

**Faculty of Engineering and Technology**

**Department of Computer and Informatics Engineering**

**“University Management System”**

**Submitted in partial fulfillment of the requirement of degree of bachelor in Informatics Engineering**

**BY**

**Christine Rami Elias Abdul Razzaq Amaar Aqmeq**

**Adham Wajih Shaheen**

**Supervised By**

**Dr. Riyad Tayfour**

**Senior Project I**

**2024-2025**

# **Abstract:**

The smart university website represents a comprehensive and integrated electronic platform designed to enhance and modernize academic and administrative processes in line with the requirements of contemporary digital education. The platform simplifies the student experience from registration to completing all academic activities by utilizing advanced technologies that improve operational efficiency and minimize reliance on traditional methods.

The website is tailored to meet the needs of various users, including students, faculty members, and administrators. It combines user-friendliness, speed, and accuracy to provide easy access to services such as course registration, academic progress tracking, and financial management. Additionally, it employs advanced security technologies to ensure the protection of user data and financial transactions.

The platform fosters an interactive learning environment by integrating digital learning tools, such as electronic study materials, student forums, and direct communication channels with professors. Furthermore, it improves the university’s administrative efficiency by offering accurate and real-time data that can be used for informed decision-making.

In summary, the smart university website serves as an effective model for digital transformation in higher education, making it a cornerstone for enhancing the experiences of students and faculty while boosting academic and institutional performance.

**Table of Contents**

[**Abstract:** 2](#_Toc70343793)

[**List of figures:** 4](#_Toc70343794)

[**List of tables:** 5](#_Toc70343795)

[**Table of Abbreviations** 6](#_Toc70343796)

[**Acknowledgement:** 8](#_Toc70343797)

[**شكر :** 8](#_Toc70343798)

[**Chapter 1 Introduction** 9](#_Toc70343799)

[**1.1 Background:** 9](#_Toc70343800)

[**1.2 Problem Statement:** 9](#_Toc70343801)

[**1.3 Objectives** 9](#_Toc70343802)

[**Chapter 2 Methodology (OR) Theoretical study** 10](#_Toc70343803)

[**2.1 Introduction** 10](#_Toc70343804)

[**2.2 Prototype Components** 10](#_Toc70343805)

[**2.2.1 Microcontrollers** 10](#_Toc70343806)

[**2.2.2 Arduino vs Raspberry pi** 11](#_Toc70343807)

[**2.2.4 The speech recognision system** 16](#_Toc70343808)

[***2.2.4.1* Android** 16](#_Toc70343809)

[**2.3 Conclusion** 17](#_Toc70343810)

[**Chapter 3 Implementation, Testing and Results** 18](#_Toc70343811)

[**3.1 Processing steps** 20](#_Toc70343812)

[**3.1.1 Procedure** 20](#_Toc70343813)

[**3.2 TESTING** 22](#_Toc70343814)

[**Chapter 4 Conclusion** 23](#_Toc70343815)

[**4.1 ADVANTAGES** 23](#_Toc70343816)

[**4.2 DISADVANTAGES** 24](#_Toc70343817)

[**4.3 Future Scope** 24](#_Toc70343818)

[**Appendix – Code** 25](#_Toc70343819)

[**References** 26](#_Toc70343820)

[**الملخص:** 28](#_Toc70343821)

# **List of figures:**

[Figure 1 basic block diagram 10](#_Toc70343822)

[When you put a figure you should explain about it in your manuscript. For example you mention: As we see in figure 2 and Figure 3 there are two main types of ….. 11](#_Toc70343823)

[Figure 2 Arduino Uno 11](file:///C:\D\mylect\IUST\2021S\senior%20project%20sample.docx#_Toc70343824)

[Figure 3 LilyPad Arduino 11](file:///C:\D\mylect\IUST\2021S\senior%20project%20sample.docx#_Toc70343825)

[Figure 4 RedBoard 12](file:///C:\D\mylect\IUST\2021S\senior%20project%20sample.docx#_Toc70343826)

[Figure 7 Bluetooth module 13](#_Toc70343827)

[Figure 8 HC-06 14](#_Toc70343828)

[Figure 9 architecture 16](#_Toc70343829)

[Figure 22 robot car from beside 18](file:///C:\D\mylect\IUST\2021S\senior%20project%20sample.docx#_Toc70343830)

[Figure 23 robot car in front 18](file:///C:\D\mylect\IUST\2021S\senior%20project%20sample.docx#_Toc70343831)

[Figure 12 Schema 18](#_Toc70343832)

[Figure 13 BLOCKDIAGRAM 19](#_Toc70343833)

[Figure 14 FlowChart 21](#_Toc70343834)

# **List of tables:**

[Table 1 Arduino vs Raspberry PI 11](#_Toc70343835)

[Table 2 Comparing between different Arduino microcontrollers [6] 12](#_Toc70343836)

[Table 3 HC-06 Pin Configuration 15](#_Toc70343837)

[Table 4 Testing result 22](#_Toc70343838)

# **Table of Abbreviations**

|  |  |
| --- | --- |
| CPU | Central Processing Unit |
| APP | Application |
| AI | Artificial Intelligence |
| MCU | Micro Controller |
| OTP | One Time Programmable |
| OTPROM | One Time Programmable Read Only Memory |
| RAM | Random-Access Memory |
| DSP | Digital Signal Processor |
| USB | Universal Serial Bus |
| OS | Operating System |
| TTL | Transistor Transistor Logic |
| SPP | Serial Port Profile |
| HTC | High Tech Computer |
| AMR | Android Meets Robots |
| AVR | Audio/Video Receiver |
| Wi-Fi | Wireless Fidelity |
| GPS | Global Positioning System |
| LED | Light Emitting Diode |
| ICSP | In Circuit Serial Program |
| EEPROM | Electrically Erasable Programmable Read Only Memory |
| SRAM | Static Random-Access Memory |
| SR | Speech Recognition |
| ASR | Automatic Speech Recognition |
| STT | Speech to Text |
| WER | Word Error Rate |
| SWER | Single Word Error Rate |
| CSR | Command Success Rate |
| PLP | Perceptual Linear Prediction |
| LPC | Linear Predictive Coding |
| HMM | Hidden Markov Model |
| DTW | Dynamic Time Warping |
| ANN | Artificial Neural Networks |
| FFNN | Feed-Forward Neural Network |
| RNN | Recurrent Neural Network |
| BP | Back Propagation |
| FHSS | Frequency Hopping Spread Spectrum |
| VCC | Voltage Common Collector |
| GND | Ground |

# **Acknowledgement:**

.

I would like to extend my sincere gratitude and appreciation to Dr. Hayyan Al-Sibai, Head of the Department, for his constant support and valuable guidance, which played a significant role in the successful completion of this project.

I am also deeply thankful to Dr. Riyad Tayfour, my supervisor, for his unwavering support and insightful mentorship throughout all stages of preparing this thesis, which greatly contributed to achieving these results.

Additionally, I would like to express my gratitude to my colleagues who participated in the project for their fruitful collaboration and outstanding efforts, which were the cornerstone of this achievement.

This work is the result of collective efforts and continuous support from everyone, and I extend my heartfelt thanks and appreciation to all of you.

# **Chapter 1 Introduction**

# **1.1 Background:**

With the rapid advancement of technology and the emergence of digital transformation as a fundamental element across various sectors, the need to develop innovative educational systems that align with the demands of the digital age has become essential. Higher education is one of the areas most impacted by digitalization, as it strives to enhance academic and administrative processes to better meet the needs of students and faculty members.

In the past, universities relied heavily on traditional methods to manage their operations, such as paper-based registration, cash payments, and manual grade issuance. These methods often resulted in slow processes and increased the likelihood of human errors. With the advent of the internet and information technology, universities began adopting Learning Management Systems (LMS) and dedicated electronic platforms to improve the quality of education and administrative services.

The concept of smart university websites emerged as a direct result of these developments. These systems are designed to provide integrated solutions that include course registration, financial management, access to educational materials, and tracking academic performance. These platforms leverage modern technologies such as cloud-based databases, artificial intelligence, and big data analytics to deliver a seamless and secure user experience.

Moreover, the shift to remote learning and the increasing demand for flexibility in educational systems, particularly following the COVID-19 pandemic, have driven universities to adopt advanced digital platforms capable of meeting these requirements. The smart university website has become an essential tool to ensure the continuity of educational processes and foster greater interaction between students and faculty members.

Therefore, the background of this topic stems from the evolving needs of higher education and the impact of technology on improving academic and administrative operations, with a focus on delivering a comprehensive and secure digital educational experience that meets the expectations of the modern era.

## **1.2 Problem Statement:**

In the traditional university system, students, faculty members, and administrative staff face numerous challenges due to outdated systems and inefficient processes. Manual or semi-digital methods for tasks such as course registration, fee payment, access to study materials, and grade reporting are often time-consuming, error-prone, and lack transparency. These issues lead to student frustration, administrative overload, and hinder the overall efficiency of academic operations.

Many universities struggle to develop systems that meet the expectations of a tech-savvy generation. Existing platforms often lack integration, user-friendliness, and the ability to provide a personalized, secure, and seamless user experience.

This project aims to address the need for a Smart University Website, a unified platform designed to overcome these challenges. The platform will streamline essential processes such as course registration, financial management, academic performance tracking, and resource access while ensuring data security and enhancing the user experience. This solution seeks to improve the operational efficiency of universities, provide a more convenient experience for students and faculty, and align higher education with the demands of the modern digital age.

## **1.3 Objectives**

In the traditional university system, students, faculty members, and administrative staff face numerous challenges due to outdated systems and inefficient processes. Manual or semi-digital methods for tasks such as course registration, fee payment, access to study materials, and grade reporting are often time-consuming, error-prone, and lack transparency. These issues lead to student frustration, administrative overload, and hinder the overall efficiency of academic operations.

Many universities struggle to develop systems that meet the expectations of a tech-savvy generation. Existing platforms often lack integration, user-friendliness, and the ability to provide a personalized, secure, and seamless user experience.

This project aims to address the need for a Smart University Website, a unified platform designed to overcome these challenges. The platform will streamline essential processes such as course registration, financial management, academic performance tracking, and resource access while ensuring data security and enhancing the user experience. This solution seeks to improve the operational efficiency of universities, provide a more convenient experience for students and faculty, and align higher education with the demands of the modern digital age.

# **Chapter 2**

# **System Analysis**

## **2.1 Introduction**

Figure 1 shows a simple diagram of the system: The proposed application allows users to view the main page of the University Management System and explore the available services. Users can then create an account, where they will be required to provide relevant information such as academic details, personal preferences, and administrative requirements. This information will be used to process and manage tasks like student registration, course enrollment, exam scheduling, and academic record-keeping efficiently. Additionally, the AI will contribute to optimizing processes such as course recommendations and academic progress tracking to ensure a smooth experience for both students and administration.

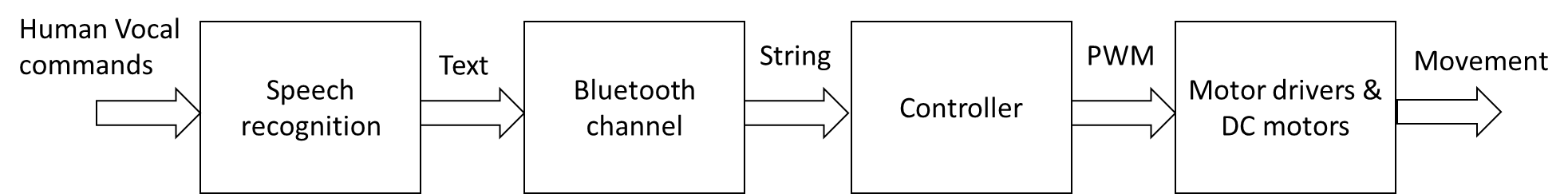


Figure 1 basic block diagram

## **2.2 System Phases**

Web application is divided into three main phases which are :

1. User Interface Design and Development: This phase focuses on creating an intuitive and user-friendly front-end interface for students, faculty, and administrators. It includes the design of dashboards, forms, and navigation systems.
2. Database and Information Management: This involves designing and managing databases to store and retrieve critical information such as student records, course details, grades, and staff data.
3. System Integration and Backend Development: This phase ensures the seamless functionality of the system by integrating all modules, implementing secure login systems, and developing APIs for real-time communication between the database and the user interface

### **2.2.1 User Interface Design And Development :**

React.js:

A JavaScript library used for building interactive and dynamic user interfaces.

In a University Management System, React.js is utilized to develop front-end features such as login pages, dashboards, and student registration pages.

It is known for its reusable components, which streamline development efficiency.

2. GSAP (GreenSock Animation Platform):

A library for creating interactive and visually appealing animations.

Example: Adding fade-in or slide-in effects when navigating between pages like course registration or academic results.

3. AOS (Animate On Scroll):

A library used for adding scroll-based animations.

Example: Displaying sections such as academic programs or announcements dynamically as the user scrolls through the homepage.

4. Axios:

A library for making HTTP requests to fetch data from the backend.

Example: Retrieving student data or course details from the server and displaying them on the dashboard.

5. CLSX:

A utility library for managing dynamic class names.

Example: Dynamically changing button styles based on the registration status (e.g., Registered/Not Registered) in the course page.

6. dotenv:

A library for managing environment variables like API keys or server URLs securely.

Example: Using a .env file to store the backend server's address for secure access.

7. Formik:

A library that simplifies form management in React.js.

Used for creating forms like login, course registration, or personal information updates.

It streamlines input validation and form submission.

8. js-cookie:

A library for handling cookies.

Example: Storing session information such as authentication tokens when the user logs in.

9. React-Router-Dom:

A library for managing navigation between pages in a React.js application.

Example: Navigating between the login page, dashboard, and academic results without reloading the page.

10. React-Hot-Toast:

A library for displaying temporary notifications (toasts).

Example: Showing success messages like "Course registered successfully" or error messages like "Login failed."

11. Redux:

A library for managing the application's state in a centralized way.

In the University Management System, Redux stores user information (name, role, registered courses) and makes it accessible to all components in the application.

12. Redux-Persist:

A library used alongside Redux to save application state in local storage or session storage.

Example: Ensuring user data, like login status, remains intact even after refreshing the page.

13. Yup:

A library for validating input data.

It integrates with Formik to define validation rules for form fields.

Example: Ensuring that the email is in a valid format or that the password meets specific character requirements.

Integration of These Tools:

Login Process:

The login form is built using Formik, with validation handled by Yup.

User credentials are sent to the server using Axios.

The authentication token is stored in a cookie using js-cookie.

A success notification is displayed using React-Hot-Toast.

Page Navigation:

Navigation between system pages is managed with React-Router-Dom.

Scroll animations are added using AOS for a smoother user experience.

User Interface Enhancements:

Redux and Redux-Persist are used to manage and persist user data.

GSAP and AOS improve the overall interactivity with animations and transitions.

These tools collectively create an interactive and seamless user interface for the University Management System website.

### **2.2.2 Database And Information Management :**

in this phase, the focus is on managing, storing, and retrieving critical data for the University Management System. This involves designing efficient database schemas, implementing secure data storage, and enabling real-time data retrieval to ensure the system operates smoothly.

Key Components:

1. Database Design:

The database is structured to store information for students, courses, faculty, and administrative data.

Common entities include:

Students: Stores student details such as name, ID, email, enrollment status, and grades.

Courses: Stores course details like course ID, name, schedule, and credits.

Faculty: Contains faculty profiles, assigned courses, and contact details.

Administration: Manages data related to roles, permissions, and announcements.

2. Data Relationships:

Relational Databases (e.g., MySQL, PostgreSQL):

Use tables with relationships like "one-to-many" (e.g., one faculty can teach many courses) and "many-to-many" (e.g., students enrolling in multiple courses).

NoSQL Databases (e.g., MongoDB):

Useful for large-scale unstructured or semi-structured data, such as logs or user interactions.

3. CRUD Operations:

CRUD (Create, Read, Update, Delete) operations are implemented to handle data management.

Create: Adding new student records, courses, or grades.

Read: Viewing course details or student profiles.

Update: Modifying course schedules or updating student grades.

Delete: Removing old or incorrect data.

4. Security:

Encryption: Ensuring sensitive data like student grades or login credentials are encrypted in transit and at rest.

Authentication and Authorization: Using role-based access control to ensure only authorized users can access or modify specific data.

5. APIs for Data Access:

RESTful APIs: Allow secure communication between the database and the front-end using tools like Axios to fetch data.

GraphQL: Enables flexible data retrieval, allowing users to query exactly what they need.

6. Data Backup and Recovery:

Regular backups are implemented to prevent data loss in case of system failures.

Example Workflow:

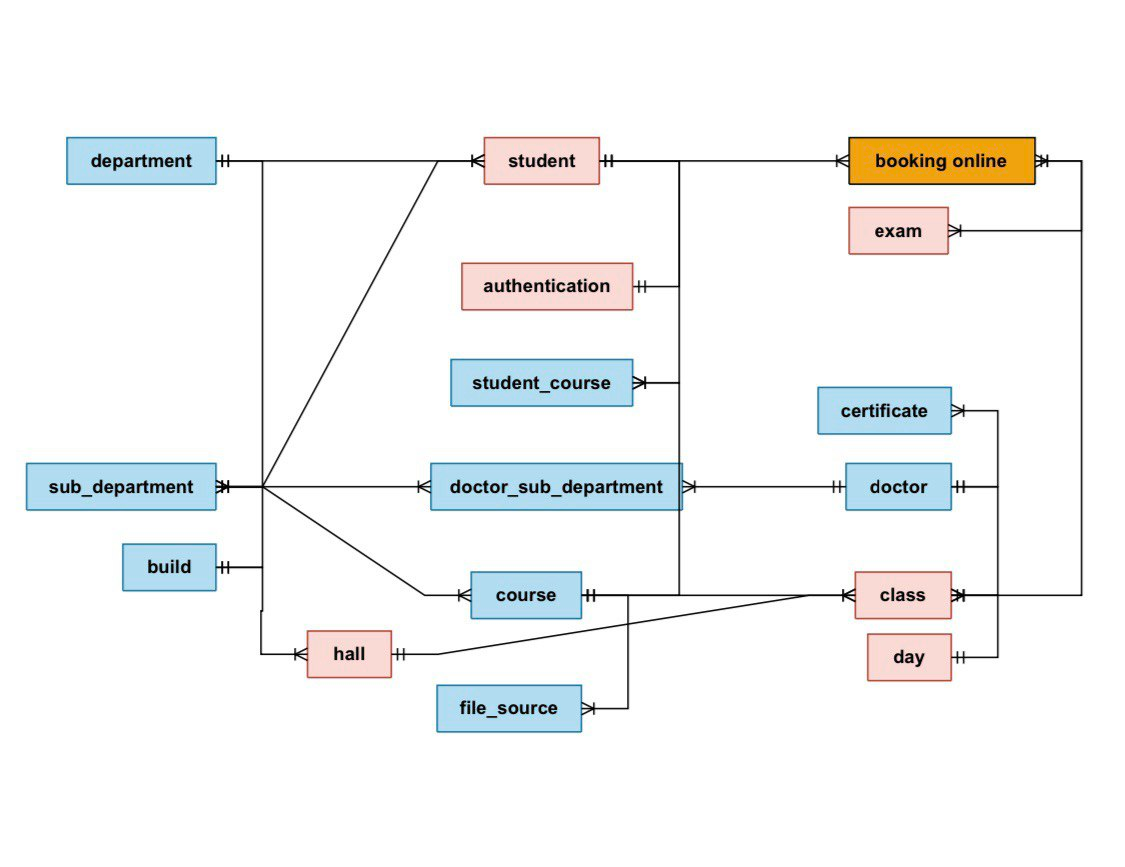
A student logs in and views their enrolled courses.

The front-end sends a request to the backend API using Axios.

The backend queries the database to retrieve the student's courses.

The results are sent back to the front-end and displayed to the student.

Table 1 database



Create the database:

CREATE TABLE department (

   dep\_id SERIAL PRIMARY KEY,

   department\_name VARCHAR(100) NOT NULL,

   build\_number VARCHAR(50) NOT NULL,

   boss VARCHAR(100) NOT NULL

);

CREATE TABLE build (

   build\_id SERIAL PRIMARY KEY,

   build\_number VARCHAR(50) NOT NULL,

   build\_name VARCHAR(100),

   build\_boss VARCHAR(100)

);

CREATE TABLE class (

   class\_id SERIAL PRIMARY KEY,

   course\_id INT NOT NULL,

   doctor\_id INT NOT NULL,

   day\_id INT NOT NULL,

   time\_in TIME,

   time\_out TIME,

   available\_seats INT,

   hall\_id INT NOT NULL,

   description TEXT,

   CONSTRAINT unique\_schedule UNIQUE (course\_id, doctor\_id, day\_id, time\_in, time\_out),

   CONSTRAINT fk\_course FOREIGN KEY (course\_id) REFERENCES course(course\_id) ON DELETE CASCADE,

   CONSTRAINT fk\_doctor FOREIGN KEY (doctor\_id) REFERENCES doctor(doctor\_id) ON DELETE CASCADE,

   CONSTRAINT fk\_day FOREIGN KEY (day\_id) REFERENCES day(day\_id) ON DELETE CASCADE,

   CONSTRAINT fk\_hall FOREIGN KEY (hall\_id) REFERENCES hall(hall\_id) ON DELETE CASCADE

);

#### **2.2.3 System Interface And Backend Development :**

1. Express.js

Express.js is used as a framework to build the server for the application. It integrates all modules by setting up routes to handle user requests efficiently. For instance, routes are configured for login, retrieving data from the database, and file uploads.

2. Bcrypt.js

Bcrypt.js is used to secure the login process by encrypting passwords before storing them in the database. When a new user registers, their password is hashed for protection. During login, the entered password is compared with the hashed one in the database for verification.

3. Multer

Multer is responsible for handling file uploads, such as images or documents. Files are either stored on the application server or sent to external storage services. This step is essential for features like uploading profile pictures or documents in the system.

4. PG (PostgreSQL)

The PG library is used to interact with the PostgreSQL database. It facilitates operations like inserting user data, updating records, or retrieving information. This tool acts as a bridge between the system and the database.

5. CORS

CORS (Cross-Origin Resource Sharing) is used to allow the frontend to communicate with the backend, even if they are hosted on different domains. This ensures smooth integration between the client-side and server-side components.

How Do These Tools Work Together?

Express.js manages all requests between users and the server.

Bcrypt.js secures the login process by hashing passwords.

Multer handles file uploads.

PG manages database communication for storing and retrieving data.

CORS ensures secure communication between the frontend and backend.

These tools collectively ensure the system operates securely and efficiently.

Table 2 tools and functionality

##### **2.2.3.1 MySQL**

MySQL is an open-source relational database management system (RDBMS) widely used to manage data related to students, teachers, courses, and academic and administrative activities in educational institutions. MySQL is known for its ease of use, high performance, and scalability, making it the ideal choice for storing and retrieving data in various educational environments. With features like security, speed, and flexibility, MySQL is considered one of the most popular tools for managing databases in modern educational applications.

Here are some key features of MySQL:

Open Source: MySQL is free to use and open-source, making it highly accessible for developers and businesses.

High Performance: MySQL is known for its fast data retrieval and high performance, which makes it suitable for large-scale applications.

Scalability: It can handle large volumes of data and can scale easily with increased workloads or traffic.

Cross-Platform Support: MySQL works across various platforms, including Windows, Linux, and macOS.

Data Security: MySQL provides strong data security with features such as encrypted connections, user authentication, and access control.

ACID Compliant: MySQL supports ACID (Atomicity, Consistency, Isolation, Durability) properties, ensuring reliable transactions.

Replication: MySQL supports data replication, which helps with data redundancy and load balancing.

Backup and Recovery: MySQL offers tools for data backup and recovery, ensuring data integrity and availability.

Support for SQL Queries: It supports standard SQL queries for managing databases and provides robust tools for querying and managing data.

Extensibility: MySQL supports various storage engines, enabling customization to suit specific use cases.

Community Support: Being open-source, MySQL has a large, active community that offers free support, tutorials, and plugins.

advantages of using MySQL:

Cost-Effective: As an open-source database, MySQL is free to use, which reduces the overall cost for businesses and developers.

High Performance: MySQL is designed to handle large volumes of data quickly and efficiently, making it ideal for high-traffic websites and applications.

Scalability: MySQL can scale horizontally and vertically, allowing it to handle increasing data and user load over time, making it suitable for both small and large applications.

Reliability and Stability: MySQL is known for its reliability, with robust features for data consistency and durability, ensuring a stable database system.

Security: MySQL provides multiple layers of security such as SSL encryption, data access control, and user authentication, ensuring the protection of sensitive data.

Cross-Platform Compatibility: MySQL runs on various platforms (Linux, Windows, macOS, etc.), offering flexibility for developers working in different environments.

Ease of Use: With an intuitive interface and user-friendly tools, MySQL makes it easy to design, manage, and maintain databases.

Large Community Support: The active MySQL community offers vast resources like documentation, tutorials, forums, and plugins, helping developers solve problems quickly.

Regular Updates and Improvements: As an open-source project backed by Oracle, MySQL regularly receives updates, bug fixes, and new features that

Some of the most common use cases include:

*Web Applications*: MySQL is widely used as the database backend for websites and web applications, particularly for dynamic sites that need to store and retrieve user data, such as login credentials, product catalogs, and content management systems (CMS).

*E-Commerce:* Many e-commerce platforms, like Magento or WooCommerce, rely on MySQL to store product information, customer details, order history, and payment data.

*Content Management Systems (CMS):* MySQL is commonly used with CMS platforms such as WordPress, Joomla, and Drupal to store and manage website content, user data, and configurations.

*Enterprise Applications*: MySQL is employed in various enterprise applications to handle large datasets, such as customer relationship management (CRM), enterprise resource planning (ERP), and supply chain management systems.

*Data Warehousing*: In some organizations, MySQL is used for data warehousing, storing large amounts of historical data for analysis and reporting.

## **2.3 Conclusion**

There are numerous tools available for creating a web application, and for our project, we selected the following tools:

For front-end development, I chose React.js, GSAP, AOS, Axios, clsx, dotenv, Formik, js-cookie, react-router-dom, react-hot-toast, Redux, redux-persist, and Yup. These technologies were selected based on their popularity, flexibility, and the benefits they provide in terms of performance, scalability, and ease of integration. React.js allows for dynamic user interfaces, while GSAP and AOS enhance animations. Axios ensures seamless HTTP requests, and Redux along with redux-persist manage application state efficiently. The rest of the tools simplify environment configuration, form handling, validation, and more.

For back-end development, I used Express.js, Bcrypt.js, Multer, PG, and CORS. Express.js facilitates fast and flexible routing, Bcrypt.js provides secure password hashing, Multer handles file uploads, PG supports database operations with PostgreSQL, and CORS enables smooth communication between the frontend and backend. Together, these tools help build a secure, efficient, and scalable web application.

# **Chapter 3 Implementation, Testing and Results**

In this chapter the final product after assembling and connecting all the components that have been discussed and mentioned above, is described. Figures 23 shows the built robot car, the schematic and detailed physical block diagram.

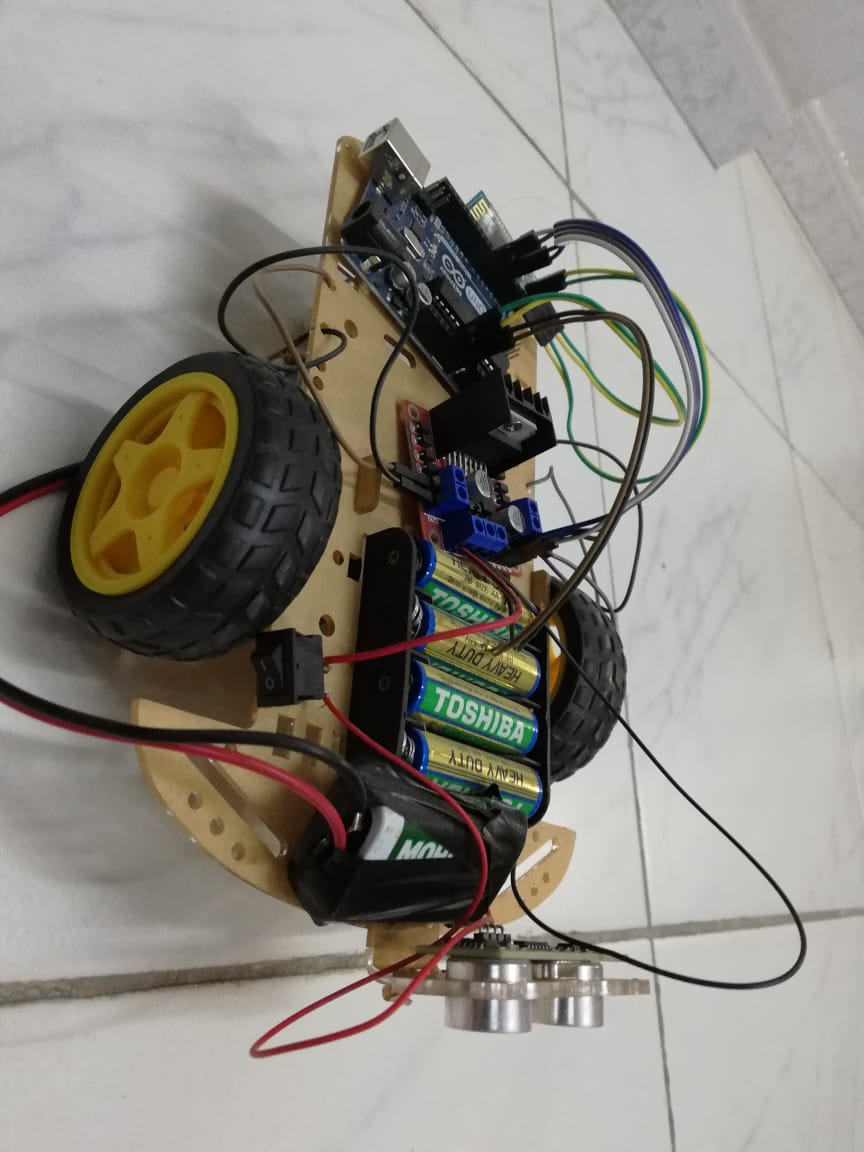
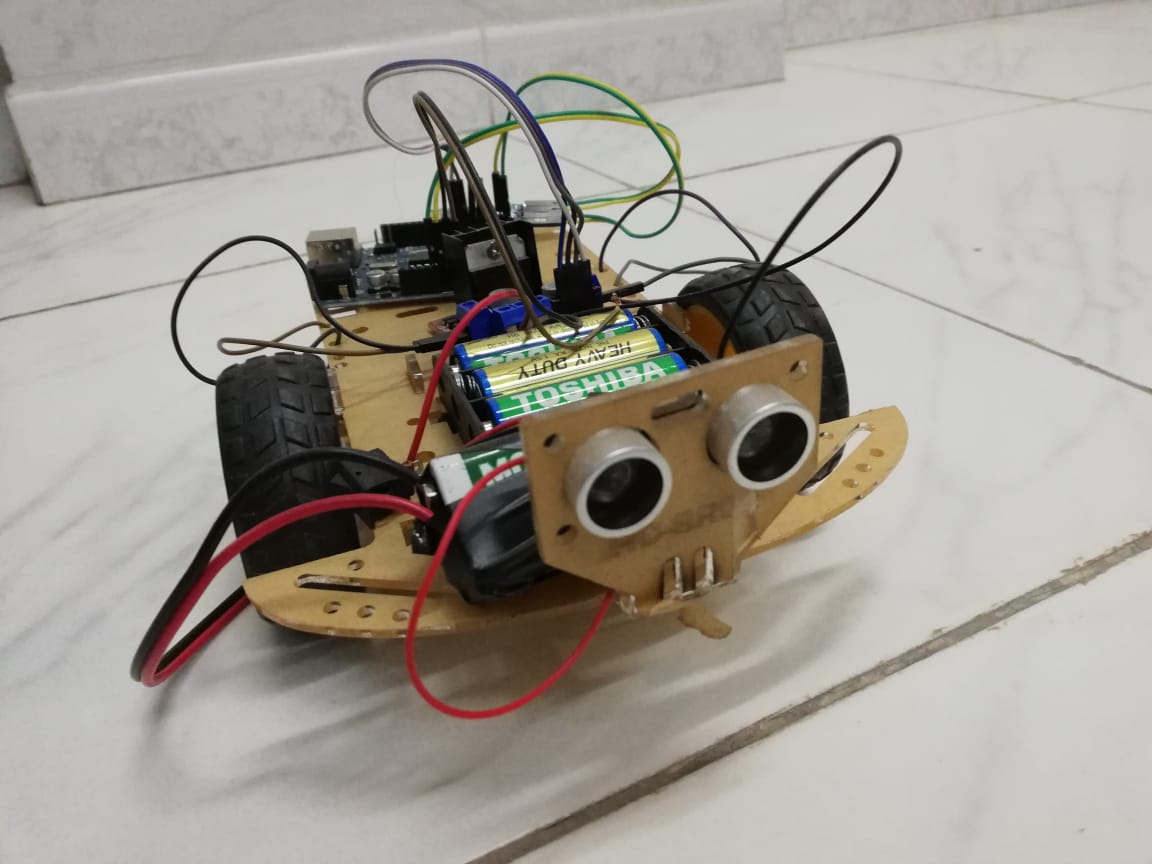


Figure robot car from beside

Figure robot car in front

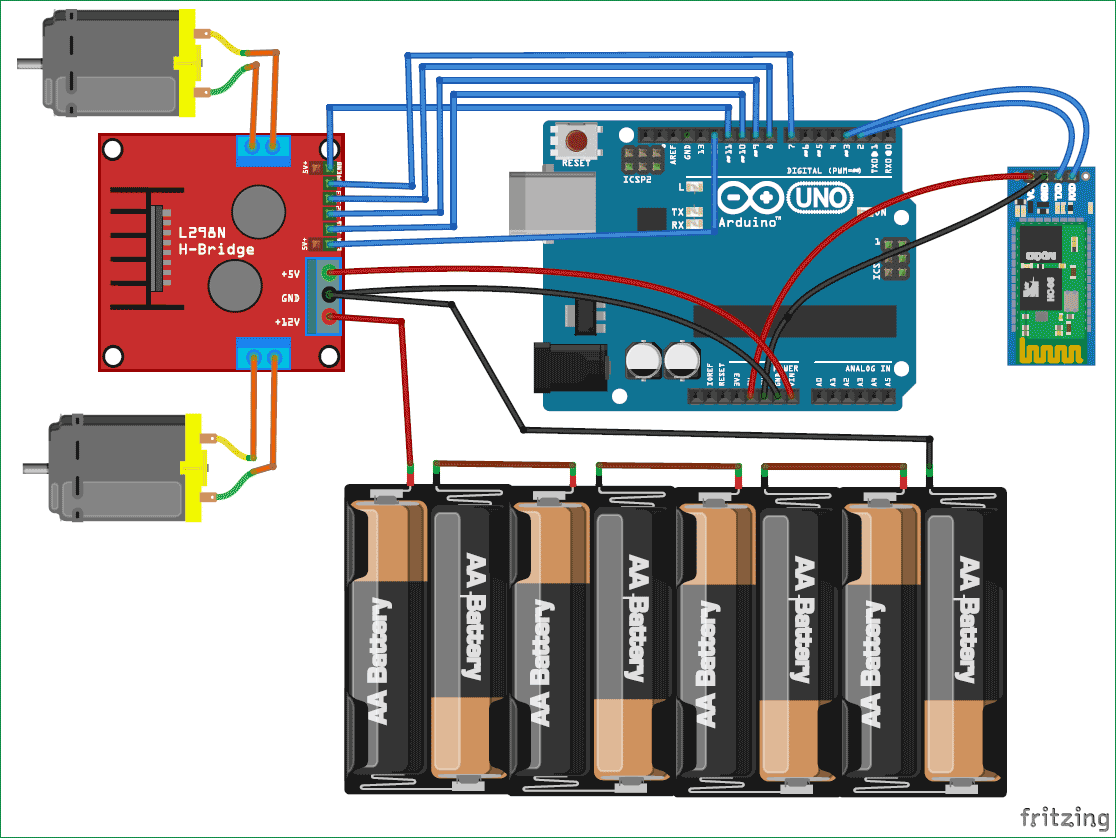


Figure 12 Schema

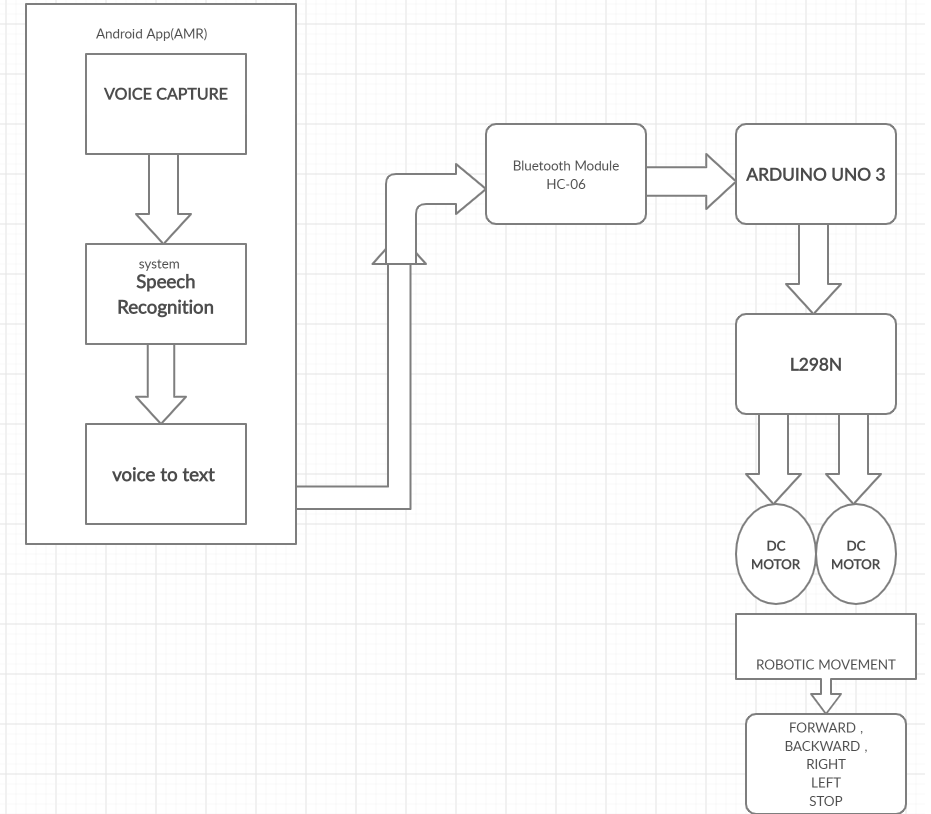


Figure 13 BLOCKDIAGRAM

## **3.1 Processing steps**

As shown in figure 24, the dc motors are connected to to l298n h-bridge, and connected the Bluetooth module hc-06 where the TXD connected to RX in Arduino and RXD to TX in Arduino. And the VCC to 5v and GND to ground. And in1, in2, in3 and in4 in the l298n connected to digital inputs 4, 5, 6 and 7 respectively in the Arduino. And from Arduino vin connected to 5v slot in l298n and GND from Arduino to GND in l298n and connected the positive wire of the battery 9v to 12v slot in l298n and the negative wire to GND in l298n.

### **3.1.1 Procedure**

Here are the steps for how to use android application to control the robotic vehicle.

1. Open AMR The Voice
2. Check once you get started with the application, the Bluetooth of the mobile is automatically enabled.
3. Click on “connect robot” option present in options menu.
4. Now select HC-06 to get paired with the module. After pairing it is ready to use.
5. Click the Mic Icon to start speaking
6. When you speak “Go ahead” your speech gets recognized and converted into text. That text is transferred to robot through Bluetooth HC-06.
7. The robot receives the string, decodes it and compares it with the Instructions that are described in the program and moves the robot in forward direction.
8. The same in the case for the other commands.
9. As per command given from android app, motor is drive in up, down, left, right and stop in this way

The bellow figure shows the flow chart of the procedure steps

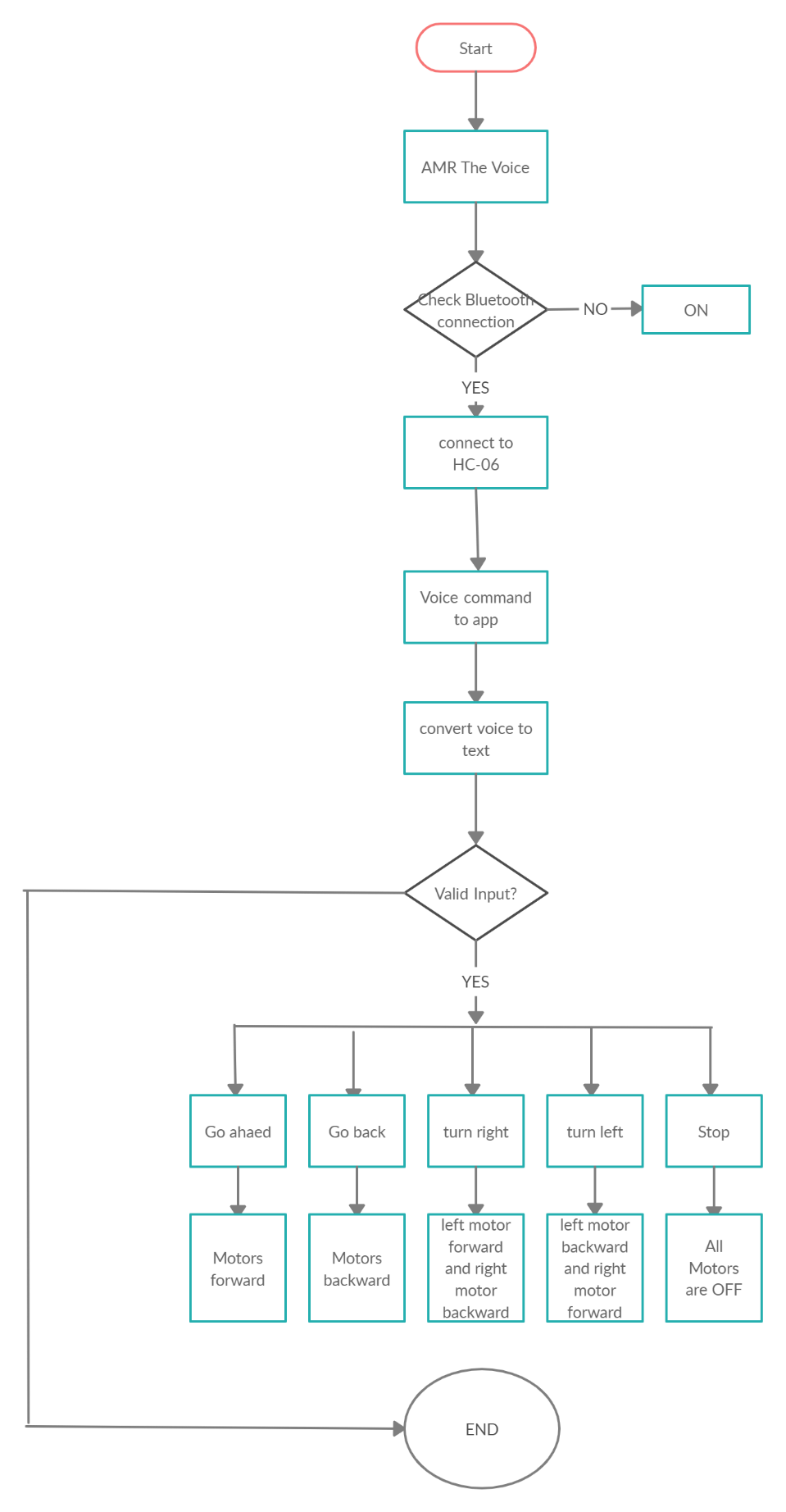


Figure 14 FlowChart

## **3.2 TESTING**

From the following commands our robot response is .

Table 4 Testing result

|  |  |
| --- | --- |
| **COMMAND** | **ROBOT MOVEMENT** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Add statistics like confusion matrix and calculate recall, precision…

# **Chapter 4 Conclusion**

Ahighly reliable and easy system to accomplish a purpose design specific task such as distribution of medicine and food to the bed ridden patients specially in infected & inaccessible areas of the hospitals and medical Centre have been reported. The on-board intelligence helps providing situational awareness a basic requirement of the system to be operated by voice / tele confined for ascertaining a majority of other tasks in open loop environment. The operation by voice command could best be used for handicapped.

The outcome of the thesis is a simple robot which is controlled by a smart android phone& also receives the voice commands. This thesis aims to provide simple guidelines for people interested in building robots. As mentioned earlier, the project has been carried out several times and the aim of this thesis is to familiarize the students with fundamentals of Arduino and Android to build anything possible. Although the thesis projects very little about the robot’s use in real world, but with the help of guidelines and the abundance of resources the outcome could be very beneficial for many people in the world. People with physical limitations such as handicapped people could use the feature to their wheel chair from this thesis to compensate their abilities.

## **4.1 ADVANTAGES**

* By using robots we can control live video feed by giving voice commands.
* We can use voice control robot for a multiple ways. We can move any clockwise, anticlockwise direction, forward and backward by giving voice commands.
* In industries we can control the machines by using robots.
* You can program them to make them do exactly what you want them to do.
* They are more accurate than humans Eg no shaking when in a very important surgery, puts every screw in fabricating a car etc.
* You can send them to very dangerous places.
* You can make them do your job for you.

## **4.2 DISADVANTAGES**

* They are very expensive to make they can reproduce but it could cost money for the materials.
* You need the right materials to make them.
* They can be very hard to program.
* You need highly trained people to make them.
* People can lose jobs in factories.
* It needs maintenance to keep it running.

## **4.3 Future Scope**

The purpose of this robot is to help people for transferring goods in a dangers area and to keep in safe during the corona virus COVID-19. So, to do this. I’ll program ultrasonic sensor that keeps the robot in the guidance with avoiding obstacles. And I will develop a mobile application using MIT APP INVENTOR instead of using AMR the Voice.

# **Appendix – Code**

const int LEFTF = 7; // left forward wheel intilize

const int LEFTB = 6; // left backward wheel intilize

const int RIGHTB = 5; // right backward wheel intilize

const int RIGHTF = 4; // right forward wheel intilize

String voice;

void setup() {

Serial.begin(9600); // start the communication with bluetooth

pinMode (LEFTF, OUTPUT);

pinMode (LEFTB, OUTPUT);

pinMode (RIGHTB, OUTPUT);

pinMode (RIGHTF, OUTPUT);

// put your setup code here, to run once:

}

void loop() {

}

# **References**

[1] F. C. P. David Calcutt, 8051 Microcontrollers, 2004.

[2] "https://en.wikipedia.org/wiki/ATmega328," ATMega328, 10 may 2020.

[3] jayant, "arduino vs rasbperry pi," https://circuitdigest.com/article/arduino-vs-raspberryp-pi-difference-between-the-two.

[4] arduino, http://arduino.cc/en/.

[5] Ben, what is Arduino, Sparkfun, learn.sparkfun.com/tutorials/what-is-an-arduino/all#whats-on-the-board.

[6] A. comparison, https://learn.sparkfun.com/tutorials/arduino-comparison-guide/all?print.

[7] hc-05, "https://www.electronicwings.com/sensors-modules/bluetooth-module-hc-05-".

[8] H.-0. HC-04, "http://smarpl.com/content/bluetooth-module-hc04hc06-configuration-tool".

[9] hc-06, "https://components101.com/wireless/hc-06-bluetooth-module-pinout-datasheet".

[10] L. Guangzhou HC Information Technology Co., Bluetooth HC-06 Datasheet.

[11] android, "https://en.wikipedia.org/wiki/Android\_(operating\_system)#:~:text=Android%20is%20a%20mobile%20operating,such%20as%20smartphones%20and%20tablets.&text=Some%20well%20known%20derivatives%20include,wearables%2C%20both%20developed%20by%20Google.".

[12] a. m. robot, "https://amr-voice.en.aptoide.com/app".

[13] H. Automation, "https://en.wikipedia.org/wiki/Home\_automation".

[14] T. S. ,. K. Masato Nishimori, Voice Controlled Intelligent Wheelchair, 17-20 Sept. 2007.

[15] https://learn.sparkfun.com/tutorials/what-is-an-arduino/all, "Arduino".

[16] servomotor, "https://components101.com/servo-motor-basics-pinout-datasheet".

[17] L298n, "https://opencircuit.shop/Product/L298N-motor-driver-module-H-Bridge".

[18] A. family, "https://learn.sparkfun.com/tutorials/what-is-an-arduino/all".

[19] e. a. Wilson, "Servo Motor - Types and Working Principle", Electronics Hub, www.electronicshub.org/servo-motors/., 24 Dec. 2017.

[20] ANN, http://www.jclinepi.com.

[21] A. advantages, http://www.researchgate.net.

[22] http://en.wikipedia.org/wiki/Artificial\_neural\_network.

[23] J. Tebelskis, Speech Recognition using Neural Networks, May 1995.

[24] S. Recognition, http://en.wikipedia.org/wiki/Speech\_recognition..

[25] National Institute of Standards and Technology, The History of Automatic Speech Recognition Evaluation.

[26] L. X. ,. K. C. B. R. Deng Y., ''Continuous Feature Adaptation for Non-Native Speech Recognition, Li X. , Kwan C.,Raj B.,Stern R., ''Continuous Feature Adaptation for Non-Native Speech Recognition'', International Journal of Computer, Information Science and Engineering, No:6, 2007..

[27] SP, http://en.wikipedia.org/wiki/Speech\_recognition..

[28] Arduino, arduino software http://arduino.cc/en/Main/Software.

[29] "ANN," http://en.wikipedia.org/wiki/Artificial\_neural\_network.

# **الملخص:**

نكتب هنا ملخص عن كل فصول الاطروحة جملة أو جملتين مقدمة مع تلخيص المشكلة وأهداف المشروع، جملتين أو ثلاثة عن الفكرة النظرية والأدوات المستخدمة، جمليتين أو ثلاثة عن التنفيذ، جملة أو جملتين عن التجريب والنتائج.



**كلية الهندسة والتكنلولوجيا**

**قسم هندسة الحاسوب و المعلوماتية**

**"اسم المشروع"**

**يغطي هذا البحث الجزء من متطلبات نيل شهادة البكالوريوس في هندسة الحاسوب**

**اعداد**

**اسم الطالب**

**بإشراف**

**د.محمد حيان السباعي**

**مشروع التخرج الأول**

**2021-2022**