cart, transaction history, and user preferences.

Power Supply: The power supply provides electrical energy to the various components of the smart trolley system, ensuring uninterrupted operation during use.

External Systems: These represent any external systems or services that interact with the smart shopping trolley, such as the store's point-of-sale system, inventory management software, or cloud-based services for data storage and analysis.

2.2 COMPONENTS

2.2.1 LIST OF HARDWARE COMPONENTS

1. RASPBERRY PI 3B+.
2. 12V DC MOTOR.
3. MOTOR DRIVER L298.
4. 12V DC POWER SUPPLY.
5. 5V DC POWER SUPPLY FOR RASPBERRY.
6. 7 ARDUINO UNO.
7. 8 BLUETOOTH MODEL.

2.2.2 HARDWARE COMPONENTS

1. Raspberry PI3b+

Raspberry Pi is a low priced, small sized computer that plugs into a computer monitor or TV, and uses normal peripheral like keyboard and mouse. It is a capable little device that enables people of all ages to explore computing, and to learn how to program in languages like Scratch and Python. It's capable of doing everything you'd expect a desktop computer to do, from browsing the internet and playing high-definition video, to making spreadsheets, word-processing, and playing games. The Raspberry Pi was first launched in 2012, and there have been a few changes and variations performed from that point forward. The first Pi had a single core 700MHz CPU and simply 256MB RAM, and the most recent model has a quad-core 1.4GHz CPU with 1GB RAM. All over the world, individuals use Raspberry Pi to get the skills of programming abilities, , do home automation, and even use them in modern applications. The Raspberry Pi works in the open source environment: it runs Linux (diverse distribution), and its principle supported working system, Raspbian, is open source and runs a suite of open source programming. The Raspberry Pi Foundation adds to the Linux part and Raspberry Pi is a low cost credit card size computer that plugs into a computer monitor or TV and uses a standard keyboard and mouse. Most importantly it's open source hardware. Computing Programmable Language like python and scratch under Linux platform different other open source extends just as releasing its very own lot programming as open source. Here Raspberry pi is being used as a main controller to derive other features like face recognition and detection which we are doing in our project.. Raspberry Pi 3 model B has CPU 1400MHZ quad-core ARM cortex-A53 processor. The Ethernet adaptor is connected to an additional USB port. In model A and A+ the USB port is connected directly to the Silicon on Chip (SoC).



Fig2.1 : Visual representation of RPI 3B

2. DC motor



Fig 2.2: Pictorial view of DC motor

A DC motor is any class of rotary electrical machines that converts direct current electrical energy into mechanical energy. The most common types rely on the forces produced by magnetic fields. Almost all types of DC motors have some internal mechanism, either electromechanical or electronic, which periodically changes the direction of current flow in part of the motor. A coil of wire with a current running through it generates an electromagnetic field aligned with the center of the coil. One

can change the direction and the magnitude of the magnetic field by changing the direction and magnitude of the current flowing through it. Nearly all types of DC motors have some internal mechanism, either electromechanical or electronic; to periodically change the direction of current flow in the part of DC motor. DC motors were the first type widely used, since they could be powered from existing direct current lighting power distribution systems. A DC motor's speed can be controlled over a wide range, using either a variable supply voltage or by changing the strength of current in its field windings. Small DC motors are used in tools, toys, and appliances.

2.3 L298N Motor Driver

L 298 is a dual full bridge driver that has a capability to bear high voltage as well as high current. It receives basic TTL (Transistor Transistor Logic) logic levels and is able to operate the different loads such as DC motors, stepper motors, relays etc. L-298 has two enable input to control any device by enabling or disabling it. L 298 IC is most commonly used to make motor drivers or motor controllers. These motor controllers can be controlled by any micro controller e.g Arduino, PIC, Raspberry Pi etc. hey receives input from micro controllers and operate the load attached to their output terminals correspondingly. L-298 motor driver (H-Bridge) is able to control two different DC motors simultaneously. While it can control a single stepper motor as well. L 298 has two Pulse Width Modulation (PWM) pins. PWM pins are used to control the speed of the motor. By changing the voltage signal's polarity at its input we can rotate the motor in either clockwise or counter clockwise direction. **L298** is a high current and high voltage IC. Its receives TTL logic signals and operates different loads like motors, solenoid, relays etc. It is mostly used in motor driver's designing. It has two specific pins for enabling or disabling the particular device attached at its output. Its features include low saturation voltage, over temperature protection.

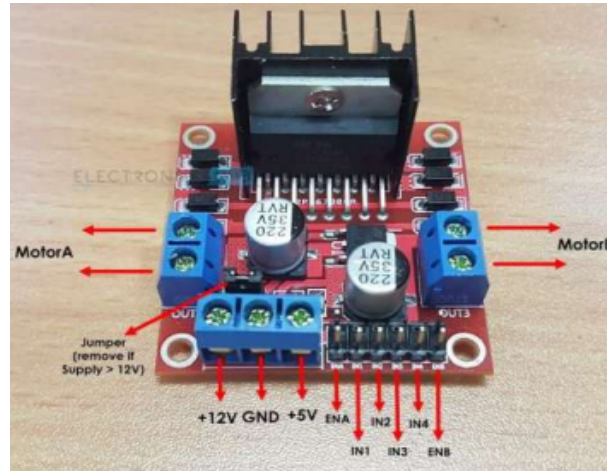


Fig 2.3: Schematic view of motor driver L298N

2.4 12V DC SUPPLY FOR MOTOR DRIVER

12V power supplies (or 12VDC power supplies) are one of the most common power supplies in use today. In general, a 12VDC output is obtained from a 120VAC or 240VAC input using a combination of transformers, diodes and transistors. We use a 1.5A 12V DC adapter in the project.



Fig 2.4: Pictorial view of 12V DC supply

2.5 5V DC SUPPLY FOR RASPBERRY PI3B+

Recommended and easiest way to power the Raspberry Pi is via the Micro USB port on the side of the unit. The recommended input voltage is 5V, and the recommended input current is 2A. A 5V 2.4A DC power supply is used in the project.



Fig 2.5: Pictorial view of 5V DC supply

2.6 Arduino UNO

Fig 2.6 Arduino UNO

- ❓ Microcontroller: ATmega328
- ❓ Operating Voltage: 5V
- ❓ Input Voltage (recommended): 7-12V
- ❓ Input Voltage (limits): 6-20V
- ❓ Digital I/O Pins: 14 (of which 6 provide PWM output)
- ❓ Analog Input Pins: 6
- ❓ Flash Memory: 32 KB
- ❓ Clock Speed: 16 MHz

2.2.3 SOFTWARE COMPONENTS

1. Arduino IDE



Fig 2.7 Arduino IDE

- ❓ The Arduino integrated development environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in the programming language Java.

- ❓ It includes a code editor with features such as text cutting and pasting, searching and replacing text, automatic indenting, brace matching, and syntax highlighting, and provides simple one click mechanisms to compile and upload programs to an Arduino board.

2.3 MODULES NAME

- ☐ Item Recognition Module.
- ☐ Inventory Management Module.
- ☐ User Interface Module.
- ☐ Communication Module.
- ☐ Sensor Integration Module.
- ☐ Data Processing Module.
- ☐ Power Management Module.
- ☐ Display and Interaction Module.
- ☐ Security and Authentication Module.
- ☐ Remote Control and Monitoring Module.

CHAPTER 3 METHODOLOGY

3.1 PROPOSED METHODOLOGY

The methodology for implementing the automated smart shopping trolley system involves several key steps:

Hardware Setup: Assemble the Raspberry Pi 4 Model B, LM298N motor driver, and two 12-volt DC motors onto the shopping trolley.

Software Development: Develop software for the Raspberry Pi to control motor movement, interface with the billing and scanning systems via web and mobile app, and manage overall system functionality.

Integration with Website and Mobile App: Create a user-friendly interface for customers to scan products using their smartphones and automatically add them to their virtual shopping carts. Integrate this interface with the billing system to generate invoices in real-time.

Testing and Optimization: Conduct extensive testing to ensure the system operates smoothly and efficiently. Optimize algorithms and hardware configurations to improve performance and reliability.

3.2 IMPLEMENTATION

Implementing a smart shopping trolley involves several steps and considerations to create a functional and user-friendly system. Here's an explanation of the implementation process:

Requirement Analysis: The first step is to gather requirements from stakeholders, including retailers, customers, and technology providers. Understanding their needs and expectations will guide the design and development process.

Design Phase: Based on the requirements, a detailed design of the smart shopping trolley system is created. This includes defining the hardware components, software

architecture, user interface design, and integration with existing systems.

Hardware Development: The hardware components of the smart shopping trolley, such as sensors, microcontrollers, displays, and communication modules, are selected and assembled according to the design specifications. Prototyping and testing may be conducted to ensure compatibility and functionality.

Software Development: Concurrently, the software for the smart shopping trolley is developed. This includes firmware for microcontrollers, applications for user interfaces, communication protocols, and backend systems for data processing and management. Testing is carried out at each stage to identify and address any issues.

Integration and Testing: Once the hardware and software components are developed, they are integrated to create the complete smart shopping trolley system. Integration testing is performed to verify that all components work together as intended and meet the requirements.

User Testing and Feedback: The smart shopping trolley prototype is tested by real users in a controlled environment, such as a retail store. Feedback from users is collected to identify usability issues, bugs, and areas for improvement.

Refinement and Iteration: Based on user feedback and testing results, refinements are made to the system design, hardware, and software. This iterative process continues until the smart shopping trolley meets the desired level of functionality, reliability, and user satisfaction.

Deployment: Once the system is deemed ready for deployment, it is installed in retail stores or shopping environments. Installation may involve training store staff on how to use and maintain the smart shopping trolleys.

Maintenance and Support: After deployment, ongoing maintenance and support are provided to ensure the smooth operation of the smart shopping trolleys. This includes software updates, troubleshooting, and addressing any issues that arise during regular use.

Monitoring and Optimization: Continuous monitoring of the smart shopping trolley system allows for optimization and improvement over time. Data analytics may be used to gather insights into customer behavior, product trends, and system performance, leading to further enhancements and refinements.

3.3 Application Framework

An application framework for a smart shopping trolley provides a structured approach to developing the software components that control and manage the functionalities of the trolley. It defines the architecture, design patterns, and reusable components that streamline the development process and ensure consistency and scalability. Here's an explanation of the application framework for a smart shopping trolley:

Modular Architecture: The framework is designed with a modular architecture, where different functionalities of the smart shopping trolley are divided into separate modules or components. This modular approach allows for easier maintenance, testing, and scalability of the system.

User Interface (UI) Framework: The framework includes a UI framework that provides tools and libraries for designing and implementing the user interface of the smart shopping trolley. This includes components for displaying product information, navigation menus, interactive buttons, and feedback mechanisms.

Communication Framework: A communication framework is integrated into the application framework to facilitate communication between the smart shopping trolley and external systems or devices. This includes protocols for data exchange, such as Wi-Fi, Bluetooth, RFID, or cellular communication.

Data Management Framework: The framework includes tools and libraries for managing data collected by the smart shopping trolley, such as product information, inventory data, user preferences, and transaction history. This includes databases, data storage mechanisms, and data processing algorithms.

Sensor Integration Framework: For trolleys equipped with sensors, the framework provides a sensor integration layer that allows for seamless integration of various sensor technologies, such as RFID readers, weight sensors, proximity sensors, and temperature sensors. This layer includes drivers, APIs, and utilities for interfacing with sensors and processing sensor data.

Event Handling and Control Logic: The framework includes mechanisms for handling user inputs, sensor events, and system events. This includes event-driven programming paradigms, state machines, and control logic for managing the behavior of the smart shopping trolley in response to different stimuli.

Security and Authentication: Security features are integrated into the framework to ensure the integrity and confidentiality of data collected and processed by the smart shopping trolley. This includes encryption mechanisms, authentication protocols, and access control mechanisms to protect sensitive information.

Testing and Debugging Tools: The framework includes tools and utilities for testing and debugging the smart shopping trolley software. This includes automated testing frameworks, simulation tools, and diagnostic utilities for identifying and fixing software bugs and issues.

Documentation and Guidelines: Comprehensive documentation and guidelines are provided as part of the framework to assist developers in understanding and using the framework effectively. This includes API documentation, coding standards, design patterns, and best practices for developing smart shopping trolley applications.

Community Support and Resources: Finally, the framework may include community support forums, online resources, and developer communities where developers can seek help, share knowledge, and collaborate on building smart shopping trolley applications.

Android Runtime

Android includes a collection of core libraries that has most of the practicality offered within the core libraries of the Java programming language. Every mechanical man application runs in its own method, with its own instance of the virtual machine referred to as Dalvik virtual machine. Dalvik has been written in order that a tool will run multiple VMs with efficiency. The Dalvik VM executes files within the Dalvik feasible (.dex) format that is optimized for lowest memory footprint. The VM is register-based, and runs categories compiled by a Java language compiler that are reworked into the .dex format by the enclosed "dx" tool. The Dalvik VM depends on

the Linux kernel for underlying practicality like threading and low-level memory management.

Linux Kernel

Android's core system services, such as security, memory management, process management, network stack, and driver model, are all based on Linux version 2.6. Between the hardware and the rest of the software stack.

ANDROID SDK

Rather than installing a new SDK starter kit, you can use the Android SDK and AVD Manager to upgrade to the current tools or framework
http://dl.google.com/android/android-sdk_r16-macosx.zip

Here's a rundown of the steps you'll need to take to install the Android SDK: Prepare your development computer and ensure it meets the system requirements.

Install the SDK starter package from the table above. (If you're on Windows, download the installer for help with the initial setup.)
you'll need to install the ADT Plugin for Eclipse.

To your SDK, Add Android platforms and other components. Examine the Android SDK's contents.

System Requirements

Supported Operating Systems

Windows XP (32-bit), Vista (32- or 64-bit), or Windows 7 (32- or 64-bit) Mac OS X

10.5.8 or later (x86 only)

Linux (tested on Ubuntu Linux, Lucid Lynx) GNU C Library (glibc) 2.7 or later is required.

On Ubuntu Linux, version 8.04 or later is required.

64-bit distributions must be capable of running 32-bit applications. For information about how to add support for 32-bit applications Supported Development

Environments for Android SDK

Eclipse IDE

Eclipse 3.6 (Helios) or greater

Note: Eclipse 3.5 (Galileo) is no longer supported with the latest version of ADT.

Eclipse JDT plugin (included in most Eclipse IDE packages)

If you need to install or update Eclipse, you can download it from

<http://www.eclipse.org/downloads/>.

Several types of Eclipse packages are available for each platform. For developing Android applications, we recommend that you install one of these packages:

Eclipse IDE for Java Developers Eclipse Classic

Eclipse IDE for Java EE Developers

JDK 5 or JDK 6 (JRE alone is not sufficient)

Android Development Tools plugin (recommended)

Not compatible with Gnu Compiler for Java (gcj) Other development environments or IDEs

JDK 5 or JDK 6 (JRE alone is not sufficient)

Apache Ant 1.8 or later

Not compatible with Gnu Compiler for Java (gcj)

INSTALLING THE SDK

Step 1. Preparing Your Development Computer

Before getting started with the Android SDK, take a moment to confirm that your development computer satisfies the system's requirements. If you don't already have it, you'll probably need to install the JDK. If you're going to build in Eclipse with the Android Development Tools (ADT) Plugin—which is the recommended route if you're new to Android—make sure you have the right version of Eclipse installed on your device, as mentioned in the System Requirements. document.

If you need to install Eclipse, you can download it from this location:

<http://www.eclipse.org/downloads/>

The "Eclipse Classic" version is recommended. Otherwise, a Java or RCP version of Eclipse is recommended.

Step 2. Downloading the SDK Starter Package

The SDK starter package is not a full development environment—it includes only the core SDK Tools, which you can use to download the rest of the SDK components (such as the latest Android platform).

If you haven't already, get the latest version of the SDK starter package from the [SDK download page](#).

If you downloaded a .zip or .tgz package (instead of the SDK installer), unpack it to a safe location on your machine. By default, the SDK files are unpacked into a directory named android-sdk-<machine-platform>.

If you downloaded the Windows installer (.exe file), run it now and it will check whether the proper Java SE Development Kit (JDK) is installed (installing it, if necessary), then install the SDK Tools into a default location (which you can modify).

Make a note of the name and location of the SDK directory on your system—you will need to refer to the SDK directory later, when setting up the ADT plugin and when using the SDK tools from the command line.

Step 3. Installing the ADT Plugin for Eclipse

Android offers a custom plugin for the Eclipse IDE, called Android Development Tools (ADT), that is designed to give you a powerful, integrated environment in which to build Android applications. It extends the capabilities of Eclipse to let you quickly set up new Android projects, create an application UI, debug your applications using the Android SDK tools, and even export signed (or unsigned) APKs in order to distribute your application. In general, developing in Eclipse with ADT is a highly recommended approach and is the fastest way to get started with Android.

If you'd like to use ADT for developing Android applications, install it now. Read [Installing the ADT Plugin](#) for step-by-step installation instructions, then return here to continue the last step in setting up your Android SDK.

If you prefer to work in a different IDE, you do not need to install Eclipse or ADT. Instead, you can directly use the SDK tools to build and debug your application. The [Introduction](#) to Android application development outlines the major steps that you need to complete when developing in Eclipse or other IDEs.

Step 4. Adding Platforms and Other Components

The last step in setting up your SDK is using the Android SDK and AVD Manager (a tool included in the SDK starter package) to download essential SDK components into your development environment.

The SDK uses a modular structure that separates the major parts of the SDK—Android platform versions, add-ons, tools, samples, and documentation—into a set of separately installable components. The SDK starter package, which you've already downloaded, includes only a single component: the latest version of the SDK Tools. To develop an Android application, you also need to download at least one Android platform and the associated platform tools. You can add other components and platforms as well, which is highly recommended.

If you used the Windows installer, when you complete the installation wizard, it will launch the Android SDK and AVD Manager with a default set of platforms and other components selected for you to install. Simply click Install to accept the recommended set of components and install them. You can then skip to Step 5, but we recommend you first read the section about the Available Components to better understand the components available from the Android SDK and AVD Manager.

You can launch the Android SDK and AVD Manager in one of the following ways:

From within Eclipse, select **Window > Android SDK and AVD Manager**.

On Windows, double-click the SDK Manager.exe file at the root of the Android SDK directory.

On Mac or Linux, open a terminal and navigate to the tools/ directory in the AndroidSDK, then execute:

Android

To download components, use the graphical UI of the Android SDK and AVD Manager to browse the SDK repository and select new or updated components (see figure 1). The Android SDK and AVD Manager installs the selected components in your SDK environment. For information about which components you should download, see Recommended Components.

VERSION HISTORY

mechanism has seen variety of updates since its original unleash, every fixing bugs and adding new options. every version is called, in alphabetical order, when a sweet. 2.3 cake refined the computer programme, improved the soft keyboard and copy/paste options, improved recreation performance, additional SIP support (VoIP calls), and additional support for close to Field Communication.

3.0 Honeycomb was a tablet-oriented unleash that supports larger screen devices and introduces several new computer programme options, support for multi-core processors, hardware acceleration for graphics and full system encoding. the primary device that includes this version, the Motorola Xoom pill, went on sale in Feb 2011.

3.1 Honeycomb, discharged in could 2011, additional support for further input devices, USB host mode for transferring info directly from cameras and alternative devices, and therefore the Google Movies and Books apps.

Honeycomb, discharged in July 2011, additional improvement for a broader vary of screen sizes, new "zoom-to-fill" screen compatibility mode, loading media files directly from Coyote State card, associated an extended screen support API. Huawei MediaPad is that the initial seven in. pill to use this version.

4.0 frozen dessert Sandwich, declared on Gregorian calendar month nineteen, 2011, brought Honeycomb options to smartphones and additional new options together with identity verification unlock, network information usage watching and management, unified social networking contacts, photography enhancements, offline email looking, app folders, and knowledge sharing exploitation NFC. Android 4.0.3 frozen dessert Sandwich is that the latest automaton version that's out there to phones. The ASCII text file of automaton four.0.1 was discharged on Gregorian calendar month one4, 2011.



[Prefer a video? We have one for you.](#)

Python Program

Python programs are composed of modules Modules contain statements

Statements contain expressions Expressions create and process objects

Features of Python

Open source: Python is publicly available open source software,any one can use source code that doesn't cost anything.

Easy-to-learn: Popular (scripting/extension) language, clear and easy syntax, no type declarations, automatic memory management, high-level data types and operations, design to read (more English like syntax) and write (shorter code compared to C, C++, and Java) fast.

High-levelLanguage:

High-level language (closer to human) refers to the higher level of concept from machine language (for example assembly languages). Python is an example of a high- level language like C, C++, Perl, and Java with low-level optimization.

Portable:

High level languages are portable, which means they are able to run across all major hardware and software platforms with few or no change in source code. Python is portable and can be used on Linux, Windows, Macintosh, Solaris, FreeBSD, OS/2, Amiga, AROS, AS/400 and many more.

Object-Oriented: Python is a full-featured object-oriented programming language, with features such as classes, inheritance, objects, and overloading.

Python has an interactive console where you get a Python prompt (command line) and interact with the interpreter directly to write and test your programs. This is useful for mathematical programming.

Interpreted : Python programs are interpreted, takes source code as input, and then compiles (to portable byte-code) each statement and executes it immediately. No need to compiling or linking.

Extendable : Python is often referred to as a "glue" language, meaning that it is capable to work in mixed-language environment. The Python interpreter is easily extended and can add a new built-in function or modules written in C/C++/Java code.

Libraries : Databases, web services, networking, numerical packages, graphical user interfaces, 3D graphics, others.

Supports :Support from online Python community

Python Interpreter

In interactive mode, type Python programs and the interpreter displays the result:

Type python into your terminal's command line

After a short message, the >>> symbol will appear

The above symbol signals the start of a Python interpreter's command line.

Python interpreter evaluates inputs (For example >>> 4*(6-2) return 16)

How stable is Python?

Very stable. New, stable releases are popping out roughly each half dozen to eighteen

months since 1991, and this appears seemingly to continue. presently there are sometimes around eighteen months between major releases. The latest stable releases will invariably be found on the Python transfer page. There are 2 counseled production-ready versions at this time in time, as a result of at the instant there are 2 branches of stable releases: a pair of .x and 3.x. Python 3.x could also be less helpful than a pair

of.x, since presently there's additional third party code accessible for Python a pair of

than for Python three. Python a pair of code can typically not run unchanged in Python three.

History

The name Python was elect from "Monty Pythons Flying Circus" that was a British sketch comedy series created by the comedy cluster Monty Python and broadcast by the BBC from 1969 to 1974. Python was created within the early Nineties by Guido van Rossum at the National analysis Institute for arithmetic and technology in Holland. Python was created as a successor of a language known as rudiment (All Basic Code) and discharged publically in1991. Guido remains Python's principal author, though it includes several contributions from active user community. Between 1991 and 2001 there area unit many versions discharged, current stable unharness is three.2. In 2001 the Python software system Foundation (PSF) was shaped, a non-profit organization created specifically to possess Python-related holding. Zope Corporation could be a sponsoring member of the terrorist organization. All most all Python releases area unit Open supply. to check the main points of unharness versions and licence agreement of Python check here.

Major uses of Python

System utilities (system admin tools, command line programs).

Web Development.

Graphical User Interfaces (Tkinter, gtk, Qt).

Internet scripting.

Embedded scripting.

Database access and programming.

Game programming.

Rapid prototyping and development.

Distributed programming

Organizations Using Python (sector wise)

Web Development : Yahoo Maps, Yahoo teams, Google, Zope Corporation, Ultraseek,

Linux Weekly News, ElasticHosts Cloud Servers, Mojam.com, hunch, Shopzilla, Movieplayer.it, Multiplayer.it. Games: field of battle a pair of, Crystal area, Star Trek Bridge Commander, The Temple of Elemental Evil, Vampire: The Masquerade: Bloodlines, Civilization four, QuArK (Quake Army Knife) Graphics : Industrial light-weight & Magic, filmmaker Feature Animation, HKS, Inc. (ABAQUS/CAE), RoboFog, Caligari Corporation, Blender 3D, Jasc code, Paint look professional. Financial :Altis Investment Management, ABN AMRO Bank, Treasury Systems, Bellco bank, Journyx Timesheet and Resource Management code. Science : National Weather Service, measuring device Remote Sensing cluster, Applied Maths, Biosoft, The National analysis Council of Canada, town National Laboratory (LANL) Theoretical Physics Division, AlphaGene, Inc., LLNL, NASA, Swedish earth science and Hydrological Institute (SMHI), Environmental Systems analysis Institute (ESRI), Objexx Engineering,

Nmag procedure Micromagnetics Electronic style Automation: Ciranova, Productivity style Tools, Object Domain, Pardus, Red Hat, SGI, Inc., MCI Worldcom, Nokia, Education : University of Calif., Irvine, Smeal school of Business, The Pennsylvania State University, New island Digital Library, IT Certification examination preparation, SchoolTool, Business code : Raven Bear Systems Corporation, Thawte Consulting, Advanced Management Solutions opposition., IBM, Arakn, RealNetworks, dSPACE, Escom, the little Company, Nexedi, Piensa Technologies - Bufete Consultor First State United Mexican States, Nektra, WuBook.

Is Python a good language for beginning programmers?

Yes. it's still common to start out students with a procedural and statically written

language like Pascal, C, or a set of C++ or Java. Students could also be higher served by learning Python as their natural language. Python contains a terribly straightforward and consistent syntax and an outsized normal library and, most significantly, victimization Python in a very starting programming course lets students focus on vital programming skills like drawback decomposition and knowledge sort style. With Python, students may be quickly introduced to basic ideas like loops and procedures. they will in all probability even work with user-defined objects in their terribly 1st course. For a

student World Health Organization has ne'er programmed before, employing a statically written language appears unnatural. It presents extra complexness that the scholar should master and slows the pace of the course. the scholars are attempting to be told to assume sort of a pc, decompose issues, style consistent interfaces, and encapsulate knowledge. whereas learning to use a statically written language is very important within the future, it's not essentially the most effective topic to deal with within the students' 1st programming course. Many different aspects of Python build it an honest natural language. Like Java, Python contains a massive normal library so students may be assigned programming comes terribly early within the course that do one thing.

Assignments aren't restricted to the quality four-function calculator and check equalization programs. By victimization the quality library, students will gain the satisfaction of acting on realistic applications as they learn the basics of programming. victimization the quality library additionally teaches students regarding code apply.

Third- party modules like PyGame are useful in extending the students' reach.

CHAPTER 4 SIMULATION THEORY

4.1 PROTEUS DESCRIPTION

Proteus is a famous electronic layout automation (EDA) software program that offers a platform for designing, simulating, and analyzing electronic circuits. It is utilized by engineers, hobbyists, and students to create schematic diagrams, PCB layouts, and three-D fashions of digital circuits.

Proteus gives a wide range of capabilities and tools for electronic layout, such as:

Schematic Capture: Proteus lets in customers to create and edit schematic diagrams of digital circuits the usage of a drag-and-drop interface. It helps numerous additives, along with resistors, capacitors, transistors, ICs, and microcontrollers.

Simulation: Proteus provides a simulation environment that permits users to test and analyze the behavior of digital circuits before constructing them. It supports diverse simulation models, which includes SPICE, VHDL, and Verilog.

PCB Layout: Proteus lets in users to create PCB layouts of digital circuits the use of a graphical interface. It supports numerous PCB layout rules, which includes trace width, clearance, and thru length.

3-d Modeling: Proteus offers a 3-D modeling environment that allows users to visualize and examine the bodily properties of digital circuits. It helps various 3-D fashions, inclusive of enclosures, connectors, and cables.

Library: Proteus consists of a huge library of electronic additives, such as resistors, capacitors, transistors, ICs, and microcontrollers, that may be utilized in electronic designs.

Proteus is widely used in various industries, inclusive of car, aerospace, scientific, and purchaser electronics, for designing and simulating electronic circuits. It is to be had in diverse editions, together with Proteus 8 Professional, Proteus 8 Lite, and Proteus eight SCH, that cater to exclusive design wishes and budgets.

Simulation is the process of creating a digital version of a device or a system and jogging experiments on it to analyze its conduct and overall performance. In the context of electronic circuits, simulation refers back to the procedure of making a digital model of an digital circuit and going for walks experiments on it to analyze its electric conduct and performance.

4.2 RASPBERRY PI SIMULATION

Simulating a smart shopping trolley using Raspberry Pi involves leveraging its capabilities as a miniature computer to emulate the functionalities of the trolley's hardware and software components. Here's how you can simulate a smart shopping trolley using Raspberry Pi:

Setup Raspberry Pi: Begin by setting up your Raspberry Pi board. Install the operating system (such as Raspbian or Raspberry Pi OS) and ensure it's connected to the internet.

Emulate Hardware Components: Use external hardware components or simulate them using software libraries to mimic the sensors, user interface, communication modules, and other hardware elements of the smart shopping trolley. For example:

Sensors: Simulate RFID readers, weight sensors, and other sensors using GPIO pins, or connect external sensors compatible with Raspberry Pi.

User Interface: Use a display screen (such as the official Raspberry Pi touchscreen) or simulate a graphical user interface (GUI) using Python libraries like Tkinter.

Communication Modules: Simulate communication with external systems using network protocols like Wi-Fi or Bluetooth, or use emulators to mimic interactions with external devices.

Actuators: If applicable, simulate motor control or other physical actions using GPIO pins or external actuators.

Develop Software Logic: Write the software logic to control and manage the simulated hardware components. Use Python or other programming languages supported by Raspberry Pi to develop the software. Break down the functionalities into modules and classes, following best practices for software design and

architecture.

Implement Functionalities: Implement the functionalities expected from a smart shopping trolley, such as:

Item recognition using RFID or barcode scanning.

Inventory management and updating.

User interface for displaying product information, promotions, and navigation.

Communication with external systems, such as store databases or checkout terminals.

Integration with cloud services for data storage or analytics.

Testing and Debugging: Test the simulated smart shopping trolley software on the Raspberry Pi. Debug any issues and refine the software logic as needed. Use debugging tools and logging to identify and troubleshoot errors.

User Interaction Simulation: Emulate user interactions with the smart shopping trolley software to simulate real-world scenarios. This could involve simulating user inputs through the GUI, or programmatically triggering events to simulate sensor readings or external communications.

Iterative Development: Iterate on the software design and implementation based on testing results and feedback. Refine the functionalities, improve performance, and enhance user experience as needed.

CHAPTER 5

Software program developed for the proposed project

5.1 Automatic speed control arduino code

```
int motor1pin1=2;
```

```
int motor1pin2=3;
```

```
int motor2pin1=4;
```

```
int motor2pin2=5;
```

```
char choice="", oldchoice="";
```

```
void setup()
```

```
{
```

```
Serial.begin(9600);
```

```
pinMode(motor1 pin1, OUTPUT);
```

```
pinMode(motor1 pin2,OUTPUT); pinMode(motor2pin 1,OUTPUT);
```

```
pinMode(motor2pin2,OUTPUT);
```

```
}
```

```
void loop()
```

```
{

if (Serial.available()>0)

{

Serial.println("Connected");

choice = Serial.read();

Serial.println(choice);

if(choice=='a')

{

if(oldchoice=="")

{

forward(); Serial.println("Forward"); delay(2000); stopp(); Serial.println("Stop");

}

else if(oldchoice=='a')

{

stopp();
```

```
Serial.println("Stop");
```

```
delay(1000);
```

```
}
```

```
else if(oldchoice=='b')
```

```
{ reverse(); Serial.println("Reverse"); delay(2000); stopp();
```

```
}
```

```
Serial.println("Stop");else if(oldchoice=='c')
```

```
{ reverse(); Serial.println("Stop"); delay(4000); stopp(); Serial.println("Stop"); }
```

```
else if(oldchoice=='d')
```

```
{ reverse(); Serial.println("Stop"); delay(6000); stopp(); Serial.println("Stop");
```

```
} oldchoice = choice;
```

```
}
```

```
else if(choice=='b')
```

```
{
```

```
if(oldchoice=="")
```

```
{ forward(); Serial.println("Stationary"); delay(4000); stopp(); Serial.println("Stop");
```

```
} else if(oldchoice=='a')

{

forward(); Serial.println("Forward"); delay(2000); stopp(); Serial.println("Stop");

} else if (oldchoice=='b')

{ stopp(); Serial.println("Stop"); delay(1000);

} else if(oldchoice=='c')

{

reverse();

Serial.println("Reverse");

delay(2000);

stopp();

Serial.println("Stop");

}

else if(oldchoice=='d')

{ reverse(); Serial.println("Reverse"); delay(4000); stopp(); Serial.println("Stop");

} oldchoice = choice;
```

```
}
```

```
else if(choice=='c')
```

```
{
```

```
if(oldchoice=="")
```

```
{ forward(); Serial.println("Fruits & Vegetables"); delay(6000); stopp();  
Serial.println("Stop");
```

```
}
```

```
else if(oldchoice=='a')
```

```
{ forward(); Serial.println("Forward"); delay(4000); stopp(); Serial.println("Stop");
```

```
}
```

```
else if(oldchoice=='b')
```

```
{ forward(); Serial.println("Forward"); delay(2000); stopp();
```

```
Serial.println("Stop");
```

```
}
```

```
else if(oldchoice=='c')
```

```
{
```

```
stopp();
```

```
Serial.println("Stop");
```

```
}
```

```
delay(1000);
```

```
else if(oldchoice=='d')
```

```
{
```

```
reverse();
```

```
Serial.println("Reverse"); delay(2000); stopp(); Serial.println("Stop"); }
```

```
}
```

```
oldchoice = choice;
```

```
else if(choice=='d')
```

```
{
```

```
if(oldchoice=="")
```

```
{ forward(); Serial.println("Cosmetics"); delay(8000); stopp(); Serial.println("Stop"); }
```

```
else if(oldchoice=='a')
```

```
{ forward(); Serial.println("Forward"); delay(6000); stopp(); Serial.println("Stop");  
  
}
```

```
else if(oldchoice=='b')
```

```
{ forward(); Serial.println("Forward"); delay(4000); stopp(); Serial.println("Stop");  
  
}
```

```
else if(oldchoice=='c')
```

```
{ forward(); Serial.println("Forward"); delay(2000); stopp();
```

```
Serial.println("Stop");
```

```
}
```

```
else if(oldchoice=='d')
```

```
{
```

```
stopp();
```

```
Serial.println("Stop");
```

```
}
```

```
delay(1000);
```

```
oldchoice = choice;
```

```
}
```

```
}
```

```
}
```

```
void forward()
```

```
{    digitalWrite(motor1    pin1,    HIGH);    digitalWrite(motor1    pin2,    LOW);  
digitalWrite(motor2pin1, HIGH); digitalWrite(motor2pin2,LOW); }
```

```
void reverse()
```

```
{    digitalWrite(motor1    pin    1,LOW);    digitalWrite(motor1    pin2,    HIGH);  
digitalWrite(motor2pin1,LOW); digitalWrite(motor2pin2, HIGH); }
```

```
void stopp({    digitalWrite(motor1    pin1,LOW);    digitalWrite(motor1    pin2,    LOW);  
digitalWrite(motor2pin1,LOW); digitalWrite(motopin2,LOW); }
```

5.2 Automatic speed control based raspberry pi code

```
import RPi.GPIO as GPIO
```

```
import time
```

```
import sys
```

```
# Define GPIO pins
```

```
motor1_pin1 = 2
```

```
motor1_pin2 = 3
```

```
motor2_pin1 = 4
```

```
motor2_pin2 = 5
```

```

# Set GPIO mode
GPIO.setmode(GPIO.BCM)
GPIO.setup(motor1_pin1, GPIO.OUT)
GPIO.setup(motor1_pin2, GPIO.OUT)
GPIO.setup(motor2_pin1, GPIO.OUT)
GPIO.setup(motor2_pin2, GPIO.OUT)

def forward():
    GPIO.output(motor1_pin1, GPIO.HIGH)
    GPIO.output(motor1_pin2, GPIO.LOW)
    GPIO.output(motor2_pin1, GPIO.HIGH)
    GPIO.output(motor2_pin2, GPIO.LOW)

def reverse():
    GPIO.output(motor1_pin1, GPIO.LOW)
    GPIO.output(motor1_pin2, GPIO.HIGH)
    GPIO.output(motor2_pin1, GPIO.LOW)
    GPIO.output(motor2_pin2, GPIO.HIGH)

def stopp():
    GPIO.output(motor1_pin1, GPIO.LOW)
    GPIO.output(motor1_pin2, GPIO.LOW)
    GPIO.output(motor2_pin1, GPIO.LOW)
    GPIO.output(motor2_pin2, GPIO.LOW)

def main():
    old_choice = ""
    try:
        print("Connected")
        while True:
            choice = sys.stdin.readline().strip() # Read input from stdin

```

```
print(choice)
if choice == 'a':
    if old_choice == "":
        forward()
        print("Forward")
        time.sleep(2)
        stopp()
        print("Stop")
    elif old_choice == 'a':
        stopp()
        print("Stop")
        time.sleep(1)
    elif old_choice == 'b':
        reverse()
        print("Reverse")
        time.sleep(2)
        stopp()
        print("Stop")
    elif old_choice == 'c':
        reverse()
        print("Stop")
        time.sleep(4)
        stopp()
        print("Stop")
    elif old_choice == 'd':
        reverse()
        print("Stop")
        time.sleep(6)
        stopp()
        print("Stop")
    old_choice = choice
```

```
elif choice == 'b':
    if old_choice == "":
        forward()
        print("Stationary")
        time.sleep(4)
        stopp()
        print("Stop")
    elif old_choice == 'a':
        forward()
        print("Forward")
        time.sleep(2)
        stopp()
        print("Stop")
    elif old_choice == 'b':
        stopp()
        print("Stop")
        time.sleep(1)
    elif old_choice == 'c':
        reverse()
        print("Reverse")
        time.sleep(2)
        stopp()
        print("Stop")
    elif old_choice == 'd':
        reverse()
        print("Reverse")
        time.sleep(4)
        stopp()
        print("Stop")
    old_choice = choice
```

```
elif choice == 'c':
    if old_choice == "":
        forward()
        print("Fruits & Vegetables")
        time.sleep(6)
        stopp()
        print("Stop")
    elif old_choice == 'a':
        forward()
        print("Forward")
        time.sleep(4)
        stopp()
        print("Stop")
    elif old_choice == 'b':
        forward()
        print("Forward")
        time.sleep(2)
        stopp()
        print("Stop")
    elif old_choice == 'c':
        stopp()
        print("Stop")
        time.sleep(1)
    elif old_choice == 'd':
        reverse()
        print("Reverse")
        time.sleep(2)
        stopp()
        print("Stop")
    old_choice = choice
```

```
elif choice == 'd':
    if old_choice == "":
        forward()
        print("Cosmetics")
        time.sleep(8)
        stopp()
        print("Stop")
    elif old_choice == 'a':
        forward()
        print("Forward")
        time.sleep(6)
        stopp()
        print("Stop")
    elif old_choice == 'b':
        forward()
        print("Forward")
        time.sleep(4)
        stopp()
        print("Stop")
    elif old_choice == 'c':
        forward()
        print("Forward")
        time.sleep(2)
        stopp()
        print("Stop")
    elif old_choice == 'd':
        stopp()
        print("Stop")
        time.sleep(1)
    old_choice = choice
```

```
except KeyboardInterrupt:
    print("\nExiting...")
finally:
    GPIO.cleanup()

if __name__ == "__main__":
    main()
```

CHAPTER 6

SIMULATION RESULT AND HARDWARE RESULT

6.1 SIMULATION RESULT

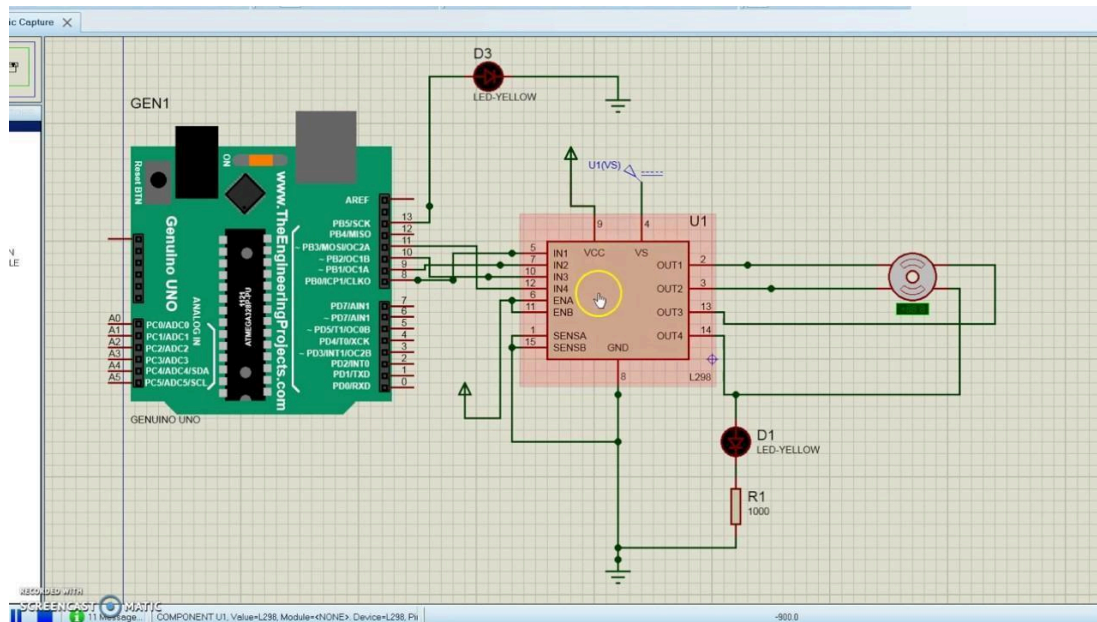


Fig 6.1 Simulation Result

In a smart shopping trolley simulation, the results would reflect how well the simulated system performs in emulating the functionalities of a real smart shopping trolley. Here's an explanation of what the simulation results might entail:

Functionality Testing: The simulation results would indicate whether the various functionalities of the smart shopping trolley, such as item recognition, inventory management, user interface interaction, and communication with external systems, are working as intended. This includes verifying that sensors accurately detect items, the user interface displays information correctly, and communication with external systems is successful.

Accuracy of Item Recognition: One crucial aspect of the simulation is assessing the accuracy of item recognition. The results would indicate whether the simulated system can correctly identify items using RFID, barcode scanning, or other methods. Any discrepancies or errors in item recognition would need to be addressed to ensure the reliability of the system.

Inventory Management: The simulation results would provide insights into how effectively the system manages inventory, including adding and removing items from

the cart, updating inventory counts, and generating alerts for out-of-stock items or expired products. This helps evaluate the efficiency of inventory management processes simulated by the system.

User Interface Interaction: User interface simulation results would evaluate the usability and effectiveness of the graphical interface presented to users. This includes assessing the clarity of product information, ease of navigation through menus, and responsiveness to user inputs. Positive results indicate that the simulated user interface enhances the shopping experience and facilitates interaction with the smart shopping trolley.

Communication with External Systems: The simulation results would assess the system's ability to communicate with external systems, such as store databases, checkout terminals, or cloud services. This involves verifying data exchange protocols, transmission reliability, and error handling mechanisms. Successful communication ensures seamless integration with existing retail infrastructure and data management systems.

Performance Metrics: The simulation results may include performance metrics such as response time, processing speed, and resource utilization. These metrics help evaluate the efficiency and scalability of the simulated system, identifying any bottlenecks or areas for optimization.

6.2 APPLICATION RESULT

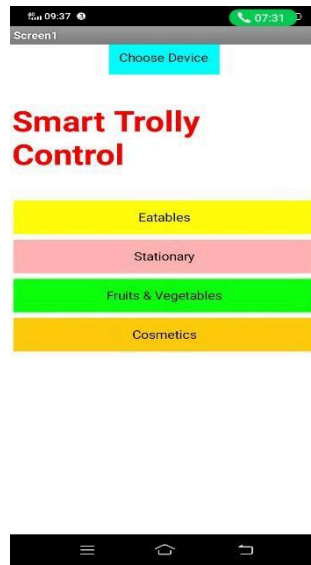


Fig 6.2 Application Result

The application results of a smart shopping trolley refer to the performance and functionality of the software application that controls and manages the trolley's operations. Here's how the application results might be explained:

Item Recognition Accuracy: The application results would assess the accuracy of item recognition functionalities within the smart shopping trolley application. This involves evaluating whether items are correctly identified through RFID scanning, barcode scanning, or other recognition methods. Higher accuracy indicates a more reliable system for tracking items in the shopping cart.

Inventory Management Efficiency: Application results would indicate how effectively the smart shopping trolley application manages inventory. This includes tracking the addition and removal of items from the cart, updating inventory counts in real-time, and generating alerts for out-of-stock items or expired products. Efficient inventory management ensures accurate stock levels and reduces the likelihood of product shortages or wastage.

User Interface Usability: The application results would evaluate the usability and user experience of the smart shopping trolley application's graphical user interface (GUI). This involves assessing the clarity of product information, ease of navigation

through menus, responsiveness to user inputs (e.g., touch gestures or button presses), and overall intuitiveness of the interface. Positive results indicate that the application enhances the shopping experience for users.

Communication Reliability: Application results would measure the reliability of communication between the smart shopping trolley application and external systems or devices. This includes assessing the stability of wireless connections (e.g., Wi-Fi, Bluetooth), the speed of data exchange, and the ability to integrate with store databases or checkout terminals. Reliable communication ensures seamless integration with existing retail infrastructure and facilitates efficient data exchange.

Error Handling and Recovery: The application results would indicate how well the smart shopping trolley application handles errors and exceptions. This involves testing error detection mechanisms, error recovery procedures, and user notification systems for informing customers or store staff about issues encountered during operation. Effective error handling minimizes disruptions to the shopping experience and ensures smooth operation of the trolley.

Performance Metrics: Application results may include performance metrics such as response time, processing speed, and memory usage. This helps evaluate the efficiency and scalability of the smart shopping trolley application, identifying any bottlenecks or areas for optimization. Improved performance leads to faster and more responsive interactions with the application.

User Feedback and Satisfaction: If user feedback is incorporated into the evaluation, the application results would reflect users' experiences and satisfaction levels with the smart shopping trolley application. This qualitative data provides valuable insights into user preferences, usability issues, and areas for improvement, guiding further refinements to the application.

6.3 HARDWARE RESULT

The hardware results of a smart shopping trolley refer to the performance and functionality of the physical components comprising the trolley. Here's how the

hardware results might be explained:

User Interface Responsiveness: Hardware results would indicate how responsive and user-friendly the physical user interface components are. This includes assessing the display screen's clarity and resolution, the responsiveness of touchscreens or buttons, and the visibility of indicators or feedback mechanisms. Positive results indicate that users can interact with the smart shopping trolley easily and intuitively.

Communication Reliability: The hardware results would evaluate the reliability of communication modules integrated into the trolley, such as Wi-Fi, Bluetooth, or RFID transceivers. This involves assessing the stability of wireless connections, the range of communication, and the ability to exchange data with external systems or devices. Reliable communication ensures seamless integration with backend systems and external services.

Durability and Robustness: Hardware results would include assessments of the physical durability and robustness of the smart shopping trolley components. This involves testing the structural integrity of the cart frame, the resilience of electronic components to mechanical stress or environmental factors, and the longevity of power sources such as batteries or power adapters. A robust hardware design ensures the trolley can withstand the rigors of everyday use in a retail environment.

Power Efficiency: Hardware results would measure the power consumption and efficiency of the smart shopping trolley components. This includes assessing the energy consumption of sensors, microcontrollers, displays, and communication modules, as well as the effectiveness of power management mechanisms such as sleep modes or low-power states. Efficient power management prolongs battery life and reduces operating costs for retailers.

Integration and Compatibility: Hardware results would evaluate how well the smart shopping trolley integrates with existing retail infrastructure and systems. This involves testing compatibility with checkout terminals, inventory management systems, and store networks, as well as assessing ease of installation and setup. Seamless integration minimizes disruptions to store operations and facilitates adoption of the smart shopping trolley technology.

Overall Performance: The hardware results would provide an overall assessment of

the smart shopping trolley's performance.

CHAPT

ER 6

CONCLUSION &

FUTURE SCOPE

6.1 CONCLUSION

The implementation of an automated smart shopping trolley system demonstrates significant potential for improving the efficiency and convenience of retail operations. By integrating automatic billing, product scanning, and motorized trolley movement, the system streamlines the shopping experience and reduces waiting times at checkout counters. Through rigorous testing and optimization, the system achieves reliable performance and user satisfaction. Future research could explore additional features such as inventory management and personalized recommendations to further enhance the shopping experience.

6.2 FUTURE SCOPE

The future scope of smart shopping trolleys holds immense potential for further innovation and enhancement of the shopping experience. Here are some possibilities for future developments in this technology:

Advanced AI and Machine Learning Integration: Incorporating advanced artificial intelligence (AI) and machine learning algorithms can enhance the capabilities of smart shopping trolleys. AI-powered recommendation systems can provide personalized product suggestions based on customer preferences, purchase history, and real-time data analysis.

Autonomous Navigation and Mobility: Future smart shopping trolleys could feature autonomous navigation capabilities, allowing them to navigate through stores independently. Utilizing technologies such as computer vision, LiDAR, and SLAM (Simultaneous Localization and Mapping), these trolleys can navigate aisles, avoid obstacles, and optimize routes for efficient shopping experiences.

Augmented Reality (AR) Shopping Assistance: Integrating augmented reality (AR) technology into smart shopping trolleys can provide shoppers with immersive shopping experiences. AR overlays can display product information, promotions, and navigation guidance directly onto the shopper's field of view, enhancing product visualization and decision-making.

Blockchain for Supply Chain Transparency: Implementing blockchain technology can enhance transparency and traceability in the supply chain. Smart shopping trolleys equipped with blockchain-enabled systems can provide shoppers with real-time information about product origins, certifications, and sustainability credentials, empowering them to make informed purchasing decisions.

Integration with Mobile Payment Solutions: Future smart shopping trolleys can seamlessly integrate with mobile payment solutions, enabling customers to complete transactions directly from the trolley. Integration with digital wallets, NFC (Near Field Communication) technology, and contactless payment methods enhances convenience and accelerates checkout processes.

Environmental Sustainability Initiatives: Smart shopping trolleys can play a role in

promoting environmental sustainability by encouraging eco-friendly shopping practices. Features such as energy-efficient designs, reusable shopping bag dispensers, and prompts for selecting sustainable products can contribute to reducing carbon footprints and promoting responsible consumption.

Health and Wellness Monitoring: Smart shopping trolleys can incorporate health and wellness monitoring features to promote healthier lifestyles. Integrated sensors and AI algorithms can analyze shoppers' dietary preferences, allergies, and nutritional requirements, providing personalized recommendations for healthier food choices and meal planning.

Integration with Smart Home Ecosystems: Seamless integration with smart home ecosystems enables shoppers to extend their shopping experiences beyond the store. Smart shopping trolleys can sync shopping lists, preferences, and purchase histories with home automation systems, facilitating automated replenishment, meal preparation, and pantry management.

Social Integration and Community Engagement: Integrating social media platforms and community engagement features into smart shopping trolleys fosters social interaction and community building among shoppers. Features such as sharing shopping lists, product reviews, and recommendations with friends and family enhance the social aspect of shopping experiences.

Continuous Innovation and Collaboration: The future scope of smart shopping trolleys relies on continuous innovation and collaboration among technology providers, retailers, and consumers. By leveraging emerging technologies, consumer insights, and industry partnerships, smart shopping trolleys can evolve to meet evolving consumer preferences, market trends, and retail demands.

REFERENCES

[1] Ambekar K, Dhole V, Sharma S, Wadekar T. (2015). Smart shopping trolley using

RFID. International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) 4(10): 3875-3877.

[2] Balaji S, Balamurugan S, Marimuthu R. (2017). Smart shopping cart. IEEE Internet of Things Journal.

[3] Bedi H, Goyal N, Kumar S, Gupta A. (2017). Smart trolley using Smart phone and Arduino. Journal of Electrical & Electronic Systems 2(12): 6.
<https://doi.org/10.4172/2332-0796.1000223>

[4] Chandrasekar P, Sangeetha T. (2014). Smart shopping cart with automatic billing system through RFID and Zigbee. Information Communication and Embedded Systems (ICICES 2014), pp. 1–4. <https://doi.org/10.1109/ICICES.2014.7033996>.

[5] Dawkhar K, Dhomase S, Mahabaleshwarkar S. (2015). Electronic shopping cart for effective shopping based on RFID. International Journal of Innovative Research in Electrical, Electronic, Instrumentation and Control Engineering 3(1): 84-86.
<https://doi.org/10.17148/IJIREEICE.2015.3117>

[6] Gade A, Bhatt N, Thakare N. (2018). Survey on energy efficient cloud: A novel approach towards green computing. Helix 5(5): 3976-3979.
<https://doi.org/10.29042/2018-3976-3979>.

[7] Karpagam V, Balapriya S, Kalairubini G, Kalaivani A. (2017). Smart trolley with smart billing. International Journal of Computer Systems 4(3): 55-58.
<https://doi.org/10.15680/IJIRCCE.2015.0310116>

[8] Machike K, Golait M, Rathod R, Petkar R, Goche P. (2017). A new technology of smart trolley using RFID and ZIGBEE. International Journal on Recent and Innovation Trends in Computing and Communication 5(2): 256-259.

[9] Prasad JS, Kumar BOP, Roopa D, Arjun AK. (2011). A novel low-cost intelligent shopping cart. IEEE 2nd International Conference on Networked Embedded Systems for Enterprise Applications, pp. 1-4.

[10] Rupali Sawant, Kripa Krishnan, Shweta Bhokre, Priyanka Bhosale (2015). The RFID based smart shopping cart. International Journal of Engineering Research and General Science 3(2): 275-280.

REFERENCES

- [1] Machike K, Golait M, Rathod R, Petkar R, Goche P. (2017). A new technology of smart trolley using RFID and ZIGBEE. *International Journal on Recent and Innovation Trends in Computing and Communication* 5(2): 256-259. [2] Thiagarajan M, Aejaz M, Kumar M. (2017). RFID based advanced trolley for super market. *Special Issue* 8. [3] Prasad JS, Kumar BOP, Roopa D, Arjun AK. (2011). A novel low-cost intelligent shopping cart. *IEEE 2nd International Conference on Networked Embedded Systems for Enterprise Applications*, pp. 1-4. [4] Karpagam V, Balapriya S, Kalairubini G, Kalaivani A. (2017). Smart trolley with smart billing. *International Journal of Computer Systems* 4(3): 55-58. [5] Gade A, Bhatt N, Thakare N. (2018). Survey on energy efficient cloud: A novel approach towards green computing. *Helix* 5(5): 3976-3979. <https://doi.org/10.29042/2018-3976-3979> [6] Chandrasekar P, Sangeetha T. (2014). Smart shopping cart with automatic billing system through RFID and Zigbee. *Information Communication and Embedded Systems (ICICES 2014)*, pp. 1-4. <https://doi.org/10.1109/ICICES.2014.7033996> [7] Ms. Rupali Sawant, Kripa Krishnan, Shweta Bhokre, Priyanka Bhosale (2015). The RFID based smart shopping cart. *International Journal of Engineering Research and General Science* 3(2): 275-280. [8] Dawkhar K, Dhomase S, Mahabaleshwarkar S. (2015). Electronic shopping cart for effective shopping based on RFID. *International Journal of Innovative Research in Electrical, Electronic, Instrumentation and Control Engineering* 3(1): 84-86. <https://doi.org/10.17148/IJIREEICE.2015.3117> [9] Ambekar K, Dhole V, Sharma S, Wadekar T. (2015). Smart shopping trolley using RFID. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)* 4(10): 3875-3877. [10] Shelke CJ, Karde P, Thakre VM. (2015). Study of various perspectives of android security. *International Journal of Innovative Research in Computer and Communication Engineering* 3(10): 9667-9672. <https://doi.org/10.15680/IJIRCCE.2015.0310116> [11] Balaji S, Balamurugan S, Marimuthu R. (2017). Smart shopping cart. *IEEE Internet of Things Journal*. [12] Bedi H, Goyal N, Kumar S, Gupta A. (2017). Smart trolley using Smart phone and Arduino. *Journal of Electrical & Electronic Systems* 2(12):

6. <https://doi.org/10.4172/2332-0796.1000223> [13] Thiyagarajan M, Aejaaz M, Kumar M. (2017). RFID based advanced shopping trolley for super market. Research Gate Journal 8. [14] Berdaliyev Y, James AP. (2016). Smart shopping cart using Zigbee department of electrical and electronic engineering school of engineering. Nazarbayev University Astana, Kazakhstan. [15] Saad SS, Nakad ZS. (2011). A standalone RFID indoor positioning system using passive tags. IEEE Transactions on Industrial Electronics 58(5): 1961-1970. <https://doi.org/10.1109/TIE.2010.2055774> [16] Chandra Babu DVS. (2012). Wireless intelligent billing trolley for supermarket. International Journal of Advanced Research in Technology 3(1). [17] Yewatkara A, Inamdarb F, Singh R, Bandale AA. (2018). Smart billing trolley via application. International Journal of Advance Research in Engineering, Science & Technology 5(3). [18] Larsan Aro Brian A, Arockiam L, Sheba Kezia Malarchelvi PD. (2014). An IOT based secured smart library system with NFC based book tracking. International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE) 11(5): 1821. [19] Priyanka S. Sahare, Anup Gade, Jayant Rohankar (2019). A Review on Automated Billing for Smart Shopping System Using IOT. review of computer engineering studies. <https://doi.org/10.18280/rces.06010>.