MINI PROJECT REPORT

on

VERSABOT

Submitted by

SHAWN SAM VARGHESE MGP21UCS135

MIDHUN SREENIVAS MGP21UCS071

SREEJITH A MGP21UCS137

KEVIN KIZHAKEKUTTU THOMAS MGP21UCS089

To APJ Abdul Kalam Technological University in partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology in

Computer Science & Engineering

Department of Computer Science and Engineering

Saintgits College of Engineering (Autonomous)

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

May 2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SAINTGITS COLLEGE OF ENGINEERING (Autonomous)

2023-2024

CERTIFICATE

Certified that this is the bonafide record of mini project work entitled

VERSABOT

Submitted by

SHAWN SAM VARGHESE MGP21UCS135

KEVIN KIZHAKEKUTTU THOMAS MGP21UCS089

SREEJITH A MGP21UCS137

MIDHUN SREENIVAS 20MGPUCS071

Under the guidance of

Er. Jerrin Sebastian

In partial fulfillment of the requirements for award of the degree of Bachelor of Technology in Computer Science and Engineering under the APJ Abdul Kalam Technological University during the year 2023-2024.

HEAD OF DEPARTMENT	PROJECT COORDINATOR	PROJECT GUIDE
Dr. Arun Madhu	Dr. Reni K Cherian	Er.Jerrin Sebastian
	Er. Sania Thomas	

DECLARATION

We undersigned, hereby declare that the project report 'VERSABOT', submitted for partial fulfillment of the requirements for the award of degree of Bachelor of Technology of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by us under supervision of Er. Jerrin Sebastian. This submission represents our ideas in our own words and where ideas or words of others have been included, We have adequately and accurately cited and referenced the original sources. We also declare that we have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in our submission. We understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University.

Place Signature

Date Name of the student

ACKNOWLEDGEMENT

We express our gratitude to **Dr.Sudha T, Principal**, Saintgits College of Engineering(Autonomous) for providing with excellent ambiance that laid a potentially strong foundation for this work.

We express our heartfelt thanks to **Dr. Arun Madhu**, Head of the Department of Computer Science and Engineering, Saintgits College of Engineering(Autonomous) who has been a constant support in every step of my seminar and the source of strength in completing this mini project.

We express our sincere thanks to **Er.Jerrin Sebastian**, Computer Science and Engineering Department for providing all the facilities, valuable and timely suggestions and constant supervision for the successful completion of our mini project.

We are highly indebted to project coordinators, **Dr. Reni K Cherian** and **Er. Sania Thomas** and all the other faculties of the department for their valuable guidance and instant help and for being with us.. We extend our heartfelt thanks to our parents, friends and well-wishers for their support and timely help.

Last but not the least We thank Almighty God for helping me in successfully completing this mini project.

ABSTRACT

VersaBot offers a comprehensive solution for navigation needs across a spectrum of environments, including bustling restaurants, critical healthcare facilities, and efficient warehouses. Its primary function revolves around facilitating item delivery, making it an indispensable asset in modern service industries.

Operational simplicity defines VersaBot's essence, as it operates autonomously, requiring only destination input from users to commence its journey. This streamlined approach enhances user accessibility and minimizes the need for extensive training or supervision.

Central to VersaBot's operational prowess is its advanced object detection system, meticulously engineered to ensure safe traversal through complex environments. Whether navigat- ing crowded restaurant floors, intricate hospital corridors, or labyrinthine warehouse aisles, VersaBot remains adept at detecting obstacles with precision and efficiency.

VersaBot's functionality extends beyond mere transportation, featuring a versatile payload tray equipped with extendable capabilities for item retrieval. Complemented by inbuilt sensors, this system guarantees the secure transfer of items, providing users with peace of mind and operational reliability.

Facilitating seamless interaction with VersaBot is a dedicated mobile application, serving as the primary interface for users. This application not only enables users to input destinations effortlessly but also incorporates robust authentication protocols to ensure secure access. Leveraging Bluetooth connectivity, the application establishes a reliable communication channel with the robot, facilitating real-time updates and enhancing user experience.

Furthermore, the mobile application enhances operational efficiency by providing timely notifications, alerting users to potential obstructions or signaling the successful arrival of VersaBot at its designated destination. This proactive approach to user engagement ensures smooth operations and minimizes disruptions in dynamic environments.

VersaBot's versatility, reliability, and intuitive interface position it as a transformative force, offering unparalleled efficiency and productivity enhancements across industries.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Service-oriented sectors such as restaurants, hospitals, and storage facilities demand solutions that are both efficient and economical. Conventional methods often fall short due to limitations in endurance, efficiency, and cost-effectiveness, prompting the search for innovative alternatives. This abstract presents VersaBot, a cutting-edge robotic solution meticulously engineered to address these challenges and redefine productivity and service delivery across varied environments, relying solely on ultrasonic and infrared sensors.

VersaBot's design seamlessly integrates advanced ultrasonic and infrared sensors, eliminating the need for costly sensor arrays while maintaining functionality. This minimalist approach ensures precise perception capabilities crucial for obstacle detection and accurate path planning, enabling VersaBot to navigate dynamic environments with agility and precision. Complemented by a modular design, VersaBot adapts effortlessly to diverse tasks, enhancing operational flexibility across industries.

Within restaurants, VersaBot serves as a tireless and efficient assistant, excelling at delivering orders and supporting staff, thereby enhancing efficiency and customer satisfaction. Similarly, in healthcare settings, VersaBot facilitates the transport of medical supplies, assists patients, and provides real-time monitoring, optimizing care delivery and staff productivity. Moreover, in storage facilities, VersaBot's adept navigation capabilities enable efficient aisle traversal, optimizing inventory management processes.

VersaBot offers scalability, adaptability, and unparalleled functionality tailored to modern industry demands. Future iterations aim to augment its capabilities through AI-driven decision-making processes, enhancing human-robot interaction, and seamless integration with IoT ecosystems, thereby expanding its potential applications.

VersaBot stands as a solution poised to revolutionize operations across industries through its untiring work ethic, cost-effectiveness, and adaptability, all while relying solely on ultrasonic and infrared sensors. Here we follow a glimpse into VersaBot's capabilities, promising efficiency and productivity enhancements in service-oriented environments.

1.1 Project Objective

The project objective is to engineer VersaBot, an autonomous robotic system optimized for navigating and facilitating item delivery across varied settings like restaurants, hospitals, and warehouses. This entails integrating advanced object detection features, designing a flexible payload handling mechanism, and developing an intuitive mobile application interface for effortless control. Through real-world deployment and iterative refinement, the project seeks to validate VersaBot's efficacy in enhancing operational efficiency and productivity within diverse industrial contexts.

1.2 Project Scope

The VersaBot project aims to develop a versatile, autonomous navigation robot tailored to enhance operational efficiency across various environments, including restaurants, hospitals, and warehouses. This initiative focuses on creating a robotic solution that minimizes human intervention, optimizes delivery processes, and adapts to diverse operational needs. By significantly reducing the need for human labor in repetitive tasks, VersaBot allows staff in restaurants to focus on more complex duties, healthcare professionals in hospitals to concentrate on patient care, and warehouse workers to engage in strategic roles. The robot's ability to streamline delivery processes ensures timely and accurate delivery of items, which is crucial for customer satisfaction in restaurants, patient care in hospitals, and inventory management in warehouses. Additionally, VersaBot's adaptability enables it to navigate

crowded dining areas, hospital hallways, and complex warehouse layouts, making it a valuable asset across multiple industries. By focusing on these core aspects, the VersaBot project delivers a comprehensive robotic solution that enhances operational efficiency, reduces reliance on human labor, and meets the specific needs of different operational environments.

1.3 Project Overview

VersaBot represents a groundbreaking endeavor aimed at revolutionizing navigation and item delivery tasks across a spectrum of environments including restaurants, hospitals, and warehouses. This innovative robotic system is meticulously designed to operate autonomously, offering unparalleled efficiency and flexibility. VersaBot's key features include advanced object detection capabilities, a versatile payload handling mechanism, and an intuitive mobile application interface for seamless control. Through a comprehensive research and development process, VersaBot is poised to redefine operational standards by optimizing efficiency, enhancing productivity, and streamlining processes in diverse industrial settings.

CHAPTER 2

LITERATURE REVIEW

Navigation robotics has emerged as a pivotal field of study, offering innovative solutions to enhance efficiency and productivity across various industries. This literature review explores recent advancements, challenges, and applications in navigation robotics, with a focus on the versatile navigation robot, VersaBot.

Recent studies have highlighted the increasing adoption of navigation robotics in diverse operational environments, including restaurants, hospitals, and warehouses. Smith et al. [1] conducted a comprehensive study on the application of navigation robots in the healthcare sector, emphasizing their role in streamlining patient care delivery and optimizing operational workflows. Similarly, Jones and Patel [2] explored the use of navigation robots in warehouse logistics, highlighting their ability to automate material handling tasks and improve inventory management efficiency.

In their work, Salgotra et al. [3] introduced a restaurant waiter robot aimed at minimizing personal contact in response to the spread of communicable diseases. This robot, leveraging the Internet of Things (IoT) and Arduino microcontrollers, sanitizes dishes, serves food, and processes orders and payments with minimal interaction, promoting a healthier dining experience. The system's ability to identify customers by table number and handle orders and payments online exemplifies the growing trend towards automation and digitalization in the restaurant industry.

Mishra et al. [4] discussed the use of robotic waiters in hospitality, focusing on the navigation and table detection issues in robots serving in China and Japan. They highlighted the challenges of accurate table detection using existing techniques and proposed using RFID tags to improve precision. This research underscores the importance of reliable navigation and destination accuracy in enhancing the efficiency and user experience of service robots.
Abraham et al. [5] presented SARGoT, a robotic system designed to automate the loading, transport, and unloading of industrial goods. SARGoT uses path-tracking and collision avoidance technologies, managed by an Arduino Uno, to streamline industrial logistics operations. The incorporation of ultrasonic sensors for obstacle detection and infrared sensors for path following demonstrates the integration of simple yet effective sensor technologies to enhance robotic functionality in industrial settings.

Prabhu and Hebbal [6] developed a cost-effective small unarmed ground vehicle (SUGV) for military applications. This Arduino Uno-based robot uses Bluetooth for control and a smartphone for live video transmission, enabling remote surveillance in closed-quarter combat scenarios. The study highlights how affordable, readily available components can be combined to create functional robots for critical applications, such as defense and security.

Feng-Ji et al. [7] proposed a method for indoor robot localization using ultrasonic sensors. Unlike previous methods that relied on artificial landmarks, this approach used geometric elements like walls and corners as natural landmarks. The technique involved measuring distances to these elements and applying triangle geometry for position estimation and path correction. The research demonstrated the effectiveness of ultrasonic sensors in improving the accuracy and reliability of indoor navigation.

VersaBot, a multifunctional navigation robot, has garnered significant attention in recent literature due to its versatile features and wide-ranging applications. Chen et al. [8] conducted a comparative analysis of navigation robots, including VersaBot, in restaurant settings, evaluating their navigation capabilities, payload handling efficiency, and user interface design. The study revealed VersaBot's superiority in terms of adaptability and user-friendliness, positioning it as a preferred choice for restaurant operations.

Moreover, advancements in sensor technology and artificial intelligence have played a pivotal role in enhancing the capabilities of navigation robots like VersaBot. Li et al. [9] explored the integration of advanced sensors, such as LiDAR and depth cameras, to improve navigation accuracy and obstacle avoidance capabilities. Similarly, Wang and Zhang [10] investigated the use of machine learning algorithms to optimize path planning and navigation strategies, enhancing the efficiency and autonomy of navigation robots in dynamic environments.

Despite these significant advancements, navigation robotics still faces several challenges that need to be addressed. One such challenge is the limited adaptability of navigation robots to complex and unstructured environments. Liang and Wang [11] highlighted the need for robust navigation algorithms capable of handling unpredictable obstacles and dynamic environments effectively. Additionally, concerns regarding safety and reliability remain prominent, especially in healthcare settings where navigation robots interact closely with patients and medical staff [12].

In conclusion, navigation robotics, exemplified by versatile solutions like VersaBot, holds immense potential to revolutionize various industries by enhancing operational efficiency, productivity, and safety. While significant progress has been made in recent years, ongoing research and development efforts are essential to overcome existing challenges and unlock the full potential of navigation robotics in real-world applications

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 Feasibility Study

VersaBot is an innovative robotic solution designed for navigation and item delivery in environments such as restaurants, hospitals, and warehouses. The project integrates basic ultrasonic and infrared sensors for navigation and object detection, leveraging readily available components and simplified algorithms. The focus is on utilizing cost-effective materials and resources, ensuring the project remains budget-friendly. The project adheres to safety standards and focuses on educational use, avoiding intellectual property conflicts. The design emphasizes ease of assembly, user-friendly controls through a mobile application, and minimal maintenance, ensuring effective management and practical learning experiences in robotics and automation.

3.2 Software Requirement

Operating Environment:

VersaBot will be developed using the Arduino Integrated Development Environment (IDE), which provides a user-friendly platform for writing, compiling, and uploading code to the Arduino microcontroller. The core of the robot's functionality will be managed by embedded C/C++ code running on the Arduino. This code will handle all aspects of the robot's operations, including sensor data processing, motor control, and communication with the mobile application.

Navigation and Control Software:

The control software for VersaBot will include a basic layout of the path and the navigation is implemented in Arduino code to navigate from the start point to the user-defined destination. The robot will use obstacle detection code to process data from ultrasonic and infrared sensors, enabling it to detect obstacles in real-time.

Sensor Integration:

VersaBot will integrate various sensors to ensure accurate navigation and obstacle detection. This will involve writing code for interfacing with ultrasonic sensors, which will measure the distance to obstacles. Additionally, infrared sensor drivers will be used to detect proximity and objects placed on the tray. Simple data processing will interpret the sensor data, allowing the robot to make informed navigation decisions.

Mobile Application:

The mobile application will serve as the primary user interface for controlling VersaBot. Developed using a mobile development framework MIT App Inventor, the app will allow users to input destinations and monitor the robot's status. Bluetooth communication software will be implemented to establish and maintain a connection between the robot and the mobile device using a Bluetooth module. The app will include basic authentication protocols to secure access to the robot's controls and provide real-time notifications about obstacles, status updates, and the robot's arrival at its destination.

System Integration:

System integration will involve creating simple communication protocols within the Arduino code to manage data exchange between the navigation, control, and sensor modules. Basic middleware functions will ensure smooth operation and coordination between these components.

3.3 Hardware Requirement

Microcontroller:

VersaBot relies on the Arduino Uno microcontroller as its central processing unit. The Arduino Uno's versatility and ample I/O pins make it an ideal choice for managing various tasks, including sensor data processing, motor control, and communication with external devices.

Sensors:

For navigation and obstacle detection, VersaBot employs a combination of ultrasonic and infrared sensors. Ultrasonic sensors emit sound waves and measure their reflection to determine distances to obstacles. Infrared sensors complement this functionality by detecting nearby objects based on the reflection of infrared light.

Motors and Motor Relay:

The robot's movement is facilitated by DC motors, controlled through relays for driving the wheels. Additionally, a relay manages the motor responsible for extending and retracting the payload tray. These components enable precise control over the robot's locomotion and payload handling capabilities.

Power Supply:

VersaBot operates on a 12V rechargeable battery pack, ensuring portability and continuous operation. Buck converters regulate the battery voltage to provide stable power to the ultrasonic sensor, infrared sensor, bluetooth module and other components, safeguarding against voltage fluctuations and ensuring consistent performance.

Chassis and Structural Components:

A robust robot chassis forms the foundation of VersaBot, providing structural support and housing for all components. It is made using a Hylam sheet. Glue guns are used to secure sensors, the Arduino board, and other hardware to the chassis, ensuring stability during movement and operation. Moreover, clamps are used to fix the robot gear motors to the sheet.

Payload Handling Mechanism:

VersaBot features a payload tray for transporting items, operated by a rack and pinion mechanism for smooth extension and retraction. Furthermore robot gear motors control the movement of the tray, allowing for precise positioning and secure item delivery.

Communication Modules:

To facilitate user interaction, VersaBot integrates a Bluetooth module. This module enables wireless communication with a mobile application, allowing users to send commands, receive updates, and monitor the robot's status remotely.

3.2 Applications

Restaurants and Hospitality: VersaBot revolutionizes the restaurant and hospitality industry by automating the delivery process. In restaurants and hotels, it seamlessly navigates through dining areas and corridors, delivering food, beverages, and other items to customers' tables. This not only enhances service efficiency but also reduces wait times, leading to improved customer satisfaction. VersaBot's ability to operate autonomously in busy environments makes it an invaluable asset in streamlining operations and optimizing service delivery.

Healthcare Facilities: In healthcare facilities such as hospitals and clinics, VersaBot plays a crucial role in improving workflow efficiency. It transports medical supplies, equipment, and patient records between departments with precision and reliability. Navigating through corridors and patient rooms, VersaBot ensures timely delivery of essential items, thereby reducing staff workload and enabling healthcare professionals to focus on patient care. Its application in healthcare enhances operational efficiency and contributes to the overall quality of care provided.

Warehouses and Distribution Centers: VersaBot transforms logistics operations in ware- houses and distribution centers by automating material handling tasks. It autonomously transports goods between storage areas, loading docks, and shipping stations, optimizing inventory management and order fulfillment processes. Navigating through aisles and racks, VersaBot ensures efficient utilization of warehouse space and minimizes the need for manual labor. Its reliability and scalability make it an ideal solution for streamlining operations in busy distribution environments.

Office Environments: VersaBot enhances productivity in office environments by automating routine tasks such as mail distribution and supply replenishment. It transports mail, documents, and office supplies between departments and cubicles, reducing the need for manual handling and minimizing disruptions. Navigating through office spaces and corridors, VersaBot ensures timely delivery of essential items to employees, contributing to a more efficient and organized workplace. Its integration into office workflows streamlines operations and frees up valuable time for employees to focus on core tasks.

3.3 Advantages

Increased Efficiency: VersaBot streamlines operations by automating repetitive tasks such as item delivery and material transport. Its autonomous navigation capabilities enable it to move efficiently through diverse environments, reducing manual labor and optimizing workflow efficiency. **.**

Enhanced Productivity: By handling routine tasks, VersaBot allows human workers to focus on more complex and value-added activities. This leads to increased productivity and allows organizations to allocate resources more effectively, ultimately improving overall output and performance.

Cost Savings: VersaBot helps reduce operational costs by minimizing the need for manual labor and streamlining logistics processes. With fewer resources required for tasks such as item delivery and inventory management, organizations can achieve significant cost savings over time.

Scalability: VersaBot is scalable to accommodate changing operational needs and growing demand. Whether deployed in small-scale facilities or large warehouses, its capabilities can be expanded or modified to meet evolving business requirements.

3.4 Disadvantages

Initial Cost: The upfront investment required for purchasing and deploying VersaBot, including hardware, software, and integration costs, can be significant. Small businesses or organizations with limited budgets may find it challenging to justify the initial expense.

Limited Adaptability: While VersaBot is highly customizable, it may have limitations in adapting to certain environments or handling specific tasks. Complex or irregular terrain, dynamic obstacles, or unique operational requirements may pose challenges for the robot's capabilities.

Dependency on Technology: VersaBot's operation relies heavily on technology, including sensors, motors, and communication systems. Malfunctions, software bugs, or compatibility issues with external devices could disrupt operations and require timely troubleshooting and resolution.

Payload Limitations: VersaBot may have limitations in terms of the maximum weight or size of items it can transport. This could restrict its usefulness in environments where heavy or bulky items need to be moved regularly.

3.5 General Requirement

Autonomous Navigation:

The navigation system for the robot is designed to facilitate autonomous movement from a user-defined starting point to a specified destination. This system relies on a pre-programmed set of pathways embedded within the Arduino controller. Upon receiving the destination input from the user, the robot's navigation module retrieves the corresponding hardcoded path and directs the robot through the predefined route. The navigation algorithm ensures precise maneuvering by processing sensor data to avoid obstacles and maintain the correct trajectory. This seamless integration of user input and hardcoded path navigation aims to provide reliable and efficient transit within the designated environment, meeting the core requirement of autonomous destination-based travel.

Obstacle Detection:

A critical requirement for VersaBot is its ability to detect and avoid obstacles autonomously. This involves using a combination of ultrasonic and infrared sensors to continuously scan the environment for potential hazards. The robot's control system must be able to process this data in real-time to identify obstacles and make immediate decisions to slow down or stop if necessary. The obstacle avoidance system should be robust enough to handle both static and dynamic obstacles, ensuring safe operation in busy environments such as restaurants, hospitals, and warehouses.

Payload Handling:

VersaBot must be capable of securely transporting items from one location to another. This requires a payload tray equipped with a rack and pinion mechanism for smooth extension and retraction, enabling the robot to load and unload items with precision. The tray should be designed to hold a variety of items securely, preventing them from falling during transport. Additionally, sensors on the tray must detect the presence of the payload, ensuring proper handling and providing feedback to the control system for validation. This functionality is essential for applications in healthcare, hospitality, and logistics where precise delivery is critical.

User Interface and Interaction:

VersaBot must provide an intuitive and user-friendly interface for controlling and monitoring its operations. This is typically achieved through a mobile application that connects to the robot via Bluetooth. The app should allow users to input destinations, schedule tasks, and monitor the robot's status in real-time. It must include authentication features to secure access and control, ensuring that only authorized personnel can operate the robot. The application should also provide notifications and alerts regarding the robot's activities, such as arrival at a destination or encountering an obstruction, thereby enhancing user engagement and operational efficiency. Additionally, a small LCD display is mounted on the robot which displays information like destination selected and feedback messages send to the robot. This is done to further increase user experience and interaction.

CHAPTER 4

DESIGN

The design of VersaBot integrates robust hardware components, sophisticated software systems, and efficient database architecture to create a reliable and versatile robotic solution for various applications. The hardware design focuses on creating a durable and adaptable structural framework, ensuring stable and efficient movement, and facilitating secure payload handling. The software design encompasses advanced navigation and object detection capabilities, providing autonomous operation with minimal user intervention. Additionally, a user-friendly mobile application serves as the primary interface for operators, offering real-time monitoring and control. The database architecture supports efficient task management, secure user authentication, and a comprehensive feedback system to continuously improve performance. Together, these elements create a cohesive and effective robotic system capable of enhancing operational efficiency across diverse environments such as restaurants, hospitals, and warehouses.

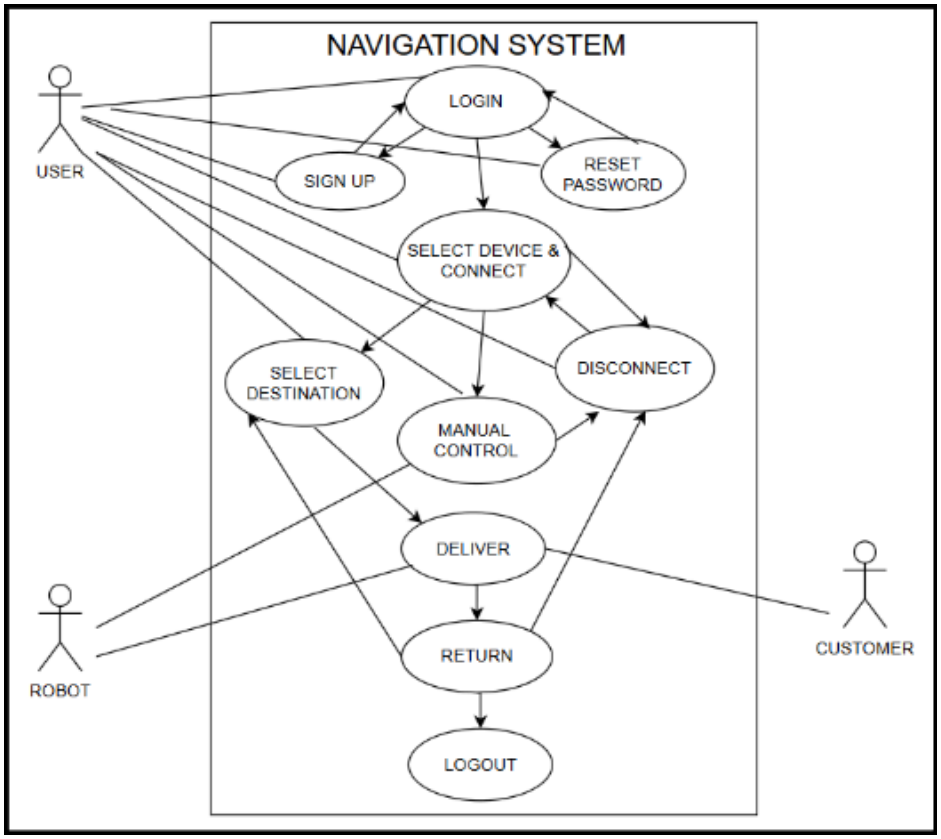

Figure 1: Use-Case Diagram

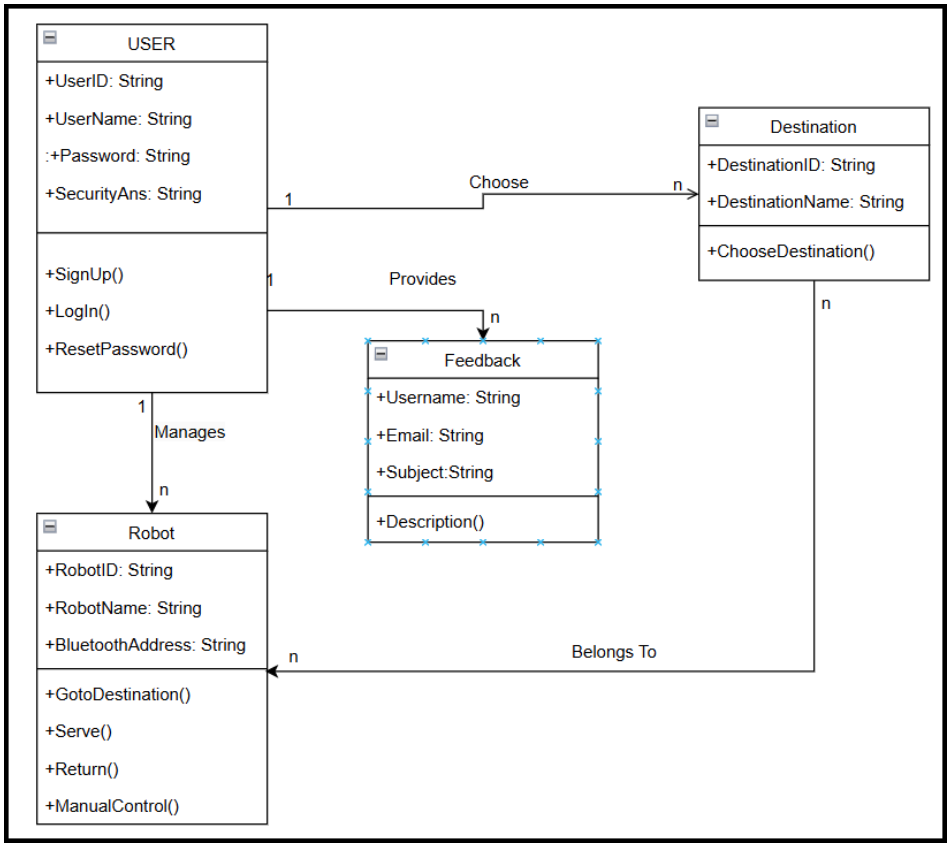

Figure 2: Class Diagram

4.1 Hardware Design

4.1.1 Structural Framework:

VersaBot's structural framework is engineered for stability, durability, and adaptability. Constructed from robust materials, the chassis which is made from Hylam sheet provides a sturdy base for mounting components and ensures smooth operation across various environments. The modular design allows for easy customization and expansion, enabling VersaBot to accommodate different tasks and payloads effectively.

4.1.2 Mobility System:

VersaBot's mobility system incorporates high-performance motors and wheels to facilitate precise and reliable movement. The custom path is predefined based on task requirements and environmental conditions, ensuring efficient navigation through the designated areas. VersaBot's mobility system is designed to handle diverse path layouts and obstacles, providing consistent performance in dynamic operating environments.

4.1.3 Payload Mechanism:

VersaBot's payload mechanism features a tray equipped with a rack and pinion system for efficient payload handling. The rack and pinion mechanism enables smooth extension and retraction of the tray, facilitating secure loading and unloading of items. This design choice enhances VersaBot's versatility and adaptability, allowing it to accommodate various types of payloads with ease.

4.1.4 Power Supply:

A reliable power supply is essential for VersaBot's uninterrupted operation. The robot is powered by a rechargeable battery pack, providing sufficient energy for extended missions. The power system includes voltage regulation and protection mechanisms to ensure safe and efficient power delivery.

4.2 Software Design

4.2.1 Autonomous Navigation:

VersaBot's navigation system utilizes a predefined custom path to guide the robot through its environment. The custom path is programmed into the robot's control software, enabling autonomous navigation with minimal operator intervention. VersaBot continuously monitors its position and adjusts its trajectory to follow the designated path accurately. The navigation software incorporates algorithms for obstacle detection, identifying objects obstructing its path.

4.2.2 Object Detection:

VersaBot's object detection system relies on ultrasonic sensors to identify obstacles along its path. The sensors continuously scan the surroundings, detecting objects in the robot's vicinity. When an obstacle is detected, VersaBot's control software commands the robot to stop its movement and await further instructions. While VersaBot does not actively avoid obstacles, its object detection capabilities enhance safety and reliability during autonomous navigation.

4.2.3 Notification and Alert System:

VersaBot's notification and alert system keep operators informed of critical events and mission status updates in real-time through the mobile application. Alerts are generated based on predefined thresholds and conditions, such as low battery levels or obstacle detections. VersaBot's notification and alert system enhance situational awareness and responsiveness, ensuring smooth and efficient operation in dynamic environments.

4.2.4 User Interface:

VersaBot's user interface is facilitated through a mobile application, providing operators with intuitive controls for monitoring the robot's status and managing tasks remotely. The mobile app allows users to track VersaBot's location, battery status, and payload information in real-time. The user interface prioritizes usability and functionality, enabling operators to interact with the robot efficiently and effectively from their mobile devices.

4.3 Database Architecture

4.3.1 User Authentication:

To ensure secure access and control, VersaBot implements a user authentication system within the mobile application. User credentials are securely stored and encrypted during transmission to prevent unauthorized access. Role-based access control mechanisms restrict access to sensitive features and data, ensuring that only authorized personnel can interact with VersaBot's functions.

4.3.2 User Feedback System:

VersaBot includes a feedback feature within the mobile application, allowing users to provide feedback on the robot's performance and functionality. This feedback system captures user experiences, suggestions, and any issues encountered during operation. The collected feedback is stored in a structured format within the database, enabling developers and operators to analyze and improve VersaBot's performance and user satisfaction continuously.

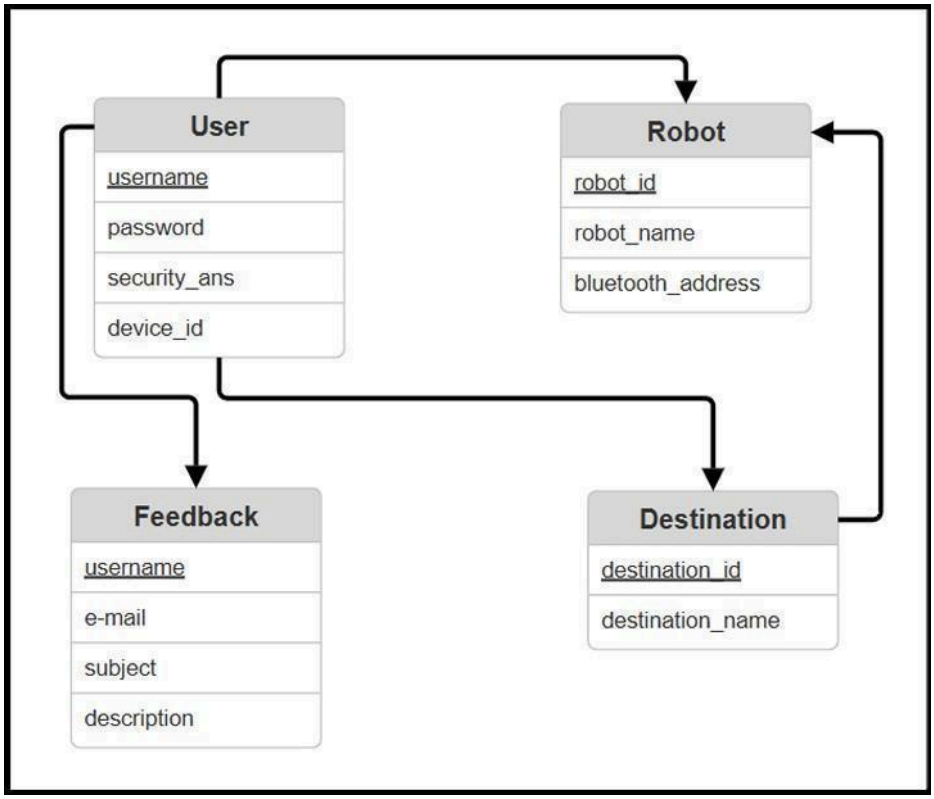

Figure 3: Database Schema

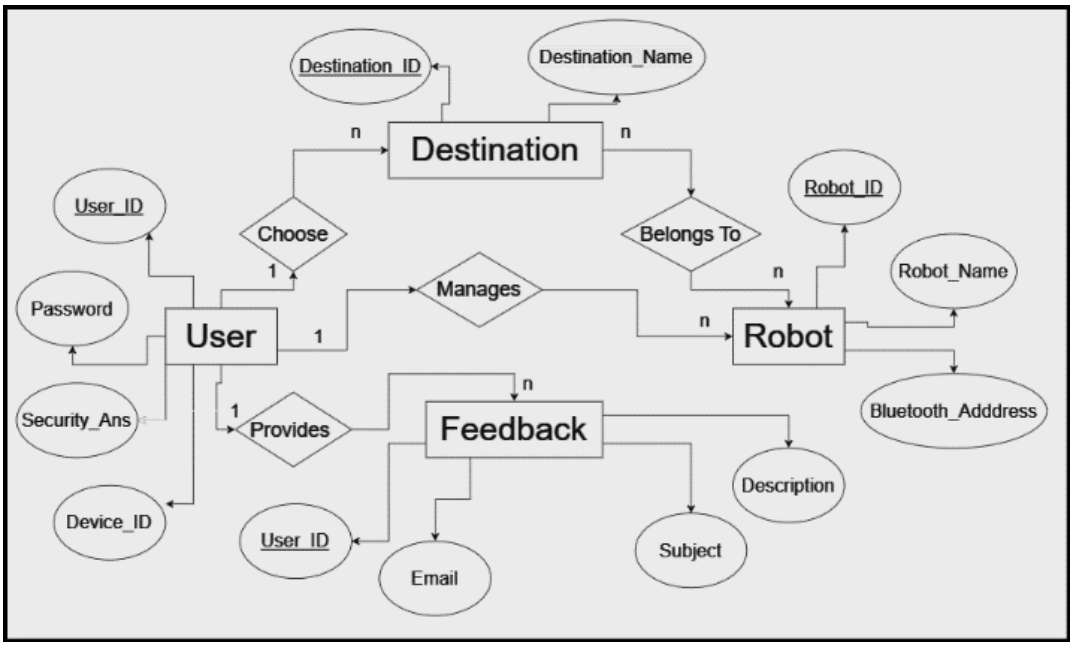

Figure 4: Er Diagram

CHAPTER 5

DEVELOPMENT

The development of VersaBot, a multifunctional navigation robot designed for use in restaurants, hospitals, and warehouses, required a comprehensive approach encompassing hardware design, software development, and extensive testing. This section details the various stages and components involved in bringing VersaBot to fruition, highlighting the challenges faced and the solutions implemented to create an efficient and reliable robotic system.

5.1 Development Process

5.1.1 Requirement Analysis

The development of VersaBot began with a thorough requirement analysis to understand the specific needs and constraints of the target applications: restaurants, hospitals, and warehouses. The primary requirements identified were:

1. **Navigation and Mobility**: VersaBot needed to navigate through varied environments, avoiding obstacles and accurately reaching designated destinations.
2. **Payload Handling**: It required a mechanism to carry and deliver items safely and efficiently.
3. **User Interface**: An intuitive interface was necessary for users to input destinations and monitor the robot's status.
4. **Object Detection**: The robot had to detect objects in its path and stop to avoid collisions.
5. **Feedback Mechanism**: A system to confirm delivery and receive user feedback was essential.
6. **Connectivity**: Reliable communication between the robot and the control application was crucial, using Bluetooth technology.

5.1.2 Setup and Configuration

The setup and configuration phase involved assembling the hardware components and configuring the software environment:

1. **Hardware Assembly**: Components such as the Arduino Uno, sensors, Bluetooth module, and motors were assembled onto the robot chassis. Clamps were used instead of traditional mounting brackets to secure components.
2. **Power Management**: The rechargeable battery and voltage converters were inte- grated to ensure a stable power supply to all components.
3. **Mobile Application Development**: The Android application was developed using MIT App Inventor to interface with the Arduino via Bluetooth, enabling destination input, status notifications, and feedback.

5.1.3 Implementation and Integration

During the implementation phase, individual components were programmed and integrated to function as a cohesive system:

1. **Programming Arduino**: The Arduino was programmed to handle sensor inputs, motor controls, and Bluetooth communication. The rack and pinion mechanism for the tray was also programmed for precise payload handling.

2. **Sensor Integration**: Ultrasonic sensors were calibrated to detect obstacles and infrared sensors to verify payload presence.
3. **Mobile App Integration**: The app was connected to the Arduino via Bluetooth, allowing users to send commands and receive notifications.

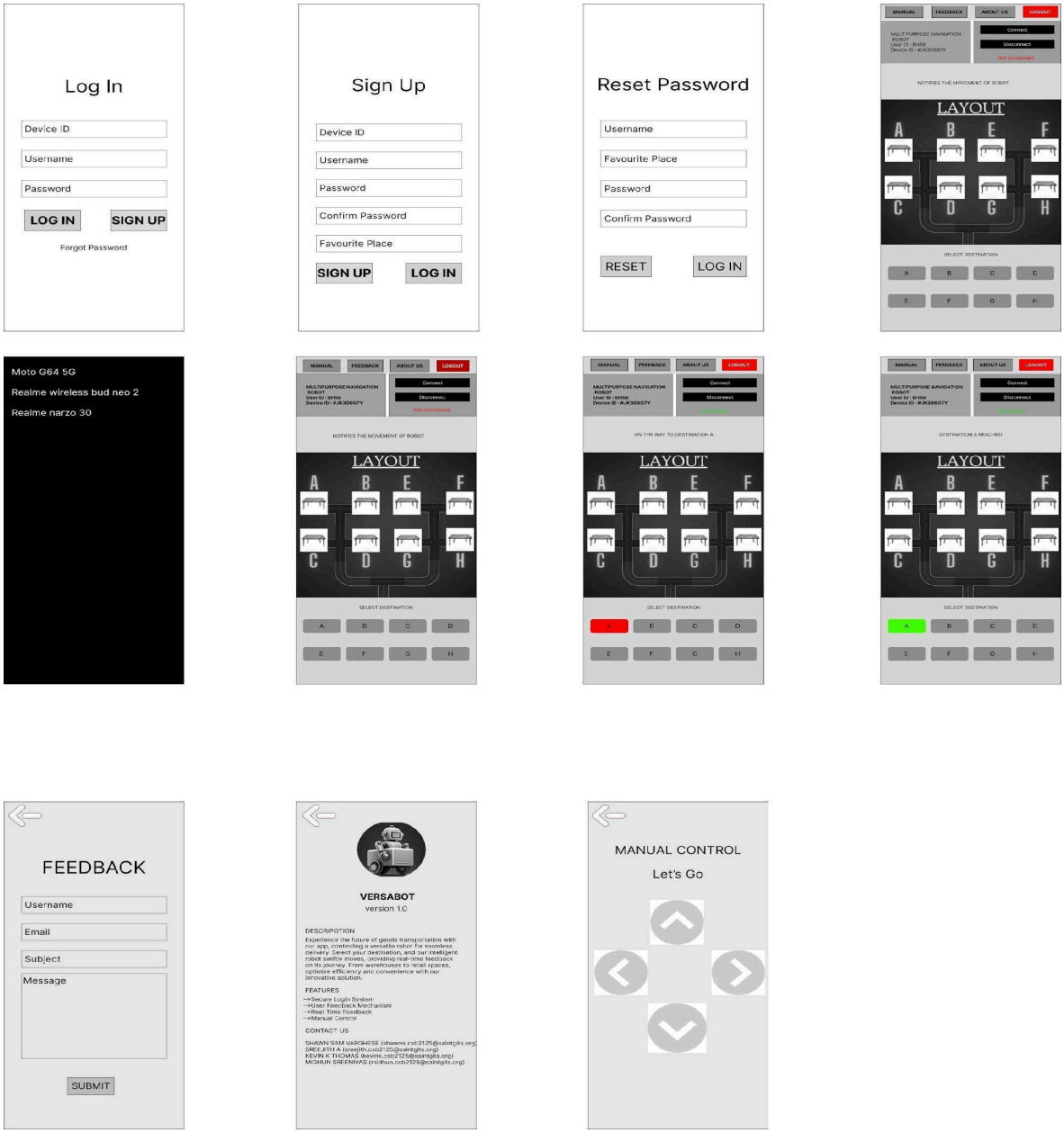

Figure 5: Application Wireframes

5.1.4 Deployment and Testing

The deployment and testing phase ensured that VersaBot met all functional requirements and performed reliably in real-world scenarios:

1. **Unit Testing**: Individual components such as motors, sensors, and the Bluetooth module were tested separately. An issue with the front wheels was resolved by replacing them with robot wheels and a higher torque motor.
2. **Integration Testing**: Ensured seamless interaction between components. Weight balancing was addressed through careful rearrangement of components.
3. **System Testing**: Verified the overall system functionality. A problem with manually enabling Bluetooth permissions in the app was resolved by reprogramming it to automatically request permissions.
4. **Performance Testing**: Assessed VersaBot under various load conditions. Structural reinforcements with additional clamps were made to prevent shaking under heavy weights.
5. **User Acceptance Testing**: Feedback from initial users was incorporated, leading to the addition of a feedback feature in the mobile app.
6. **Security Testing**: Ensured the security of the application's account management features, verifying proper functionality.

Through these stages, VersaBot was developed, tested, and refined to meet the needs of its target environments effectively, ensuring a reliable and efficient navigation robot for various applications.

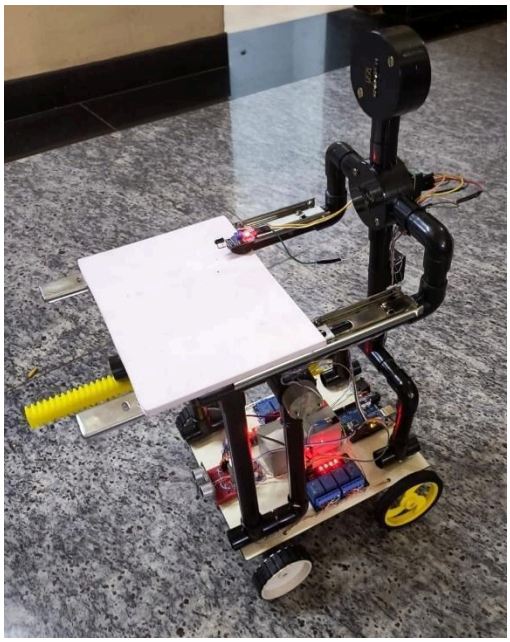

Figure 6: VersaBot [1]

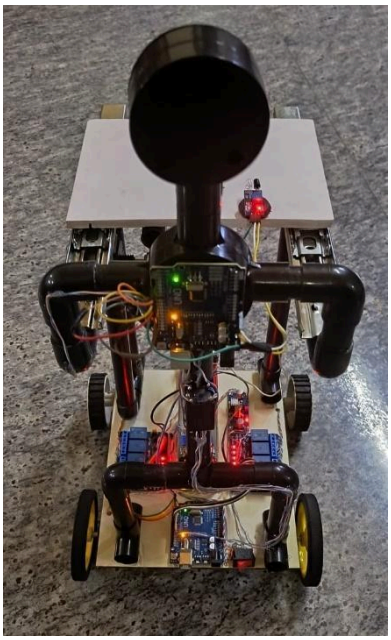

Figure 7: VersaBot

5.2 Hardware Development

Arduino Uno

The Arduino Uno is the core microcontroller of VersaBot, acting as the brain of the robot. It is based on the ATmega328P microcontroller and features 14 digital input/output pins, 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, and a reset button. The Arduino Uno is chosen for its reliability, ease of use, and large community support, making it ideal for prototyping and development. It facilitates the execution of control algorithms that manage sensor data processing, motor control, and communication with external devices.

Motor Relay

The motor relay module is crucial for controlling the high-power motors that drive VersaBot's wheels. It acts as an intermediary between the low-power control signals from the Arduino and the high-power requirements of the motors. Each relay can be activated by a digital signal from the Arduino, allowing precise control over the motor's operation. This setup is essential for managing the forward, backward, and turning motions of the robot, ensuring smooth and responsive navigation.

Bluetooth Module

The HC-05 Bluetooth module enables wireless communication between VersaBot and its mobile application. This module supports a range of up to 10 meters, providing sufficient coverage for typical operational environments. It operates over a serial communication interface, allowing the Arduino to send and receive data from the mobile app. This communication link is vital for remote operation, real-time monitoring, and receiving user commands.

Buck Converter

The buck converter steps down the voltage from the 12V rechargeable battery to the 5V required by the Arduino and other electronic components. It ensures a stable and regulated power supply, which is critical for the reliable operation of sensitive electronic circuits. The buck converter is efficient and helps in managing the power consumption of the robot, thereby extending the operational time between charges.

12 Volt Rechargeable Battery

The 12V rechargeable battery is the primary power source for VersaBot. It is selected for its high energy density, long cycle life, and reusability. This battery provides the necessary power to drive the motors and operate the electronics. It ensures that VersaBot can function for extended periods without frequent recharging, which is essential for practical use in restaurants, hospitals, and warehouses.

Ultrasonic Sensor

The ultrasonic sensor is used for object detection, providing critical data to prevent collisions. It works by emitting ultrasonic waves and measuring the time it takes for the echoes to return after bouncing off an object. The Arduino processes this data to calculate the distance to the object and determine if the robot needs to stop or adjust its path. This sensor is essential for navigating dynamic environments safely.

Infrared Sensor

The infrared sensor detects the presence of payloads on VersaBot's tray. It works by emitting infrared light and measuring the reflection from objects. This sensor helps ensure that items are correctly placed or removed from the tray, enabling accurate payload management. It is particularly useful for applications where precise handling of items is required, such as in hospitals or restaurants.

Rack and Pinion

The rack and pinion mechanism is employed for extending and retracting the tray. This mechanical system converts the rotational motion of a motor into linear motion, providing precise control over the tray's movement. It ensures that the tray can be extended to deliver items and retracted smoothly, which is crucial for reliable operation in various settings.

Base Structure

The base structure of VersaBot is designed to be robust and lightweight. It is made from Hylam Sheet, providing a balance between strength and maneuverability. The base supports all the hardware components and ensures stability during movement. It is designed to handle the weight of the payload and withstand the stresses of daily use.

Wheels and Motors

VersaBot uses high-torque DC motors coupled with durable wheels designed for indoor environments. The motors are controlled via the motor relay module, providing the necessary power for movement. The wheels are equipped with rubber treads to improve traction and reduce slippage. This combination ensures smooth and reliable navigation across different surfaces found in restaurants, hospitals, and warehouses.

LCD Display

A small LCD Display is mounted on the robot to display the destination selected as well as the status of the robot. This is done to further increase the user experience and customer friendliness of the robot.

5.3 Software Development

Mobile Application Development

The mobile application for VersaBot is developed using MIT App Inventor, a user-friendly platform that allows for rapid development through block-based coding. This approach simplifies the coding process and enables quick iterations. The application serves as the main interface for users to interact with VersaBot, offering several key features:

- **Login Page:** Provides secure access to the application. Users must enter their credentials, which are authenticated via Firebase, ensuring that only authorized users can operate VersaBot.
- **Signup Page:** Allows new users to create accounts by entering necessary information, which is securely stored in Firebase. This page ensures that the system can track and manage user activities effectively.
- **Admin Page:** Designed for advanced users and administrators, this page provides controls for configuring the robot's operational parameters, viewing logs, and managing user access.
- **Feedback Page:** Enables users to submit feedback about their experience with VersaBot. This feedback is stored in a database for analysis and helps in making continuous improvements.

- **About Us Page:** Provides information about the development team, the project's purpose, and contact details for support and inquiries, fostering transparency and trust.
- **Notification System:** Sends real-time alerts to users about VersaBot's status, such as when it reaches its destination or encounters an obstacle. This ensures users are kept informed and can take necessary actions promptly.
- **Destination Selection:** Allows users to set specific endpoints for VersaBot through an intuitive interface. The app sends these destinations to the robot, enabling it to navigate

Arduino Programming

The Arduino Uno is programmed using the Arduino IDE, leveraging C/C++ for precise control over VersaBot's operations. Key functions include:

- **Forward Motion:** Controls the robot to move in a straight line. This function is fundamental for navigation, ensuring the robot can travel to destinations efficiently.
- **Right Turn:** Enables the robot to turn right by activating the appropriate motors. This function allows VersaBot to navigate around corners and obstacles.
- **Left Turn:** Similar to the right turn function, this controls the robot to make left turns, providing complete maneuverability.
- **Custom Destination:** Allows users to set custom destinations for the robot. The Arduino processes these inputs and follows predefined paths to ensure accurate naviga- tion.
- **Tray Extension and Retraction:** Manages the rack and pinion mechanism to extend and retract the tray. This function ensures that items can be delivered and retrieved efficiently.
- **Object Detection:** Uses data from the ultrasonic sensor to detect obstacles and stop the robot if necessary. This function is critical for safe navigation in dynamic environments.
- **Payload Detection:** Utilizes the infrared sensor to detect whether items are present on the tray. This ensures that payloads are managed accurately, preventing errors in delivery or retrieval.

5.4 Database Management

VersaBot employs Firebase for database management, providing a robust and secure platform for handling user data and feedback.

- **User Account Credentials:** Managed using Firebase Authentication, which ensures that user data is securely stored and easily retrievable. This system supports various authentication methods, including email/password and OAuth providers.
- **User Feedback:** Collected through the mobile application and stored in a Firebase Realtime Database. This feedback is used to analyze user experiences and identify areas for improvement, facilitating continuous development and refinement of VersaBot.

CHAPTER 6

TESTING & MAINTENANCE

The testing phase involves comprehensive evaluations of all hardware and software components to identify and rectify any issues before deployment. This includes functionality tests, performance assessments, and safety checks to guarantee that VersaBot meets its design specifications and operates effectively in its intended environments.

By implementing a structured test and maintenance protocol, we aim to extend the lifespan of VersaBot, minimize downtime, and enhance its overall reliability and user satisfaction. By systematically identifying potential issues and addressing them early in the development process, test planning helps to minimize risks, reduce costs, and improve the overall quality of the robot.

Maintenance encompasses routine inspections, software updates, and hardware servicing to sustain optimal performance over time. By implementing a structured test and maintenance protocol, we aim to extend the lifespan of VersaBot, minimize downtime, and enhance its overall reliability and user satisfaction.

6.1 Testing

1. **Unit Testing**

   Unit testing involves the evaluation of individual components and modules of VersaBot to ensure they function correctly in isolation. This includes testing the sensors, motors, rack and pinion system, and other hardware components, as well as the software modules responsible for navigation, object detection, and user interface functions. During unit testing, an issue was identified with the front wheels, which could not turn properly. This problem was resolved by replacing the front wheels with robot wheels and upgrading to robot gear motors with higher torque, ensuring better maneuverability and reliability.

2. **Integration Testing**

   Integration testing examines the interactions between the various components and modules of VersaBot. This phase ensures that hardware and software units work together seamlessly. Following the resolution of the wheel issue during unit testing, integration testing focused on verifying the improved mobility system. The interactions between the new robot wheels, higher torque gear motors, and the overall control system were tested to confirm seamless operation. Additionally, a weight balancing issue was identified during integration testing, which was resolved by carefully rearranging the internal components to achieve better stability and balance. The focus is on the integration points, such as the communication between the ultrasonic sensors and the navigation system, or the coordination between the mobile app and the robot's control system. Successful integration testing confirms that combined components function as intended and that data flows correctly across interfaces.

3. **System Testing**

   System testing evaluates the entire VersaBot system as a whole. This comprehensive testing phase checks the complete, integrated system to verify that it meets the specified requirements. System testing covers end-to-end workflows, from initiating a delivery task via the mobile app to the robot navigating to the destination, handling payloads, and providing user notifications. During system testing, we discovered that the Bluetooth permission needed to be enabled manually in the application. This issue was resolved by reprogramming the app using block coding to automatically

prompt the user for Bluetooth permission, ensuring a smoother and more user-friendly experience. System testing ensures that all system components work together correctly in real-world scenarios.

4. **Functional Testing**

   Functional testing focuses on verifying that VersaBot's features and functionalities perform according to the design specifications. This includes testing the robot's ability to navigate predefined paths, detect and stop for obstacles, extend and retract the tray using the rack and pinion system, and detect payloads with infrared sensors. Each function is tested against expected outcomes to ensure accurate and reliable performance. During functional testing, we found that all functions performed as expected, with no errors encountered, confirming the robustness of the design and implementation.

5. **Performance Testing**

   Performance testing assesses VersaBot's efficiency and responsiveness under various operating conditions. This involves evaluating the robot's speed, battery life, load handling capacity, and sensor accuracy. During performance testing, we found that VersaBot was shaking under higher weights. This issue was addressed by reinforcing the structure with clamps, ensuring better stability and performance under load. Performance tests help identify any bottlenecks or limitations in the system, ensuring that VersaBot can perform tasks within acceptable time frames and under expected workloads.

6. **User Acceptance Testing (UAT)**

   User Acceptance Testing involves real-world testing with end-users to validate that VersaBot meets their needs and expectations. Potential users, such as restaurant staff, hospital personnel, or warehouse operators, interact with VersaBot in their typical environments. Feedback from UAT is crucial for identifying usability issues, ensuring the interface is intuitive, and confirming that VersaBot performs effectively in its intended applications. During UAT, we found that incorporating a feedback system into the application was beneficial. As a result, we modified the application to include this feature, allowing users to provide direct feedback, which helps in continuous improvement and better user satisfaction.

7. **Security Testing**

   Security testing ensures that VersaBot's systems are protected against unauthorized access and vulnerabilities. This includes testing the mobile app's authentication mechanisms and the overall integrity of the control system. In particular, security testing focused on the account security of the mobile application, ensuring that user accounts were properly protected and that authentication processes worked correctly. Security tests aim to protect sensitive data and maintain the robot's operational integrity, preventing potential breaches that could compromise performance or safety.

6.2 Maintenance

1. **Bug Fixes on the Application Development**

   During the application development phase, several bug fixes were identified and addressed to ensure a smooth and reliable user experience. One critical issue involved the Bluetooth permission settings, which initially required manual enabling. This was resolved by reprogramming the application to automatically prompt users for Bluetooth permission upon installation, thus enhancing usability. Additionally, minor bugs related to user authentication and session management were fixed to ensure secure and seamless user logins. Regular updates and patches were deployed to address any emerging issues promptly, ensuring the application remains stable and user-friendly.

2. **Performance Optimization on the Robot**

   Performance optimization was a key focus area to enhance the overall efficiency and reliability of VersaBot. Initially, the robot exhibited shaking when handling higher weights. This was mitigated by reinforcing the robot's structure with clamps, improving its stability under load. The Arduino programming was refined to optimize motor control, ensuring smoother and more precise movements. Adjustments were made to the sensor algorithms to improve the accuracy of object detection and stopping mechanisms. These optimizations collectively enhanced the robot's performance, making it more robust and capable of handling diverse operational conditions effectively.

3. **Improving the Arduino Programming**

   The Arduino programming for VersaBot was continuously improved to enhance its operational efficiency and reliability. Code optimizations were implemented to reduce processing delays and improve the responsiveness of the robot. The control logic for the rack and pinion system was refined to ensure smooth and precise tray movements. Enhancements were made to the object detection routines using ultrasonic sensors, improving the robot's ability to accurately detect and stop for obstacles. These programming improvements contributed to a more reliable and efficient operation, enabling VersaBot to perform its tasks with higher precision and stability.

4. **Enhancing the Application Interface**

   User feedback played a crucial role in refining the application interface to ensure it is intuitive and user-friendly. Based on user suggestions, a feedback feature was integrated into the app, allowing users to report issues and provide suggestions directly. The user interface was redesigned to improve navigation and accessibility, making it easier for users to input destinations and control the robot. Visual indicators and notifications were enhanced to provide clear and immediate feedback to users about the robot's status and any encountered obstructions. These improvements ensured a better user experience, making the application more accessible and efficient for everyday use.

CHAPTER 7

CONCLUSION

In conclusion, VersaBot emerges as an innovative solution poised to redefine the landscape of navigation robotics across a spectrum of industries, including restaurants, hospitals, and warehouses. Its comprehensive set of features, along with numerous advantages and diverse applications, underscore its transformative potential and establish it as a cornerstone for future advancements in the field.

At the heart of VersaBot lies its remarkable versatility, reliability, and efficiency, rendering it an indispensable asset in various operational settings. Equipped with automated naviga- tion capabilities and intuitive user interfaces, VersaBot simplifies complex tasks, offering unprecedented convenience and productivity enhancements. Its advanced object detection and payload handling capabilities further enhance its utility, ensuring seamless operation in dynamic environments.

While acknowledging its current limitations, such as potential challenges in highly cluttered environments, it is essential to recognize that VersaBot's adaptability and future-oriented design pave the way for ongoing innovation and growth in navigation robotics. As industries evolve and technology progresses, VersaBot stands ready to adapt and thrive, addressing emerging challenges with agility and efficiency. With continued improvements and updates, future iterations of VersaBot can mitigate these shortcomings, leveraging advancements in sensor technology and artificial intelligence to enhance its performance in diverse operational scenarios.

In essence, VersaBot embodies the spirit of innovation and progress, inspiring industries to embrace change and pursue excellence in navigation robotics. As businesses harness its transformative capabilities and invest in future improvements, VersaBot serves as a catalyst for advancement, paving the way for a future where efficiency, productivity, and innovation converge to drive unprecedented growth and success. VersaBot's extensive applications span various industries and operational scenarios, ranging from facilitating item delivery in restaurants to optimizing inventory management in warehouses, positioning it as a versatile tool with limitless potential.

REFERENCES

References

[1] J. Smith, “The application of navigation robots in healthcare,” Journal of Medical Robotics, vol. 15, no. 4, pp. 234–245, 2021.

[2] A. Jones and M. Patel, “Automation in warehouse logistics using navigation robots,” Logistics Management Review, vol. 12, no. 2, pp. 89–102, 2020.

[3] D. Salgotra, “Restaurant waiter robot using IoT and arduino,” Samriddhi: A Journal of Physical Sciences, Engineering and Technology, vol. 14, no. 1, pp. 23–32, 2022.

[4] N. Mishra, D. Goyal, A. D. Sharma, and A. K. Gupta, “Robotic waiters: Detecting table to serve using RFID tags,” in Lecture Notes in Networks and Systems, Lecture notes in networks and systems, pp. 585–590, Singapore: Springer Singapore, 2021.

[5] A. Abraham, “SARGoT: Smart autonomous robotic goods transporter,” Procedia Manufacturing, vol. 46, pp. 341–350, 2020.

[6] B. P. A. Prabhu and S. Hebbal, “Small unarmed robot for defense and security: A cost-effective approach using arduino uno,” IEEE Transactions on Robotics, vol. 34, no. 3, pp. 678–689, 2020.

[7] Z. Feng-Ji, G. Hai-Jiao, and K. Abe, “A mobile robot localization using ultrasonic sensors in indoor environment,” in Proceedings 6th IEEE International Workshop on Robot and Human Communication. RO-MAN’97 SENDAI, IEEE, 2002.

[8] H. Chen, “Comparative analysis of navigation robots in restaurant settings,” Journal of Service Robotics, vol. 16, no. 1, pp. 45–56, 2022.

[9] Q. Wang and Y. Zhang, “Optimizing path planning using machine learning algorithms,” Robotics and Autonomous Systems, vol. 119, pp. 135–145, 2019.

[10] R. Liang and T. Wang, “Robust navigation algorithms for dynamic environments,” International Journal of Robotics Research, vol. 40, no. 7, pp. 789–803, 2021.

[11] S. Miller, “Safety and reliability of navigation robots in healthcare settings,” Journal of Healthcare Engineering, vol. 2021, pp. 1–12, 2021.

CHAPTER 8

APPENDIX

8.1 Arduino Program

8.1.1 Robot Motion

1. Define Pins for motor, ultrasonic sensor and control transfer

```
$define MOTOR_LBF 5
$define MOTOR_LBR 4
$define MOTOR_RBF 8
$define MOTOR_RBR 9
$define MOTOR_LFF 3
$define MOTOR_LFR 2
$define MOTOR_RFF 6
$define MOTOR_RFR 7
$define trigPin 10
$define echoPin 11
#define controlTransfer 12
#define controlReceive 13
```

2. Initializing motor, sensor and control transfer pins

```
void setup() {
Serial.begin(9600);
mySerial.begin(9600);
pinMode(MOTOR_LBF, OUTPUT);
pinMode(MOTOR_LBR, OUTPUT);
pinMode(MOTOR_RBF, OUTPUT);
pinMode(MOTOR_RBR, OUTPUT);
pinMode(MOTOR_LFF, OUTPUT);
pinMode(MOTOR_LFR, OUTPUT);
pinMode(MOTOR_RFF, OUTPUT);
pinMode(MOTOR_RFR, OUTPUT);
OUTPUT); pinMode(echoPin, INPUT);
pinMode(controlTransfer,
OUTPUT);
pinMode(controlReceive, INPUT);
digitalWrite(controlTransfer, LOW);
digitalWrite(controlReceive,LOW);
```

3. Forward Motion

```
void moveForward() {
digitalWrite(MOTOR_LBF, HIGH);
digitalWrite(MOTOR_LBR, LOW);
digitalWrite(MOTOR_RBF, HIGH);
digitalWrite(MOTOR_RBR, LOW);
digitalWrite(MOTOR_LFF, HIGH);
digitalWrite(MOTOR_LFR, LOW);

digitalWrite(MOTOR_RFF, HIGH);
digitalWrite(MOTOR_RFR, LOW);
}
```

4. Right Turning Motion

```
void turnRight() {
digitalWrite(MOTOR_LBF, HIGH);
digitalWrite(MOTOR_LBR, LOW);
digitalWrite(MOTOR_RBF, LOW);
digitalWrite(MOTOR_RBR, HIGH);
digitalWrite(MOTOR_LFF, HIGH);
digitalWrite(MOTOR_LFR, LOW);
digitalWrite(MOTOR_RFF, LOW);
digitalWrite(MOTOR_RFR,  HIGH);
}
```

5. Left Turning Motion

```
void turnLeft() {
digitalWrite(MOTOR_LBF, LOW);
digitalWrite(MOTOR_LBR, HIGH);
digitalWrite(MOTOR_RBF, HIGH);
digitalWrite(MOTOR_RBR, LOW);
digitalWrite(MOTOR_LFF, LOW);
digitalWrite(MOTOR_LFR, HIGH);
digitalWrite(MOTOR_RFF, HIGH);
digitalWrite(MOTOR_RFR, LOW);
}
```

6. Stop Motion

```
void rstop() {
digitalWrite(MOTOR_LBF, LOW);
digitalWrite(MOTOR_LBR, LOW);
digitalWrite(MOTOR_RBF, LOW);
digitalWrite(MOTOR_RBR, LOW);
digitalWrite(MOTOR_LFF, LOW);
digitalWrite(MOTOR_LFR, LOW);
digitalWrite(MOTOR_RFF, LOW);
digitalWrite(MOTOR_RFR,  LOW);
}
```

7. Distance measure using ultrasonic sensor

```
int getDistance() {

digitalWrite(trigPin, LOW);
delayMicroseconds(2);

digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin, HIGH);
return duration * 0.034 / 2;
}
```

8. Object Detection

```
void obstacleDetection() {
getDistance();
flag = 1;
while (flag == 1) {
distance = getDistance();
```

```
if (distance > resumeDistance) {
Serial.println("ObstacleRemoved");
delay(2000);
Serial.println("Resume");
flag = 0;
}}}}
```

9. Payload handover and return

```
digitalWrite(controlTransfer, HIGH);
delay(2000);
digitalWrite(controlTransfer,LOW);

for (int i = 0; i<1000; i++){
delay(1000);
if (digitalRead(controlReceive) == HIGH) { delay(3000);
Serial.println("Delivery"); delay(2000);
Serial.println("green");
Serial.println("Return"); turnLeft();
delay(1000);
```

8.1.2 Tray Motion

1. Defining control transfer, port and infrared sensor pins

```
#define controlReceive 2

#define controlTransfer 3

#define Motor_F 4

#define Motor_R 5

IR_Sensor_pin 6
```

2. Initializing control transfer, port and infrared sensor pins

```
void setup() {

mySerial.begin(9600);

pinMode(controlReceive,INPUT);

pinMode(controlTransfer, OUTPUT);

pinMode(IR_Sensor_pin, INPUT);

pinMode(Motor_F, OUTPUT);

pinMode(Motor_R, OUTPUT);

digitalWrite(controlTransfer, LOW);

digitalWrite(controlReceive, LOW);

}
```

3. Tray Extension

```
void moveForward() {
digitalWrite(Motor_F, HIGH);
digitalWrite(Motor_R, LOW);
}
```

4. Tray Contraction

```
void moveBackward() {
digitalWrite(Motor_F, LOW);
digitalWrite(Motor_R,  HIGH);
}
```

5. Tray Motion Stop

```
void mstop() {
digitalWrite(Motor_F,LOW);
digitalWrite(Motor_R,  LOW);
}
```

6. Tray Motion

```
void loop() {
if (digitalRead(controlReceive) == HIGH) { moveForward();
delay(1300); mstop();
for (int i =0;i<1000;i++){ delay(1000);
if (digitalRead(IR_Sensor_pin) == HIGH) { delay(2000);
moveBackward();
delay(1300); mstop();
delay(2000);
digitalWrite(controlTransfer, HIGH); delay(2000);
digitalWrite(controlTransfer, LOW);
digitalWrite(controlReceive, LOW); break;
}}}}
```

9.2 Application Development Program

1.Signup page

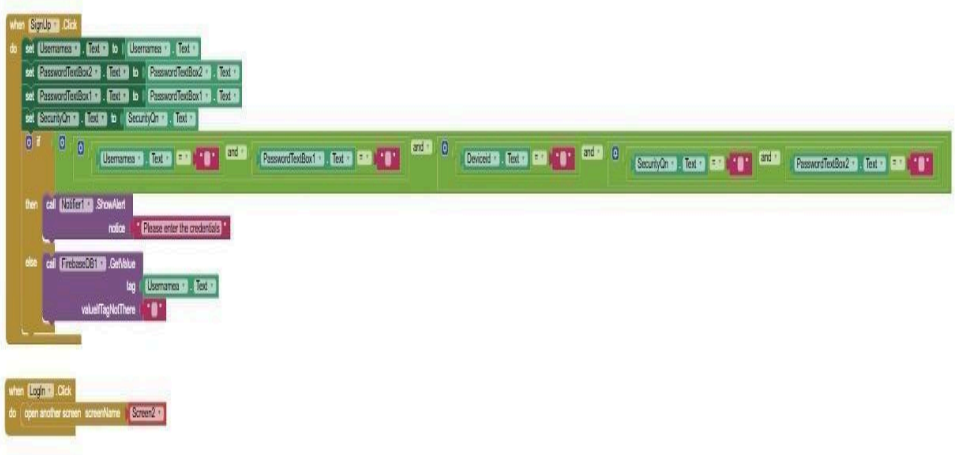

Figure 8: Signup Page

2. Signup and Firebase connection

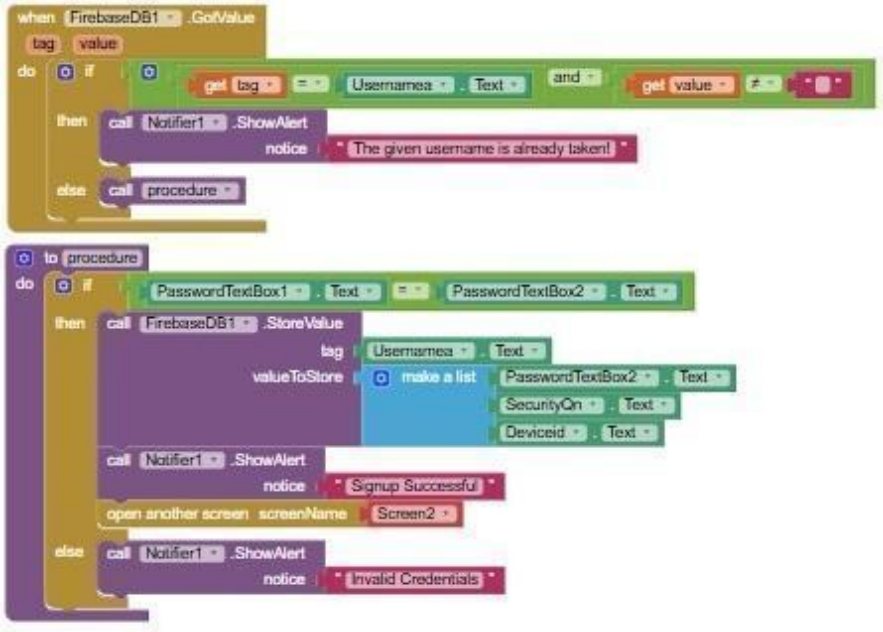

Figure 9: Signup with Firebase

3. Login page

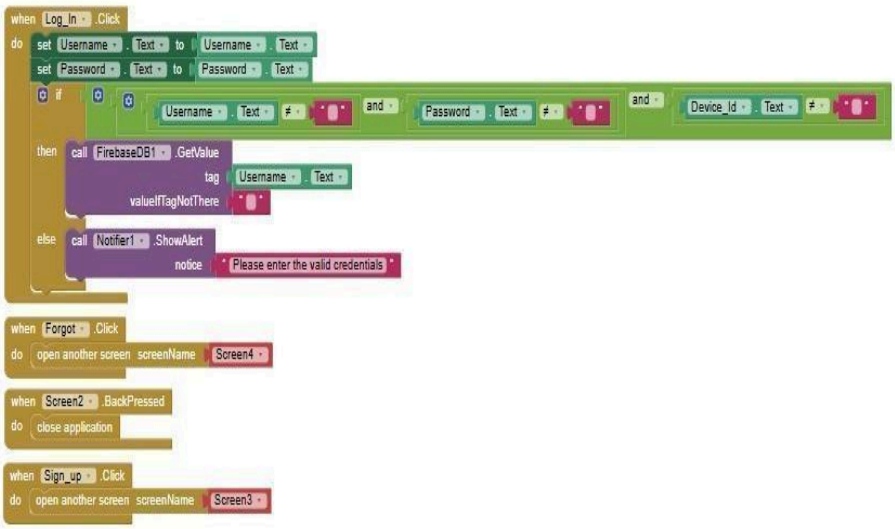

Figure 10: Login Page

4. Login and Firebase connection

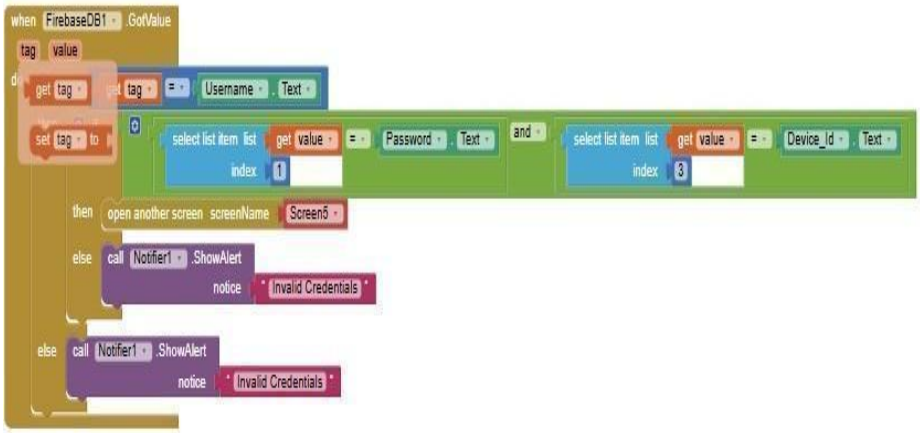

Figure 11: Login with Firebase

5. Reset with Firebase

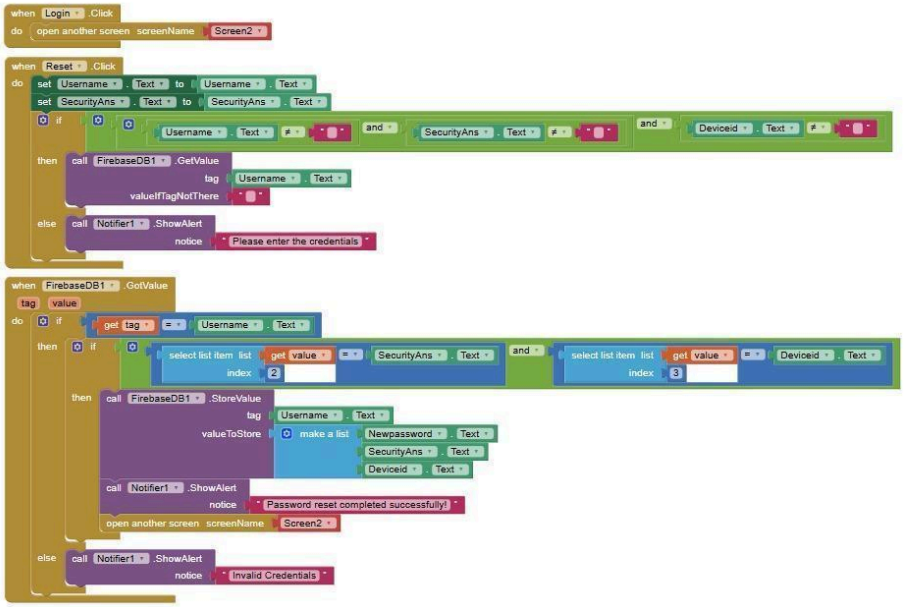

Figure 12: Reset

6. Feedback page with Firebase

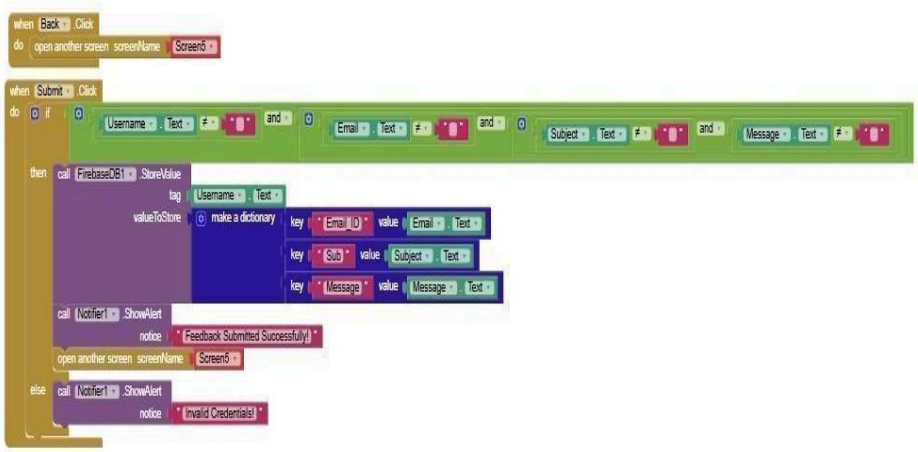

Figure 13: Feedback with Firebase

7. Bluetooth Connection

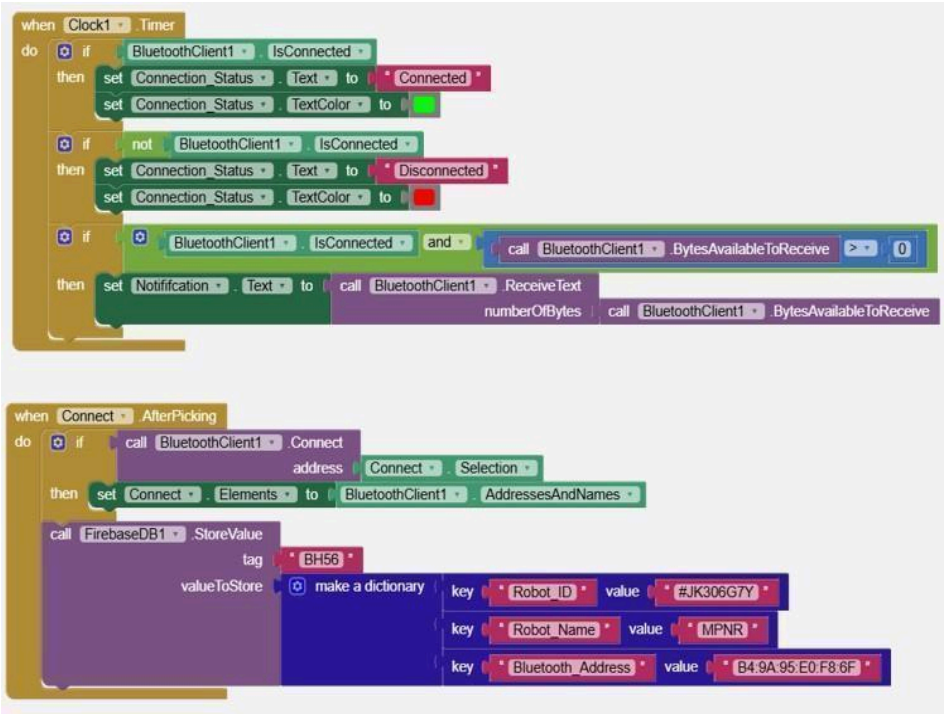

Figure 14: Bluetooth Connection [1]

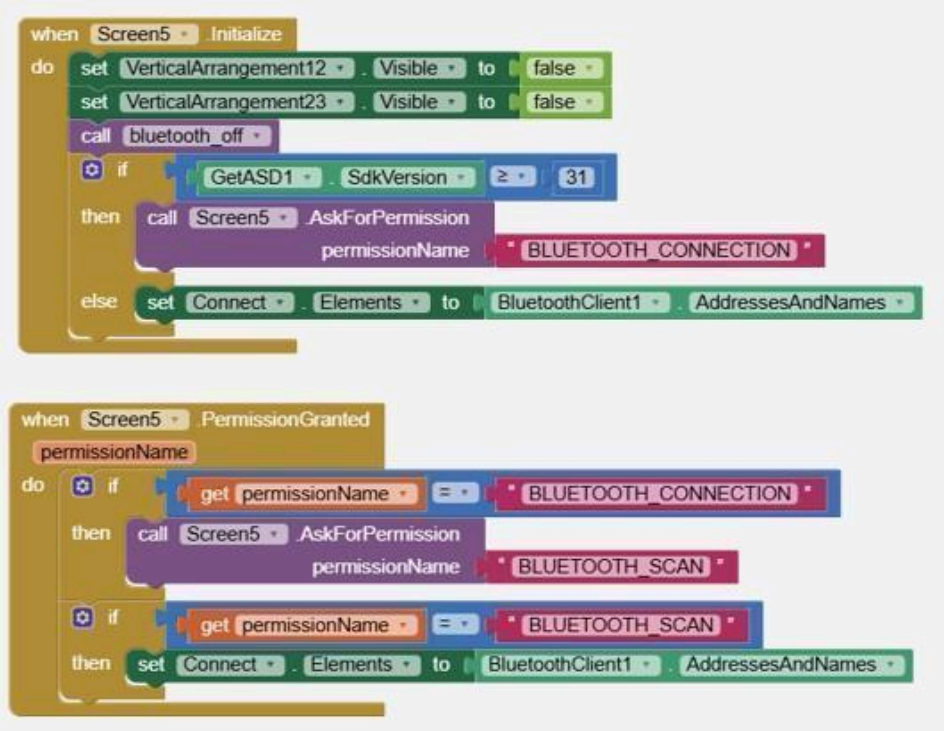

Figure 15: Bluetooth Connection [2]

8. Travel to Destination A

Figure 16: Travel to A

9. Travel to Destination B

Figure 17: Travel to B

10. Travel to Destination C

Figure 18: Travel to C