

IoT Based Library Management System

“ ILM ”



Syed Wasi Hassan Bokhari

Mubashir Maqbool

Ali Usman Bin Atteeq

Hamza Chaudhary

Supervisor

Asst. Prof. Dr. Abdul Wakeel

Co - Supervisor

Asst. Prof. Dr. Mir Yasir Umair

A thesis submitted to the Faculty of Department of Electrical Engineering,
Military College of Signals, National University of Sciences and Technology,
in partial fulfillment for the requirements of B.E Degree in Electrical Engineering.

JULY, 2021

IoT Based Library Management System

“ ILM ”



Syed Wasi Hassan Bokhari

Mubashir Maqbool

Ali Usman Bin Atteeq

Hamza Chaudhary

Supervisor

Asst. Prof. Dr. Abdul Wakeel

Co - Supervisor

Asst. Prof. Dr. Mir Yasir Umair

A thesis submitted to the Faculty of Department of Electrical Engineering,
Military College of Signals, National University of Sciences and Technology,
in partial fulfillment for the requirements of B.E Degree in Electrical Engineering.

JULY, 2021

CERTIFICATE OF CORRECTNESS & APPROVAL

This is hereby certified that the work contained in this thesis titled -

“IoT Based Library Management System (ILM) ”

is carried out by

Syed Wasi Hassan Bokhari, Mubashir Maqbool

Ali Usman Bin Atteeq , Hamza Chaudhary

under the supervision of **Dr. Abdul Wakeel** and **Dr. Mir Yasir Umair**

for partial fulfillment of Degree of Bachelor of Electrical Engineering, in Military College of

Signals (MCS), National University of Sciences and Technology(NUST), Islamabad

during the academic year, 2020-2021 is correct and approved. The material that has been

used from other sources it has been properly acknowledged/referred.

Approved by

Signature : _____

Supervisor: **Asst. Prof. Dr. Abdul Wakeel**

Signature : _____

Co-Supervisor: **Asst. Prof. Dr. Mir Yasir Umair**

Date:_____

DECLARATION

It is hereby declared that no portion of work presented in this thesis has been submitted in support of another award or qualification in either this institute or anywhere else.

Plagiarism Certificate (Turnitin Report)

This thesis has been checked for Plagiarism. Turnitin report endorsed by the supervisor is attached.

Syed Wasi Hassan Bokhari

208597

Mubashir Maqbool

223283

Ali Usman Bin Atteeq

207435

Hamza Chaudhary

213737

Signature : _____

Supervisor: **Asst. Prof. Dr. Abdul Wakeel**

Signature : _____

Co-Supervisor: **Asst. Prof. Dr. Mir Yasir Umair**

Acknowledgments

We are thankful to our Creator Allah Almighty for showering His blessings upon us throughout the project during these difficult times. We could have done nothing without His divine help.

We are thankful to our beloved parents for their continuous support, encouragement, and patience throughout the project.

We would like to thank our supervisors, Dr. Abdul Wakeel and Dr. Mir Yasir Umair, for providing us with tremendous support and cooperation throughout our final year project. This project would not have been possible without their input and support. To work under their guidance was a great privilege and honor for all the syndicate members.

Lastly, we would like to thank all our friends who provided valuable assistance and help to complete this project in the times of the Covid -19 pandemic. Special thanks to all of you this would not be possible without you!



Dedicated to our exceptional parents, respected teachers, adored siblings, friends, and family whose tremendous support and warm encouragement led us to this wonderful accomplishment.

Abstract

An IoT-based Library Management System is designed for the efficient management of books in the library. The system is fully automated and facilitates both the librarian and the user/student. An RFID card is issued to the library members which is scanned at the entrance.

A flutter mobile application is developed to have access to the complete database of the library. The user can search for books from the library through this application. The books of interest can be reserved from home as well. Each book has an RFID tag and is RFID scanned at the self-checking counter. It is automatically issued to the students for a specific period. The mobile application also keeps the record of the books issued, to whom they are issued, and fines in case of a delay in return. Messages are automatically generated on the return date of books. When returned, the book is again RFID scanned, and the database is being updated.

The system has been thoroughly tested and its reliability is guaranteed. It has made the library user's and staff's tasks easy. The system is smart, convenient, and practical with a high degree of automation. It can be easily implemented in schools, colleges, private and public libraries, and other organizations.

Key Words: *Library Management System, RFID, IoT, Flutter Mobile Application*

Table of Contents

CERTIFICATE OF CORRECTNESS & APPROVAL	i
DECLARATION	iii
Plagiarism Certificate (Turnitin Report)	iv
Acknowledgments	v
Abstract	vii
Table of Contents	viii
List of Figures	xi
List of Tables	xiii
1 CHAPTER 1: INTRODUCTION	1
1.1 Overview.....	1
1.2 Problem Statement	2
1.3 Objectives.....	2
1.3.1 Academic Objectives.....	2
1.3.2 Industrial Objectives.....	2
1.4 System Methodology	2
1.5 Scope of Project	3
1.6 Deliverables	3
1.6.1 Record-Keeping Module	3
1.6.2 Entry and Exit Gate	3
2 CHAPTER 2: LITERATURE REVIEW	4
2.1 Scholarly Articles:.....	4
2.2 Market Research	5
3 CHAPTER 3: HARDWARE AND SOFTWARE DESCRIPTION	7
3.1 Project Hardware.....	7
3.1.1 ESP-32 Micro-controller	7
3.1.2 RC-522 RFID Reader.....	9
3.1.3 Arduino Uno.....	12
3.1.4 DC Motor 12V.....	14
3.1.5 L-298N Motor Driver	15
3.2 Project Software.....	17
3.2.1 Android Studio Ide	18
3.2.2 Visual Studio Code.....	19
3.2.3 Arduino Ide	19
3.2.4 Proteus.....	20
3.2.5 Thonny	21

3.3	Database and Cloud Services	22
3.3.1	Google Firebase.....	22
4	CHAPTER 4: MOBILE APPLICATION AND THE DATABASE	23
4.1	Overview.....	23
4.2	Admin Side	23
4.2.1	Login Page.....	23
4.2.2	Dashboard.....	24
4.2.3	Available Books	25
4.2.4	Add Books.....	25
4.2.5	Delete Books	26
4.2.6	Issued Books	26
4.2.7	Issued Books - (Details)	27
4.2.8	Issued Books - (Returned).....	27
4.2.9	Registered Users.....	28
4.2.10	Overdue Books – (Generation of Fine)	28
4.2.11	Overdue Books – (Fine Paid)	29
4.2.12	Logout	29
4.3	User Side	30
4.3.1	Signup.....	30
4.3.2	Sign in	30
4.3.3	Available Books	31
4.3.4	Issued Books	32
4.3.5	Notification of Fine	32
4.3.6	User Profile - (Logout).....	33
4.4	Firebase Database of Application	33
4.4.1	User Authentication on Firebase	33
4.4.2	Cloud Firestore.....	34
5	CHAPTER 5: HARDWARE DESIGN	35
5.1	Radio Frequency Identification (RFID)	35
5.1.1	Passive Systems.....	36
5.1.2	Active Systems.....	36
5.2	Book Issue and Return	37
5.2.1	Overview	37
5.2.2	Block Diagram	37
5.2.3	ESP-32 AND RC-522	38
5.2.4	ESP-32 and Firestore.....	40
5.3	Gate Entry and Exit.....	44

5.3.1	Overview	44
5.3.2	Block Diagram	44
5.3.3	Arduino Uno and Rc-522	45
5.3.4	Arduino Uno, L-298N and Dc Motor	47
6	CHAPTER 6: CONCLUSION.....	51
6.1	Future Work	51
	APPENDIX A.....	52
	APPENDIX B.....	53
	REFERENCES	58

List of Figures

Figure 3.1: ESP-32 Microcontroller	7
Figure 3.2: ESP-32 Pinout Description	9
Figure 3.3: RC-522 Module.....	9
Figure 3.4: RC-522 Pin Configuration	11
Figure 3.5: Arduino UNO Microcontroller.....	12
Figure 3.6: Arduino UNO Pin Configuration	14
Figure 3.7: 12V DC Motor	14
Figure 3.8: L-298N Motor Driver.....	15
Figure 3.9: L-298N Driver Pinout	17
Figure 3.10: Android Studio IDE Logo.....	18
Figure 3.11: VS Code Logo.....	19
Figure 3.12: Arduino IDE Logo	20
Figure 3.13: Proteus Logo	20
Figure 3.14: Thonny Logo.....	21
Figure 3.15: Firebase Logo.....	22
Figure 4.1: Sign-in Page	24
Figure 4.2: Dashboard	24
Figure 4.3: Available Books Page	25
Figure 4.4: Add Books Page.....	25
Figure 4.5: Delete Books Page	26
Figure 4.6: Issued Books Page	26
Figure 4.7: Details of Issued Books.....	27
Figure 4.8: Issued Books (Returned) Page	27
Figure 4.9: Registered Users Page.....	28
Figure 4.10: Overdue Books Page	28
Figure 4.11: Overdue Books (Fine Paid) Page	29
Figure 4.12: Logout Page (Admin Side).....	29
Figure 4.13: Signup Page.....	30
Figure 4.14: Sign-in Page	31
Figure 4.15: Available Book (User Side)	31
Figure 4.16: Issued Books (User Side).....	32
Figure 4.17: Notifications Page	32
Figure 4.18: Logout (User Side) Page	33
Figure 4.19: Firebase User Authentication.....	34
Figure 4.20: Cloud Firestore.....	34
Figure 5.1: Block Diagram	38
Figure 5.2: Circuit Diagram.....	39
Figure 5.3: Pin configuration using micro-python.....	39
Figure 5.4: UID and Type generation	40
Figure 5.5: Wi-Fi connection.....	41
Figure 5.6: NTP server connection.....	41
Figure 5.7: Base URL of Rest API	42
Figure 5.8: Get Request.....	42
Figure 5.9: Patch request for book issue.....	42
Figure 5.10: Patch request book return.....	43
Figure 5.11: Firestore Database.....	43
Figure 5.12: Fine Information	44
Figure 5.13: Block Diagram	44
Figure 5.14: Arduino UNO and RC-522 circuit	45

Figure 5.15: Pin Configuration in Arduino IDE	46
Figure 5.16: UID Generation	47
Figure 5.17: Pulse Width Modulation	48
Figure 5.18: H-bridge	48
Figure 5.19: Arduino UNO, L-298N, and DC motor circuitry	49
Figure 5.20: Pin Configuration in Arduino IDE	49
Figure 5.21: Authorized Access	50
Figure 5.22: Unauthorized Access	50

List of Tables

Table 2-1: A list of Library automation systems using RFID.....	4
Table 3-1: Specifications of ESP-32 Microcontroller[12].	8
Table 3-2: RC-522 Specifications.....	10
Table 3-3: RC-522 Pin Descriptions.....	11
Table 3-4: RC-522 Pin Descriptions.....	13
Table 3-5: L-298N Specifications.....	16
Table 3-6: L-298N Pin Description[16].....	17
Table 5-1: Operating Frequencies of RFID [20].....	36
Table 5-2: Pin Configuration	46

CHAPTER 1: INTRODUCTION

Internet of Things (IoT) Based Library Management System i.e., ILM is a management system for libraries that is an amalgam of the Radio Frequency Identification (RFID) technology and flutter/dart cross-platform. ILM is a step into the modern world of innovation and IoT to cope with the rapid technological changes around us. The trend of technologies like IoT is increasing enormously and this point of pondering leads us to develop a modern management system for the libraries to innovate the educational sector of the country.

ILM overcomes all deficiencies and flaws of the previous old management systems and serves as the step to digitize the library. ILM is also a small contribution towards betterment in terms of the technological aspect. Education is of extreme importance in our society and one of the main sources of education and learning is the library. ILM helps us to access the library more easily and efficiently.

ILM facilitates the librarian and library members and reduces human effort. ILM is an automated system that finds its domain in the internet of things. The complete description of the prototype is discussed onwards in this report.

1.1 Overview

ILM is an IoT-based Smart Library System that uses RFID technology for its implementation. The system will be fully automated and will facilitate both the librarian and the user/student. An RFID card will be issued to the students which will be scanned at the entrance.

A mobile application will be developed to have access to the complete database of the library. The user can search for books from the library through this application. The books of interest can be reserved from home also. Each book will have an RFID tag and will be RFID scanned at the self-checking counter. It will be automatically issued to the students for a specific period. The application will also keep the record of the books issued, to whom issued, and fines in case of a delay in return. Messages will be automatically generated on the return date of books. When returned the book will again RFID scanned, and the database will be updated.

1.2 Problem Statement

In a traditional library system, there is no use of technology. Everything is based on a card and all the records of books are manually recorded in a register which has increased the material handling and made the work very hectic. For students to find the desired book in a big library is a very time-consuming task. There is no information if a book present is issued to someone or not. Also, there is no proper system to manage late returns. Both librarians and students do not have a proper and efficient system.

1.3 Objectives

The project has been selected keeping in mind the following objectives :

1.3.1 Academic Objectives

- Understand and develop IoT-based applications.
- Usage of RFID tags.
- Mobile application development.
- Microcontroller application development.

1.3.2 Industrial Objectives

- To use IoT and RFID technology to develop an application for the library.
- To have access to the library's database from anywhere.
- To automate issuance and return of books.
- To limit unauthorized entry into the library.
- Database management of the library.
- Generation of messages, fines, and alarms for late return.

1.4 System Methodology

The "ILM" system is based on IoT and RFID technology. The issuance and return of books are based on RFID. In an RFID system, a small transceiver (a transmitter and receiver) in the form of a tag or card working at radio frequency is present. When the tag or the card has a frequency

within the radio frequency transmitted from the reader, an acknowledgment is sent back. The tag or the card can therefore be scanned and identified by the system. There is a unique RFID tag attached to each book and unique RFID cards are given to students. Only authenticated students are allowed to enter the library and as there is an RFID-based entry.

The data of the books and the students are read by the RFID reader and are stored in the database using microcontroller ESP-32. There is a mobile application that has all the data regarding the books that are placed in the library. Only authenticated users can be registered in the app, they can reserve the book and later they can get the book from the library. IoT is used for the connection between the library database and the application database. It provides the linkage between our app and our library database using cloud services. IoT is, therefore, the bridge between mobile application and hardware.

1.5 Scope of Project

The system will be an intelligent library management system that will create a better quality of service with quick and effective benefits to both the librarian and the students/users. It will be easily applicable to either a small departmental library or a vast university library.

1.6 Deliverables

The final deliverable consists of two hardware modules and a mobile application.

1.6.1 Record-Keeping Module

This module includes an ESP-32 microcontroller circuit with RFID Reader RC-522 for issuance and return of books. Each student will scan a unique library RFID card that can issue or return the book and the database will be immediately updated accordingly.

1.6.2 Entry and Exit Gate

A mobile application is developed that has a database of all the books and registered users. Library members can view the available books, issued books, and any fines due from the mobile application. The application is catering to the needs of both admin and students.

The admin side of the librarian will have the ability to add or remove users and books into the database. Fines and alarms generation will also be through the application.

CHAPTER 2: LITERATURE REVIEW

In this chapter, the research related to the project which has helped in understanding and implementing the project is discussed. The research has been divided into two major portions.

- › Scholarly articles or research papers related to the automation of the library.
- › Market Research

2.1 Scholarly Articles:

For a better understanding of RFID technology and its utilization, several research papers have been studied. A brief overview of these research articles is given in Table 2-1.

Table 2-1: A list of Library automation systems using RFID.

Studies	Year	Title	Methodology
J. W. Choi <i>et al.</i> [1]	2006	Affordable Library Search System	RFID
U. Mamatha [2]	2009	Library Management System	RFID
M.I.Younis [3]	2012	A Smart Library Management System	RFID
D.Y. Li <i>et al.</i> [4]	2016	Design of Internet of Things System for Library Materials Management Using UHF RFID	RFID &UHF
J. Pandey <i>et al.</i> [5]	2017	A Study on Implementation of Smart Library Systems using IoT	RFID & IoT
Q.Huang & H.Huang [6]	2019	Intelligent Electronic Management of Library by Radio Frequency Identification Technology	RFID
N. Malipatil <i>et al.</i> [7]	2020	RFID Based Library Management System	RFID and GSM

An affordable RFID-based library management system was presented in [1]. The system facilitated the location of books in the library. In its first version RFID antennas were placed on shelves to capture information about the placement of the books. In the updated version, the antennas were replaced by cheap tags.

In [2], a library management system was proposed that used RFID technology with low-cost passive tags. The system was low-cost and reduced the burden of both the library users and the staff.

An application was developed in [3] for the libraries in Oman. The system used both RFID and IoT technology. The record of books and their search was efficiently automated.

In [4], a library management system was designed to automate Mehran University Library. A user-friendly interface was developed to provide the search of books. Reports and reminders were efficiently generated. The system incorporated the RFID technology.

Internet was incorporated with RFID to manage a library in [5]. The RFID reader Motorola MC9090 was used that could read any kind of RFID tag. Many problems like misplaced books, their search issue, and return were incorporated.

In the research papers [6–10], the combination of RFID technology and IoT was incorporated for the management of libraries. Different methodologies were applied to efficiently reduce the burden of library staff and users. All systems automated the library record.

These research papers related to RFID and Automated libraries helped us to choose the best equipment and design method to implement our desired library system. The system will be cost-effective and most importantly made in Pakistan. It will provide all the aspects that a good library system should have and will be a step towards the digitization of Pakistan.

2.2 Market Research

The objective of this research was to find if library automation systems have been implemented somewhere in our country and how a better system can be designed than the existing systems. Their drawbacks and deficiencies were also to be investigated.

It was found that currently in Pakistan there are three companies which are providing automation of library.

- › SKU Technologies
- › RFID Pakistan
- › Digital Data System

The main products of these companies are automation systems, and they also include library automation. The companies like SKU technologies and RFID Pakistan only provide automation with RFID technology but there is no Android or iOS-based application. Digital Data Systems provides the application with RFID technology, but it is too expensive. These systems are

not manufactured in Pakistan as all the equipment used in these systems is imported from different countries and is only assembled here.

Therefore, there was a need to design a reliable, cost-effective library system using local manpower and resources.

CHAPTER 3: HARDWARE AND SOFTWARE DESCRIPTION

3.1 Project Hardware

Analysis and research of different IoT-based automated systems made and published by several individuals including students and professionals in IEEE and other authentic well-reputed online platforms were carried out. The valuable advice of our respected supervisors was also considered for the decision of the hardware equipment of the project. The project comprises of following hardware equipment:

- › ESP-32 Micro-controller
- › RC-522 RFID reader
- › Arduino UNO
- › DC Motor 12V
- › L-298N Motor Driver

3.1.1 ESP-32 Micro-controller

3.1.1.1 Description

The ESP-32 Microcontroller is a powerful yet cost-effective 32-bit microcontroller [11]. The micro-controller is equipped with integrated Wi-Fi in addition to a full TCP/IP stack for establishing internet and Bluetooth connection. It is a low power microcontroller that bridges Arduino with Wi-Fi, therefore is well suited for IoT projects.



Figure 3.1: ESP-32 Microcontroller

3.1.1.2 Features

The features of the ESP-32 micro-controller include:

- › Operates at a frequency of 2.4 GHz having a data rate of 54Mbps.
- › Bluetooth support in addition to an onboard antenna.

3.1.1.3 Specifications

Some of the specifications of the ESP-32 microcontroller are listed in Table 3-1:

Table 3-1: Specifications of ESP-32 Microcontroller[12].

Operating Voltage	2.2 V to 3.6 V
GPIO	36 ports
ADC	14 ports
DAC	2 ports
Flash Memory	16 Mbyte
SRAM	250 Kbyte
Clock Speed	Up to 240 MHz
Wi-Fi	2.4 GHz

3.1.1.4 Applications

The ESP-32 microcontroller is being used in a wide range of applications such as home automation and smart devices used in the medical and energy sectors.

3.1.1.5 Pin Configuration

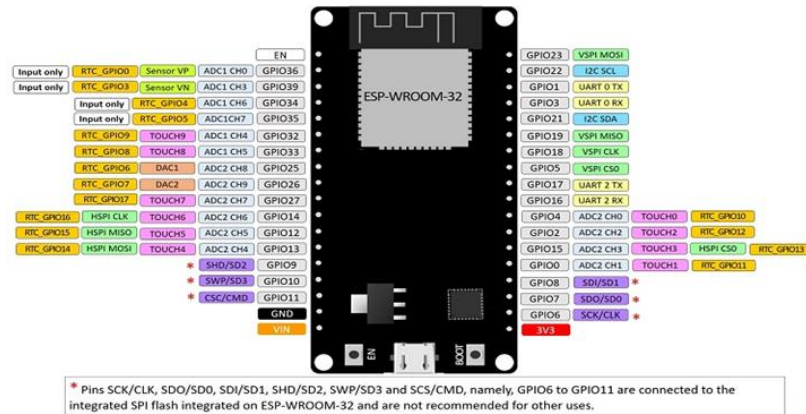


Figure 3.2: ESP-32 Pinout Description

3.1.2 RC-522 RFID Reader

3.1.2.1 Description

RC-522, as shown in Figure 3.3, is a Multi-communication RFID reader used for Arduino, ESP-32, and different microcontrollers. It is a High Frequency (HF) module operating at a frequency of 13.56 MHz [13]. It can be used for both reading and writing unique identification codes of RFID stickers, tags, and cards. It can read within a range of 8cm-10cm. The RFID cards are detected using mutual induction principles.



Figure 3.3: RC-522 Module

3.1.2.2 Features

The RC-522 has the following distinctive features:

- › It has an auto-sleep mode, thus making it power-efficient.

- › It requires 3.3V only to get activated.
- › The RFID cards, tags, or stickers are detectable from both sides in the range of 8-10 cm.
- › The module supports UART, SPI, and I2C communication making it compatible with almost any micro-controller or device.
- › It has a data rate of 10 Mb/s.

3.1.2.3 Specifications

Some of the specifications of RC-522 RFID Reader are listed in Table 3-2:

Table 3-2: RC-522 Specifications

Frequency Range	13.56 MHz
Host Interface	SPI / I2C / UART
Operating Supply Voltage	2.5 V to 3.3 V
Max. Operating Current	13-26mA
Min. Current (Power down)	10 μ A
Logic Inputs	5V Tolerant
Read Range	8-10 cm

3.1.2.4 Applications

The applications of RC-522 include but are not limited to:

- › Automated billing systems.
- › Attendance systems.
- › Verification and identification systems.
- › Access control systems.

3.1.2.5 Pin Configuration

The Pin Configuration of RC-522 is shown in Figure3.4:

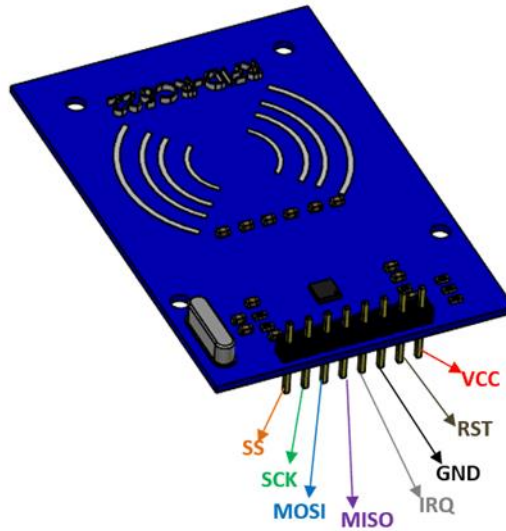


Figure 3.4: RC-522 Pin Configuration

The Table 3-3 gives a brief description of the pins of the RFID Reader.

Table 3-3: RC-522 Pin Descriptions

Pin Number	Pin Name	Description
1	VCC	Used to Power the module, typically 3.3V is used
2	RST	Reset pin – used to reset or power down the module
3	Ground	Connected to Ground of system
4	IRQ	Interrupt pin – used to wake up the module when a device comes into range
5	MISO/SCL/Tx	MISO pin when used for SPI communication, acts as SCL for I2c and Tx for UART.
6	MOSI	Master out slave in pin for SPI communication
7	SCK	Serial Clock pin – used to provide clock source
8	SS/SDA/Rx	Acts as Serial input (SS) for SPI communication, SDA for IIC and Rx during UART

3.1.3 Arduino Uno

3.1.3.1 Description

The Arduino UNO, as shown in Figure 3.5, uses an ATmega328p microcontroller which is present on board. It contains a set of input and output pins for both digital and analog communication. There are 6 analog pins amongst the 14 input pins [14]. It can be powered through a DC voltage of 6V-12V or USB connection. The board is cost-efficient, easily available in the market, and can be used for projects of any scale.



Figure 3.5: Arduino UNO Microcontroller

3.1.3.2 Features

Some of the salient features of Arduino UNO are:

- > For serial communication and to power up the board, it uses a USB connection.
- > It contains various pins such as timers, pulse width modulation, external and internal interrupts, and various types of sleep modes.
- > An inbuilt voltage regulation mechanism is present on board.
- > It acts as a serial device by using an ICSP connector.

3.1.3.3 Specifications

The specifications of Arduino UNO are listed in Table 3-4:

Table 3-4: RC-522 Pin Descriptions

Micro-controller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Digital I/O Pins	14
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB
SRAM	2 KB
EEPROM	1 KB
Clock Speed	16 MHz

3.1.3.4 Applications

The Arduino UNO is applicable in projects of any domain, some of them are:

- > Home automation
- > IoT
- > Public utility automation
- > Embedded systems
- > Robotics
- > Motion Control rig
- > Ardupilot(drones)
- > Defense and security
- > Industrial automation
- > 2D Games creation
- > Data logging

3.1.3.5 Pin Configuration

The pin configuration is given in Figure 3.6:

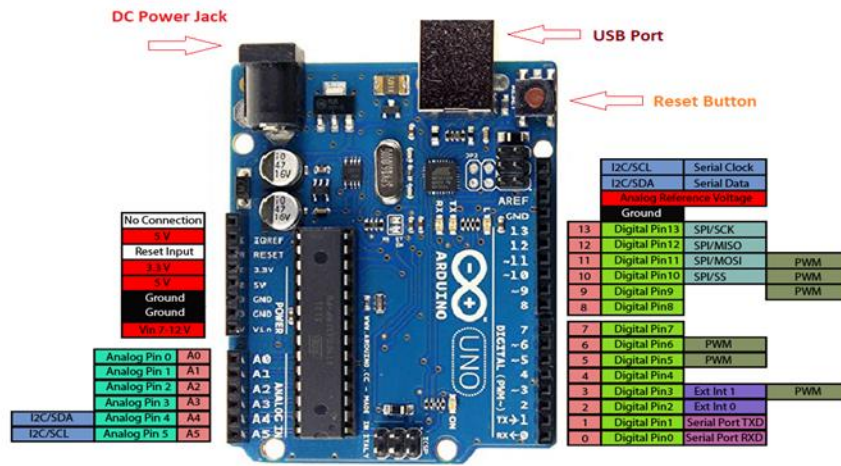


Figure 3.6: Arduino UNO Pin Configuration

3.1.4 DC Motor 12V

3.1.4.1 Description

The 12V DC motor, as shown in Figure 3.7, is a cost-efficient and small device but powerful enough to be used in various applications. The variable voltage supply changes the strength of input current in the field windings of the motor which is used to control the speed of the motor [15]. The motor produces high power and torque.



Figure 3.7: 12V DC Motor

3.1.4.2 Applications

The 12V dc motor is applicable in a vast range of equipment and projects scaling from industrial equipment to small school projects. Some of the applications of DC motor are:

- › Industrial machinery
- › Smart medical devices

- › Smart security systems
- › Public utility automation
- › Embedded systems
- › Robotics
- › Drones

3.1.5 L-298N Motor Driver

3.1.5.1 Description

The L-298N Motor driver, as shown in Figure 3.8, is used to control stepper and DC motors. It is a high current and voltage dual full-bridge motor driver and consists of a dual H-bridge circuit. It controls both the speed and direction of the rotation of motors. Pulse width modulation (PWM) is used to control the speed of the DC motor and the direction of rotation is managed by H-bridge.

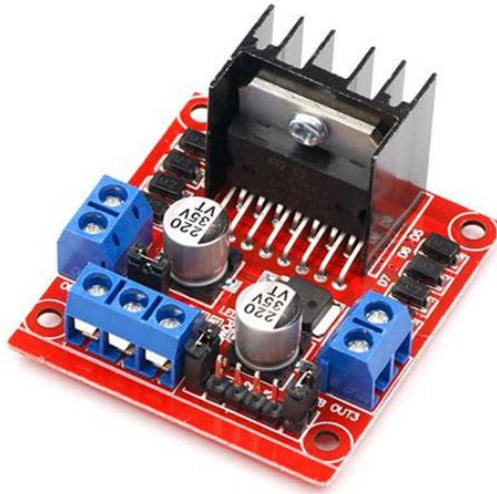


Figure 3.8: L-298N Motor Driver

3.1.5.2 Features

Some of the features of the L-298N driver are:

- › It accepts standard Transistor–transistor logic (TTL) levels.
- › It has two inbuilt H-bridges that are used to drive stepper and DC motors.

- › It can drive 2-phase DC motors and 2-phase or 4-phase stepper motors.

3.1.5.3 Specifications

The specifications of L-298N are listed in the table below:

Table 3-5: L-298N Specifications

Double H Bridge Drive Chip	L298N
Logical Voltage	5V
Drive Voltage	5V-35V
Logical Current	0-36mA
Drive current	2A
Max Power	25W

3.1.5.4 Applications

The L-298N is applicable in all projects making use of DC motors and stepper motors. Some of the applications of the L-298N motor are:

- › Industrial machinery
- › Smart medical devices
- › Smart security systems
- › Public utility automation
- › Embedded systems
- › Robotics
- › Drones

3.1.5.5 Pin Description

The pin configuration of the L-298N motor driver as shown in Figure 3.9:

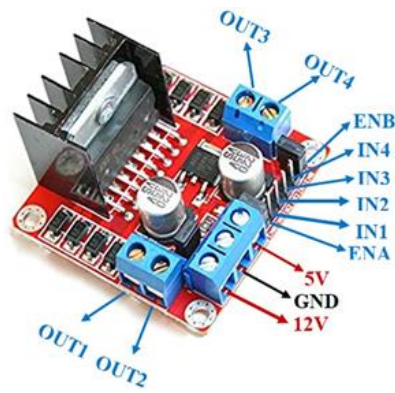


Figure 3.9: L-298N Driver Pinout

The pin description of the motor driver is listed in Table 3-6:

Table 3-6: L-298N Pin Description[16]

Pin Name	Description
IN1 & IN2	Motor A input pins. Used to control the spinning direction of Motor A
IN3 & IN4	Motor B input pins. Used to control the spinning direction of Motor B
ENA	Enables PWM signal for Motor A
ENB	Enables PWM signal for Motor B
OUT1 & OUT2	Output pins of Motor A
OUT3 & OUT4	Output pins of Motor B
12V	12V input from DC power Source
5V	Supplies power for the switching logic circuitry inside L298N IC
GND	Ground pin

3.2 Project Software

Software is the most important part of IoT and automation projects. The software that is being used in these projects was selected based on the architecture of the project that was designed after a lot of research on similar projects been done before. The software that is used in this project are:

- > Android Studio IDE
- > Visual Studio Code

- › Arduino IDE
- › Proteus
- › Thonny

3.2.1 Android Studio Ide

3.2.1.1 Description

Android Studio IDE is an Integrated Development Environment (IDE) based on IntelliJ IDEA. It has a powerful code editor and developer tools along with other distinctive features that enhance productivity making it better than many other IDEs. It is one of the most used IDE for android application development. In our project, we use emulator services of Android Studio IDE.



Figure 3.10: Android Studio IDE Logo

3.2.1.2 Features

The features of Android Studio IDE are distinctive and increase productivity.

- › It has a Gradle-based built system.
- › It has a very fast emulator that is rich in features.
- › It provides a unified environment for all Android devices.
- › It has built-in google platform support.

3.2.1.3 Version

The version used for ILM is 4.2.

3.2.2 Visual Studio Code

3.2.2.1 Description

Visual Studio (VS) Code is an open-source text editor made free developed by Microsoft. It supports multiple operating systems such as Windows, Linux, and macOS. The small size and memory use of VS Code has made it one of the most used editors today. It has multiple features that have made VS Code the most popular development environment tool in recent times. In our project, we used the Flutter software development kit (SDK) in VS Code and the programming language used for the app is Dart.



Figure 3.11: VS Code Logo

3.2.2.2 Features

Some of the features of VS Code are:

- › Support for multiple programming languages and SDKs.
- › It allows add-on and the creation of new extensions.
- › Simpler user experience.

3.2.2.3 Version

The version used for ILM is 1.51.

3.2.3 Arduino Ide

3.2.3.1 Description

Arduino. cc is the creator of Arduino IDE. It is used for writing and compiling the codes. The codes are then further uploaded to Arduino board and many other microcontrollers. It has a

simple and user-friendly Graphical User Interface (GUI). It supports all the operating systems such as Windows, Linux, etc.



Figure 3.12: Arduino IDE Logo

3.2.3.2 Features

Arduino IDE supports codes in languages that have compilers for Arduino instruction set and other languages like C and C++.

3.2.3.3 Version

The version used in our project is 1.8.49.0.

3.2.4 Proteus

3.2.4.1 Description

It is the most popular software tool primarily used for creating schematics and circuit simulation. It is used by electric circuit designers for electronic design automation. It can design PCB boards from a simple circuit schematic design.



Figure 3.13: Proteus Logo

3.2.4.2 Features

The features of proteus are:

- › Almost every component can be added in proteus in the form of libraries if not already available in proteus.
- › Easy and simple circuit schematics design.
- › It is also capable to simulate micro-controllers.
- › It is the most common software used for PCB designing.
- › The 3D imagery of the schematic design can also be generated.

3.2.4.3 Version

The version of Proteus used is 8.9.

3.2.5 Thonny

3.2.5.1 Description

Thonny is a free Python IDE having a built-in debugger and step-through expression evaluation. It has an easy-to-use debug mode that steps through the code line by line [17]. It also offers syntax highlighting, thus, making it easy to spot potential errors. It also highlights variable occurrences making it easy for users to avoid typos. It is used for python and micro-python.



Figure 3.14: Thonny Logo

3.2.5.2 Features

Thonny has many features, some of which are :

- › User-friendly interface.
- › Easy to use variables.
- › Simple debugger.

- › Scope explanation.
- › Explores Application Programming Interface(API) using the completion of codes.

3.2.5.3 Version

The version of Thonny used is 3.3.10.

3.3 Database and Cloud Services

3.3.1 Google Firebase

Firebase is a Backend-as-a-Service (BaaS) based on Google's infrastructure and is used as a backend for mobile and web applications [18]. It stores data in JSON-like documents and falls within the category of a NoSQL database. We have used Cloud Firestore service which acts as our database and provides cloud services as well.



Figure 3.15: Firebase Logo

3.3.1.1 Features

The features of google firebase are:

- › It provides authentication services.
- › It provides a real-time database.
- › It provides hosting services to applications deployed on it.
- › It has a machine learning kit in which there is an SDK for machine learning tasks.
- › Simple interface for its users.

CHAPTER 4: MOBILE APPLICATION AND THE DATABASE

4.1 Overview

The mobile application is developed using flutter and dart and contains two interfaces, one is the admin interface for librarians and the other is the user interface for students. The two interfaces are created within the same application using the super admin feature which allows the admin side to control the library catalogue and data of students along with the options of adding/removing books and students and generation of fine, whereas, a student can only view his respective interface, he does not have the rights of admin to change anything within his interface of the application. The data is stored in a firebase database developed using google firebase console services.

The core purpose of mobile application along with the firebase database is to create easy access to the data of the books of the library from any remote location so that library members can search for books through this application database accurately and swiftly because of its availability via the cloud. The application contains the information regarding the book name, the record of the books issued, to whom issued, and the date of return. It can send notifications to the members for the date of return and generate fines for the late returns. Firestore is used to store all the relevant data. The functioning of the Mobile application and Firestore are thoroughly described below.

4.2 Admin Side

The admin side has a special feature of super admin that allows some administrative options within the same mobile application for the librarian interface. The admin side has full authority to log in through special credentials only available for the librarians. Some administrative options are also available e.g., issue and return of books, add and delete books from the catalogue, add and remove users, and fine management. The admin side has the following pages:

4.2.1 Login Page

The mentioned portion below in Figure 4.1 contains an interface to login into the app via concerned credentials. You will enter the main dashboard upon successful login. Details of credentials for the admin side is as:

Username: admin@gmail.com. Password: admin123.

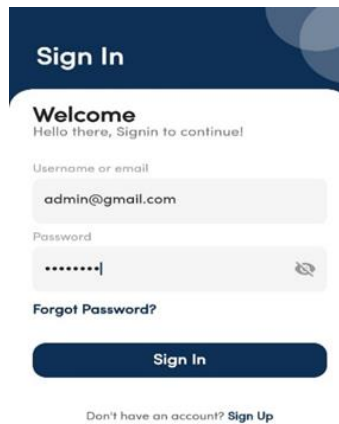


Figure 4.1: Sign-in Page

4.2.2 Dashboard

The concerned section serves as the main dashboard that contains all the relevant subpages that our application owns as shown in Figure 4.2.

- > Available books.
- > Issued books.
- > Registered users.
- > Fine generator.

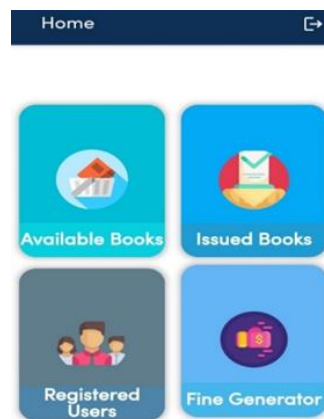


Figure 4.2: Dashboard

4.2.3 Available Books

Figure 4.3 is a manifesto of details of available books in the library book catalogue. All the available books will be shown with the titles and names of the respective authors.



Figure 4.3: Available Books Page

4.2.4 Add Books

The section shows the option that allows the admin to add any respective book along with the author's name into the library book catalogue. The description can be seen in Figure 4.4.

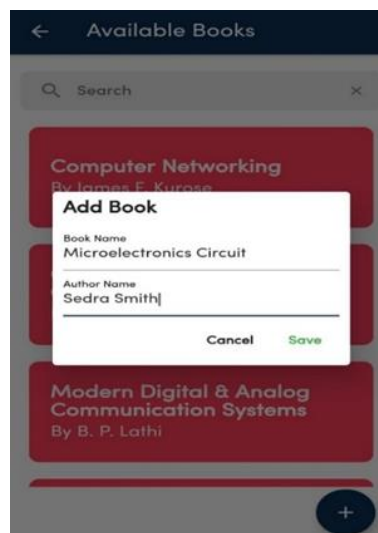


Figure 4.4: Add Books Page

4.2.5 Delete Books

The following page shows the option that allows the admin to delete any respective book from the library book catalog which is shown here in Figure 4.5.

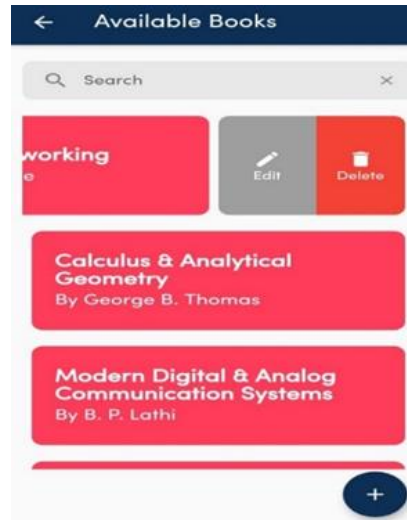


Figure 4.5: Delete Books Page

4.2.6 Issued Books

This issued book page, as shown in Figure 4.6 shows the books being issued by the librarian to students via their respective credentials, g-mail in our case. The admin has the option to issue any book to any of the students going over this page.

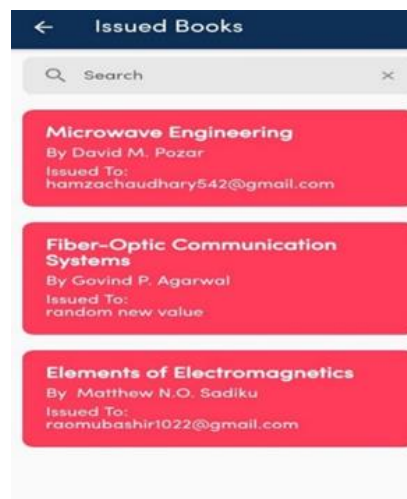


Figure 4.6: Issued Books Page

4.2.7 Issued Books - (Details)

Figure 4.7 shows the book is issued with the name of the author and issue details i.e student credentials, date of issue, and return.

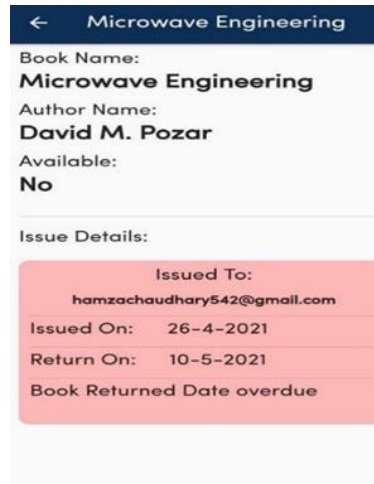


Figure 4.7: Details of Issued Books

4.2.8 Issued Books - (Returned)

The issued books page shows the option of books returned. If the book being issued is returned the librarian can click the green button so that the book can be available for other students now. The manifestation is shown in Figure 4.8.



Figure 4.8: Issued Books (Returned) Page

4.2.9 Registered Users

The registered users' page, as shown in Figure 4.9, contains the data of registered users into the mobile application i.e., username and respective e-mail. If any of the user credentials are not present in the database, it will not be considered as a registered user and will not be allowed to login into the application.

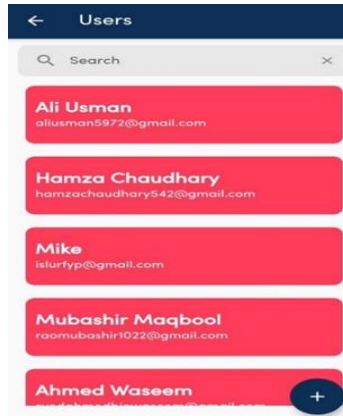


Figure 4.9: Registered Users Page

4.2.10 Overdue Books – (Generation of Fine)

The overdue books section shows the books that are not returned within the allotted time frame along with the fines generated. It contains book names, user credentials to which the book is issued, dates of issue and return, and the amount of fine generated. Following Figure 4.10 shows the overdue books.

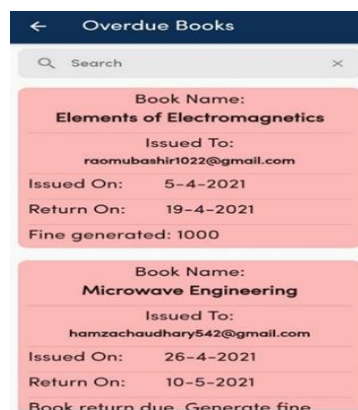


Figure 4.10: Overdue Books Page

4.2.11 Overdue Books – (Fine Paid)

This is the continuity of the last page where Figure 4.11 here shows the option of fine paid for the books that are not returned within the allotted time. If the fine is paid timely librarian can click the green button to clear the dues of the respective student.

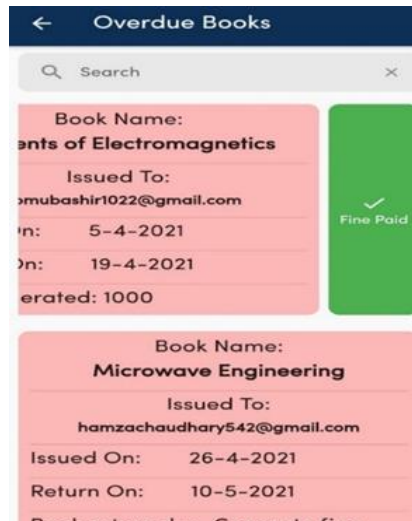


Figure 4.11: Overdue Books (Fine Paid) Page

4.2.12 Logout

The log out page shows the option to log out from the admin interface of the mobile application which is shown in Figure 4.12 below:

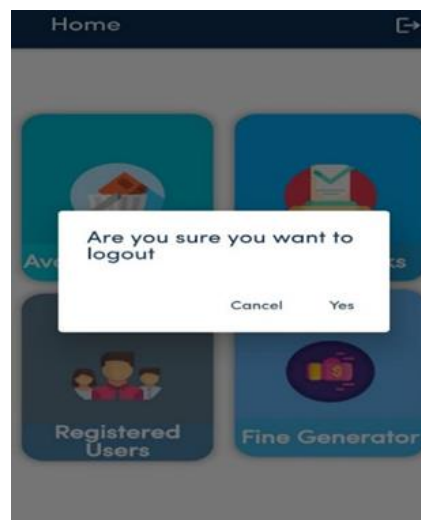


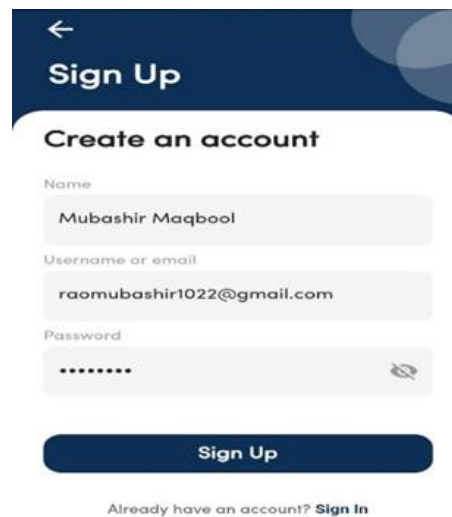
Figure 4.12: Logout Page (Admin Side)

4.3 User Side

The user side has a separate interface having limited options within the same mobile application. The user side can only log in through user credentials. Some options are also available like available books, issued books, notifications, and user profiles. The user can only view the data, he has no authority to alter anything in the data like librarians i.e., admin interface.

4.3.1 Signup

The Sign-up page here is to register a user for the mobile application. The students can sign up by giving a name to their accounts, respective e-mails, and unique passwords. Upon signup, their credentials will get registered into the database and they will be allowed to login into the user interface. Figure 4.13 is the pictorial description below.



←
Sign Up

Create an account

Name
Mubashir Maqbool

Username or email
raomubashir1022@gmail.com

Password
.....

Sign Up

Already have an account? Sign In

Figure 4.13: Signup Page

4.3.2 Sign in

This tab in the Figure 4.14 contains an interface to login into the app via concerned credentials. You will enter your username, email, and password on this page. You can sign in on the user side by making an account on the application.

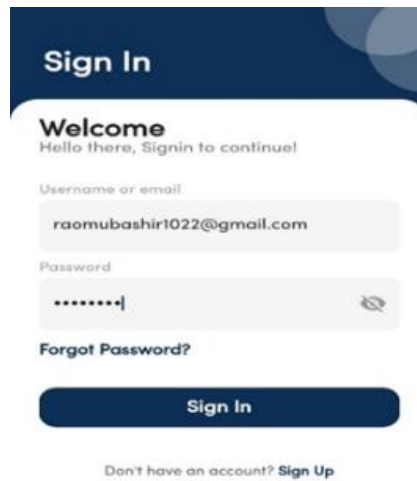


Figure 4.14: Sign-in Page

4.3.3 Available Books

The available books page in the Figure 4.15 shows details of available books in the library book catalog. All the available books will be shown with the titles and names of the respective authors. But at the user interface students are not allowed to alter any data i.e., add or remove any book, rather they can only view the available books.



Figure 4.15: Available Book (User Side)

4.3.4 Issued Books

This issued books page shows the books being issued to students via their respective credentials, g-mail in our case. The students can only view this catalog and have no authority to issue and return any book to others like the admin interface. Following Figure 4.16 is the required description.

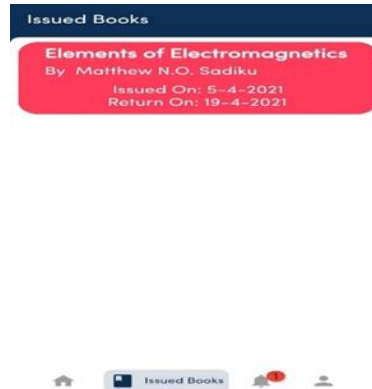


Figure 4.16: Issued Books (User Side)

4.3.5 Notification of Fine

This categorized tab shows the notification of fine to the respective student for not returning the concerned book within the allotted time frame. The pop-up notification will be: You have been fined for not returning “Book name” of “Fine Amount”. We can witness the description in Figure 4.17.



Figure 4.17: Notifications Page

4.3.6 User Profile - (Logout)

This last page of the user side in Figure 4.18 below shows the option to log out from the user interface of the mobile application.



Figure 4.18: Logout (User Side) Page

4.4 Firebase Database of Application

The mobile application's data gets stored into the database developed over the firebase console. The firebase console services are free and quite easy to use. The data gets updated into the console automatically upon processing the hardware side simultaneously. The firebase database contains the following two services:

4.4.1 User Authentication on Firebase

The user authentication window shown below in Figure 4.19 contains the credentials of users. It has the names of students, along with the date of creation of user accounts and the time slots at which the respective accounts login into the application. Each account created via g-mail is assigned a unique ID by the firebase console, only these registered users against specific Ids will be allowed to log in and all the irrelevant accounts will not be able to log in.

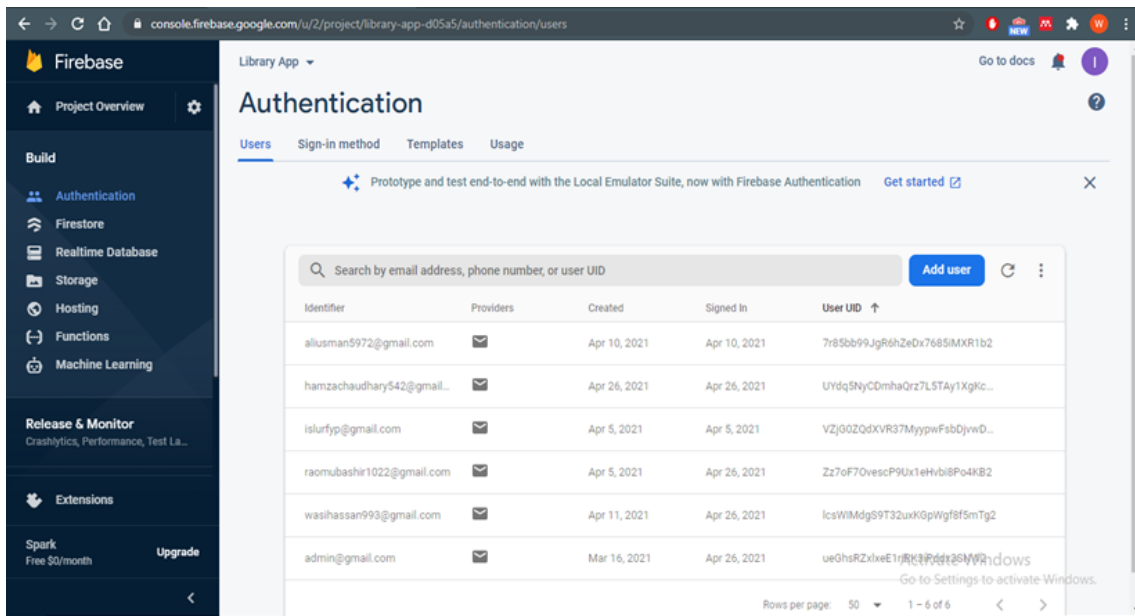


Figure 4.19: Firebase User Authentication

4.4.2 Cloud Firestore

The Cloud Firestore is the main database that contains data of books, respective users, and admin credentials. It is a cloud service that is linked to the application and enables the alteration of data upon processing. Following figure 4.20 manifests the thorough description.

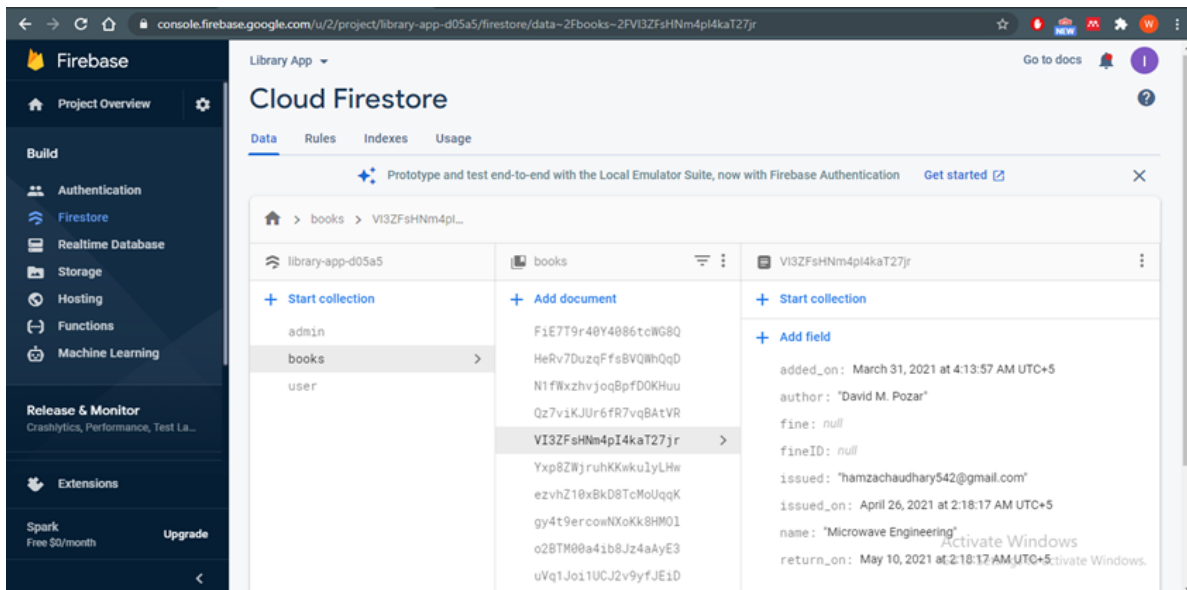


Figure 4.20: Cloud Firestore

CHAPTER 5: HARDWARE DESIGN

The hardware architecture of the project is constituting the most important aspect of any project. Any idea can only be made a successful and sustainable market product when its hardware is providing the best value to money to the customers. The project should not only be efficient but also be competitive in price as compared to any of its competitors. The hardware of the project was designed and revised whenever needed following the knowledge acquired from literature review and analysis of the projects overlapping to this domain. Our respected supervisors provided valuable guidance based on their experience and knowledge of the hardware architecture of the project.

The hardware implementation was done following the architecture design and was regularly checked for the minimal possibility of error. The hardware of the project is centralized on RFID, unlike previously implemented library management systems which used Barcodes and Near-field communication (NFC). The RFID technology is being used in automation and management projects throughout the globe as it provides features that are beyond the capabilities of previously implemented technologies.

5.1 Radio Frequency Identification (RFID)

The data encoded in RFID tags is captured by Radio Frequency Identification (RFID) using radio waves. It provides contactless scanning, unlike previous identification technologies. The RFID works by employing an RFID transponder (tag) and a reader [19]. The transponder has a microchip that stores the identification data apart from the microchip it also has an inbuilt antenna. The data is read by the reader via the antenna of the transponder. The range of the antenna defines the scanning range for the reader and the ability of the reader to read data even through certain non-metallic mediums. The RFID system may be any one of the following:

- › Passive system
- › Active system

5.1.1 Passive Systems

In passive systems, the RFID reader transmits the power for the circuit of the transponder making it respond to the reader. The transponders are dependent on the reader for power. The reader induces a current in the transponder's antenna via electromagnetic waves. The passive system has tags with smaller reading ranges. This system is most used, being most cost-effective for commercial use. We are using the passive system in our project.

5.1.2 Active Systems

The active systems constitute of battery-powered tags having boosted effective operating ranges as compared to the passive systems. The fact that transponders are battery-powered enables active systems to provide additional valuable features such as temperature sensing. The active RFID tags can independently transmit and receive data. The active systems are comparatively expensive to use.

The RFID technology uses several operating frequencies of radio waves as given in Table 5-1, below:

Table 5-1: Operating Frequencies of RFID [20]

Low Frequency (LF)	125 kHz
High Frequency (HF)	13.56 MHz
Very High Frequency (VHF)	433 MHz
Ultra-High Frequency (UHF)	860 – 960 MHz
Microwave Frequency	2.4 GHz

In the project, we have used High Frequency (HF) as it is the most cost-efficient and has optimum data transfer speed. The architecture of the hardware constitutes of two parts depending on their functionalities. The two parts are named as:

- > Book issue and return
- > Gate entry and exit

5.2 Book Issue and Return

5.2.1 Overview

The issue and return process of the book is the most important aspect of the hardware design and implementation. This part of the hardware constitutes of WROOM ESP-32 micro-controller, the RC-522 RFID reader, and the Google Firebase Firestore console services. The Firestore console services provide a real-time database for the project. The database will contain all the data related to the books and the students/members of the library. The Firestore console is further connected to the mobile application of the library management system over the Google cloud. The student willing to issue a particular book from the library will scan the book first using the RFID reader and then will scan his RFID library membership card instantly the information of the issue of the book to that student will be updated in the database and will also be seen in the mobile application.

Similarly, the process of return of the book will also require the student to scan the book first using the RFID reader and then his RFID library membership card indicating that this book has been returned by this issuer. The whole process of the issue and return of the book is automated as no library staff is required to issue and return the book to the student similarly the real-time data is available to keep a track of the issue and return time as well.

5.2.2 Block Diagram

The block diagram highlighting the architecture of the issue and return aspect of the hardware is as shown in Figure 5.1 :

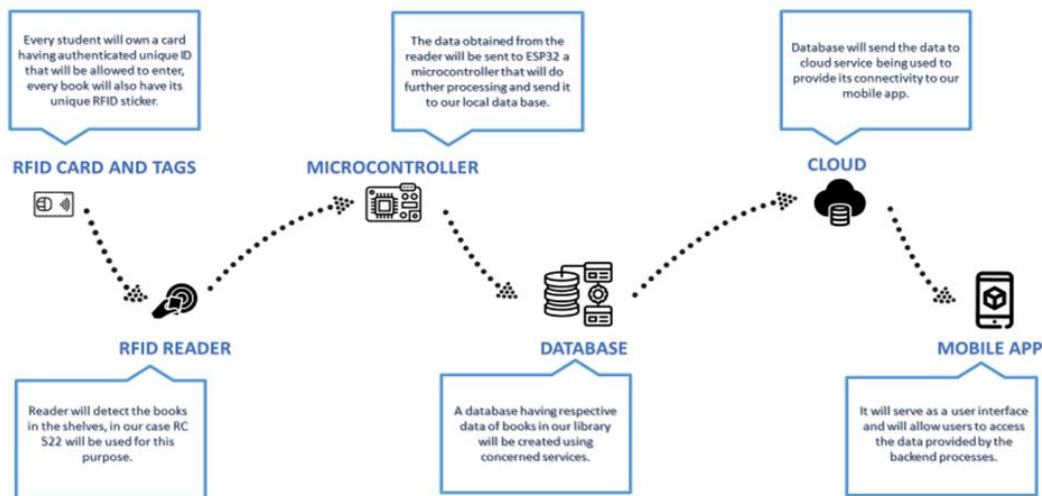


Figure 5.1: Block Diagram

It can be observed through the block diagram that this part has two interfaces of ESP-32 micro-controller:

- > ESP-32 and RC-522.
- > ESP-32 and Firestore.

5.2.3 ESP-32 AND RC-522

The ESP 32 is interfaced to RC-522 to read RFID cards and stickers. The data of the RC-522 is communicated to ESP-32 through serial communication. This Unique ID of the RFID card or sticker is then used by ESP-32 for further process.

5.2.3.1 Circuit Diagram

The connections of the respective pins of ESP-32 and RC-522 are shown in Figure 5.2:

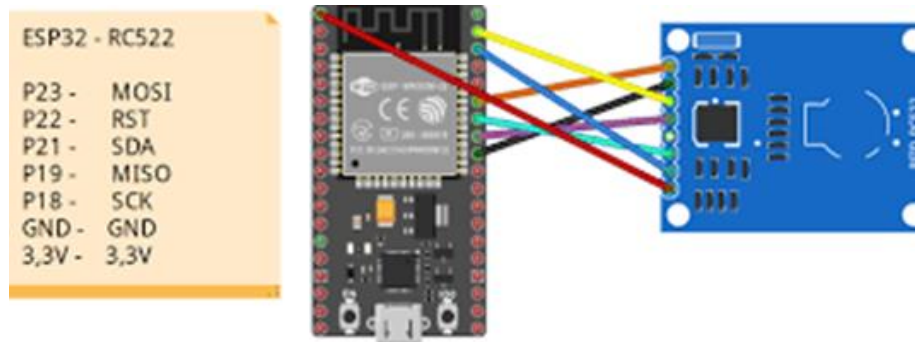


Figure 5.2: Circuit Diagram

5.2.3.2 Implementation

In the first step of the micro-python program the pin configuration was defined according to the circuit connections:

```
sck = Pin(18, Pin.OUT)
mosi = Pin(23, Pin.OUT)
miso = Pin(19, Pin.OUT)
spi = SPI(baudrate=100000, polarity=0, phase=0, sck=sck, mosi=mosi, miso=miso)
rst = Pin(13, Pin.OUT)
rst.value(1)
sda = Pin(5, Pin.OUT)
```

Figure 5.3: Pin configuration using micro-python.

The RC-522 will now read the RFID transponder and send the data through Serial Peripheral Interface (SPI) to the micro-controller. On the arrival of this data, the micro-controller runs the following commands to detect the type of the transponder scanned and generates the respective UID for further use.

```

def do_read():
    try:
        while True:
            rdr = MFRC522(spi, sda)
            uid = ""
            (stat, tag_type) = rdr.request(rdr.REQIDL)
            if stat == rdr.OK:
                (stat, raw_uid) = rdr.anticoll()

                if stat == rdr.OK:

                    rfid = ""
                    for i in raw_uid:
                        rfid +=str(i)
                    return (rfid)

                    sleep_ms(100)
            else:
                #print("No")
                pass
        except KeyboardInterrupt:
            print("Bye")
    if __name__ == "__main__":
        while True:

            rfid = "empty"
            while rfid == "empty":
                rfid = do_read()
            print(rfid, type(rfid))

```

Figure 5.4: UID and Type generation

5.2.4 ESP-32 and Firestore

The data acquired from the RC-522 is then processed by ESP-32. The ESP-32 uses the data acquired to detect the type of transponder scanned and classifies it as either a book sticker or student membership card depending on its type. After the classification of the data, the data is uploaded to the Firestore of the Google Firebase console. The ESP-32 connects with the Firestore through an internet connection made through Wi-Fi. The ESP-32 not only sends the data to the Firestore but also gets the data from the Firestore from the data. The ESP-32 can update, modify, and delete data from the firebase in an autonomous manner when required.

5.2.4.1 Implementation

The ESP-32 is connected to a local Wi-Fi connection using its SSID and password before proceeding further.

```

def connect():
    import network

    ssid = "Your Father"
    password = "Abc-2020***Black Rose+1947/Pakistan"

    station = network.WLAN(network.STA_IF)

    if station.isconnected() == True:
        print("Already connected")
        return

    station.active(True)
    station.connect(ssid, password)

    while station.isconnected() == False:
        pass

    print("Connection successful")
    print(station.ifconfig())
if __name__ == "__main__":
    connect()

```

Figure 5.5: Wi-Fi connection

After establishing the Wi-Fi connection, the Network Time Protocol (NTP) server connects to the ESP-32. The information of time for further processes is provided by Network Time Protocol server.

```

import ntptime

[year,month,date,hours,minutes,seconds,ms,us] = time.localtime()
print("Current time: %s:%s:%s, Date:%s/%s/%s" %(str(hours),str(minutes),str(seconds),str(date),str(month),str(year)))
try:
    ntptime.settime()
except:
    print("Internet Connectivity Problem")

print("After NTP setting:")
[year,month,date,hours,minutes,seconds,ms,us] = time.localtime()
print("Current time: %s:%s:%s, Date:%s/%s/%s" %(str(hours),str(minutes),str(seconds),str(date),str(month),str(year)))

```

Figure 5.6: NTP server connection

The ESP-32 is integrated with Firestore using the Representational state transfer (REST) application programming interface (API). The base URL of the API is used in this regard according to the documentation of the Firestore.

5.2.4.2 Book Issue

Once a book and student card are scanned ESP-32 makes a get request to Firestore to get information about the book. The data provided by the Firestore is then used by the ESP-32 micro-controller to evaluate whether this book is available or issued. If the book is issued then the details of the issuer, and the issuing time is also attained.

```
base_url = 'https://firestore.googleapis.com/v1/projects/library-app-d05a5/databases/(default)/documents/books/'
```

Figure 5.7: Base URL of Rest API

```
response = requests.get(request_url)
if response.status_code == 200:
    print("Request Successfull. Waiting for data")
    data = response.json()
    issued = data['fields']['issued']
    issued_value = list(issued.values())[0]
    issued_key = list(issued.keys())[0]

    issuedOn = data['fields']['issued_on']
    issuedOn_value = list(issuedOn.values())[0]
    issuedOn_key = list(issuedOn.keys())[0]
```

Figure 5.8: Get Request.

If the book is not issued previously, then the ESP-32 makes a patch request and updates the Firestore about the issuer and the time of issue.

```
if not(issued_value == ''):
    print("Book is issued to: ", issued_value)
    #Clearing the issued field
    patch_url = base_url + books[rfid_book]+"?currentDocument.exists=true&updateMask=fieldPaths=issued"
    print(patch_url)
    body = {
        "fields": {
            "issued": {
                issued_key : ""
            }
        }
    }
    data = json.dumps(body)
    #print(data)
    patch_request = requests.patch(patch_url, data=data).json()
```

Figure 5.9: Patch request for book issue

5.2.4.3 Book Return

When a book and student card are scanned for return the ESP-32 will make a get request like the request made while issuing to get the information about the book. If the book is issued earlier then the return information will be updated on the Firestore using patch request.

```
[year,month,date,hours,minutes,seconds,ms,us] = time.localtime()
dt_string = str(year)+"-"+str(month)+"-"+str(date)+"T"+str(hours)+":"+str(minutes)+":"+str(seconds)+"."+str(ms)+"Z"

patch_url = base_url + books[rfid_book]+"?currentDocument.exists=true&updateMask.fieldPaths=return_on"
print(patch_url)
body = {
    "fields" : {
        "return_on" : {
            "timestampValue" : dt_string
        }
    }
}
data = json.dumps(body)
#print(data)
patch_request = requests.patch(patch_url, data=data).json()
```

Figure 5.10: Patch request book return

The information is updated on the Firestore database about the book issuance and returns as given in the figure below:

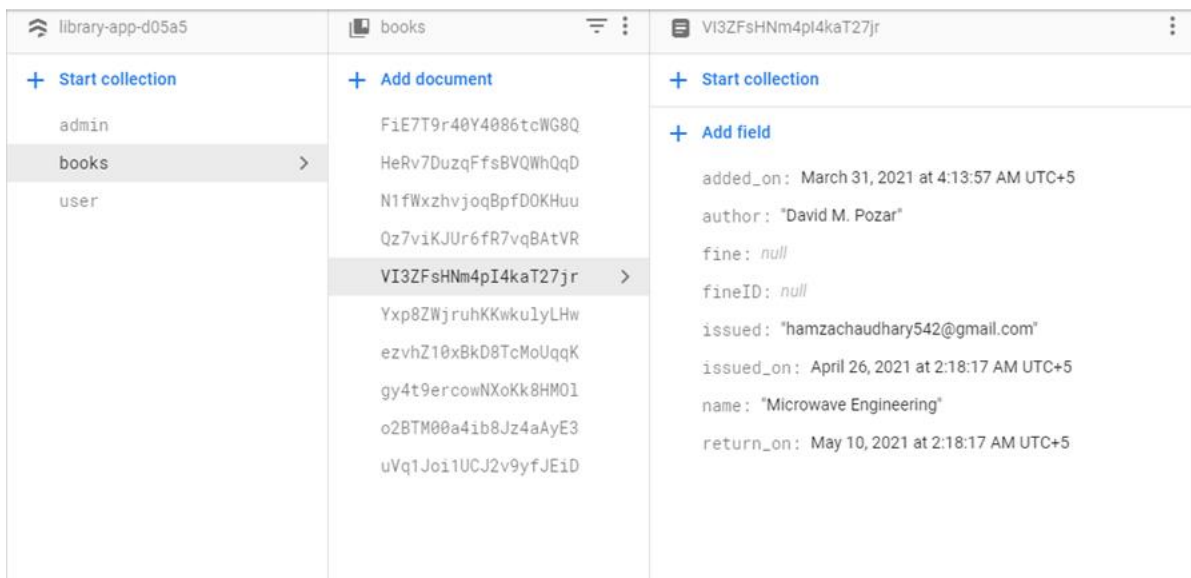


Figure 5.11: Firestore Database

In case, the student is unable to return the book in time a fine is generated, and the information of the fine is also updated on the Firestore.

```
added_on: April 5, 2021 at 5:24:31 PM UTC+5
author: " Matthew N.O. Sadiku"
fine: 1000
fineID: "I5fMEqwJ8Wo9uKTZhmZm"
issued: "raomubashir1022@gmail.com"
issued_on: April 5, 2021 at 11:49:41 PM UTC+5
name: "Elements of Electromagnetics"
return_on: April 19, 2021 at 11:49:41 PM UTC+5
```

Figure 5.12: Fine Information

5.3 Gate Entry and Exit

5.3.1 Overview

The gate entry and exit part of the hardware consists of RC-522, Arduino UNO, L-298N, and a DC motor. The RFID card of the student will be scanned using RC-522 if the student has a valid RFID card the gate of the library would slide open and if it is not a valid RFID card the student will not be able to enter the library. This process is automated as only valid library members can enter the library whereas the gate will not open for the invalid RFID cards, the library management would now not require checking each student manually as all the work here is done using the hardware designed for the entry and exit.

5.3.2 Block Diagram

The block diagram highlighting the architecture of the entry and exit aspect of the hardware is shown below:

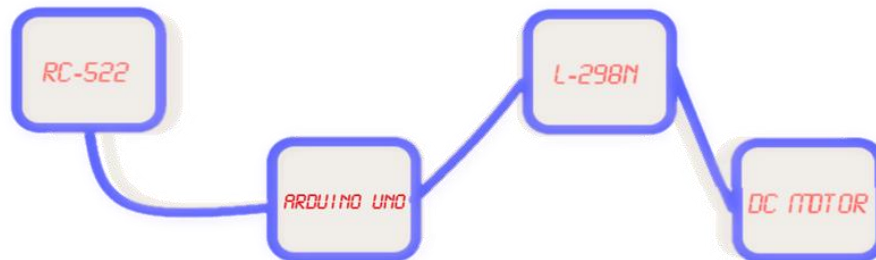


Figure 5.13: Block Diagram

In the light of the block diagram, it can be observed that this part has two interfaces of Arduino UNO micro-controller:

- › Arduino UNO and RC-522
- › Arduino UNO, L-298N and DC motor

5.3.3 Arduino Uno and Rc-522

The Arduino UNO is interfaced to RC-522 to read RFID cards. The data of the RC-522 is communicated to Arduino UNO through serial communication. This Unique ID of the RFID card is then used by Arduino UNO to control the gate movement.

5.3.3.1 Circuit Diagram

The connections of the respective pins of Arduino UNO and RC-522 are shown in Figure 5.14:

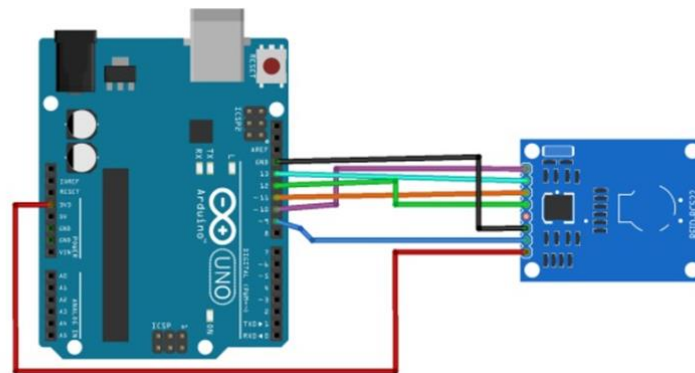


Figure 5.14: Arduino UNO and RC-522 circuit

The pin configuration of the above circuit is shown in Table 5-2:

Table 5-2: Pin Configuration

Pin	Wiring to Arduino Uno
SDA	Digital 10
SCK	Digital 13
MOSI	Digital 11
MISO	Digital 12
IRQ	unconnected
GND	GND
RST	Digital 9
3.3V	3.3V

5.3.3.2 Implementation

In the first step of the micro-python program the pin configuration was defined according to the circuit connections:

```
#define SS_PIN 10  
#define RST_PIN 9
```

Figure 5.15: Pin Configuration in Arduino IDE

The RC-522 will now read the RFID transponder and send the data through Serial Peripheral Interface (SPI) to the micro-controller. Upon the arrival of this data, the micro-controller runs the following commands to generate the respective UID for further use.

```

// Look for new cards
if ( ! mfrc522.PICC_IsNewCardPresent() )
{
    return;
}
// Select one of the cards
if ( ! mfrc522.PICC_ReadCardSerial() )
{
    return;
}
//Show UID on serial monitor
Serial.print("UID tag :");
String content= "";
byte letter;
for (byte i = 0; i < mfrc522.uid.size; i++)
{
    Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : "");
    Serial.print(mfrc522.uid.uidByte[i], DEC);
    content.concat(String(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : ""));
    content.concat(String(mfrc522.uid.uidByte[i], DEC));
}

```

Figure 5.16: UID Generation

5.3.4 Arduino Uno, L-298N and Dc Motor

Arduino UNO processes the data acquired from the RFID reader. The microcontroller compares the unique identification code, in a result of a comparison Arduino UNO sends a signal to L-298N which operates the DC motor, the direction of rotation of the motor is dependent upon the authenticity of the card scanned. If it is a valid card, the motor will open the gate.

5.3.4.1 Speed and Direction Control

The L-298N motor driver uses Pulse Width Modulation(PWM) technique to control the motor's speed and the rotation of the motor is controlled through the H-bridge.

> **Pulse Width Modulation (PWM)**

In PWM the width of the pulses is varied, where each pulse consists of the full voltage that the motor handles, in our case it is 12V. A narrow-width pulse will decrease the speed of the motor whereas speed is increased by increasing the width of the pulse.

This technique is displayed in the figure below:

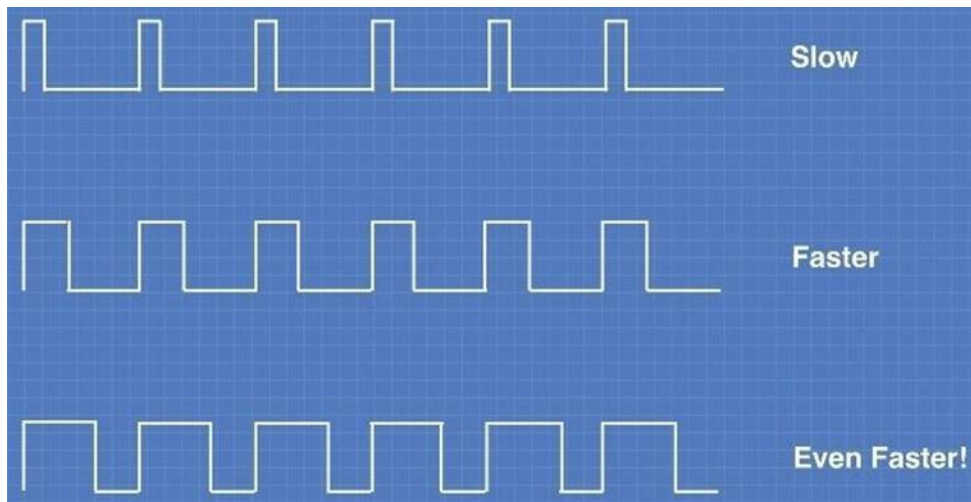


Figure 5.17: Pulse Width Modulation

> **H-Bridge**

The direction of rotation of the DC motor is done with the help of an H-bridge. It changes the polarity of input given to the motor to control its direction. When Q1 and Q4 are closed the motor rotates in a specific direction it moves in the opposite direction if Q2 and Q3 are closed.

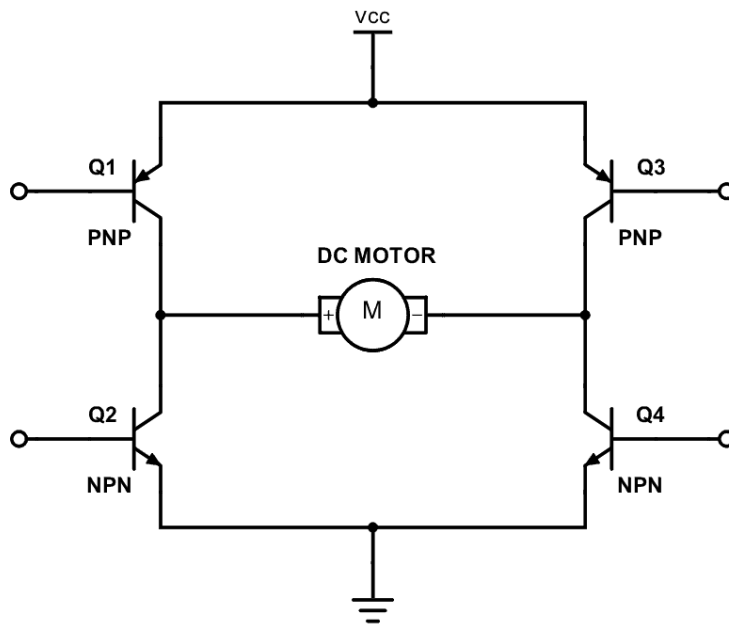


Figure 5.18: H-bridge

5.3.4.2 Circuit Diagram

The circuit of Arduino UNO, L-298N, and DC motor is given below:

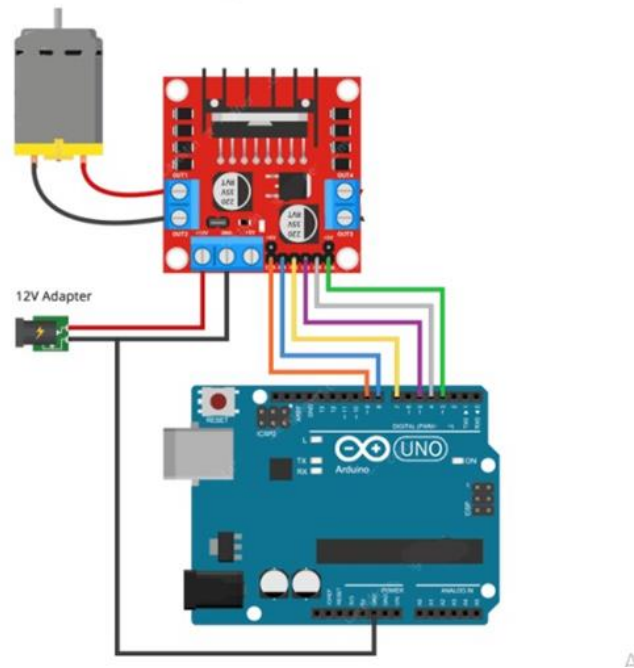


Figure 5.19: Arduino UNO, L-298N, and DC motor circuitry

5.3.4.3 Implementation

In the first step of the micro-python program the pin configuration was defined according to the circuit connections:

```
MFRC522 mfrc522(SS_PIN, RST_PIN); // Create MFRC522 instance.

int enA = 3; //Enables PWM for Motor A.
int in1 = 5; //Input for Motor A leadout 1.
int in2 = 4; //Input for Motor A leadout 2.
```

Figure 5.20: Pin Configuration in Arduino IDE

The Arduino UNO will check the validity of the scanned card and send a signal to L-298N to control the movement of the DC motor to open the gate if it is an authorized card.

```
{ Serial.println("Authorized access");  
  Serial.println();  
  delay(200);  
  analogWrite(enA, 255);  
  
  digitalWrite(in1, LOW);  
  digitalWrite(in2, HIGH);
```

Figure 5.21: Authorized Access

In case the scanned card is invalid Arduino UNO will send a signal to L-298N to control the movement of the DC motor to open the gate.

```
Serial.println(" Access denied");  
Serial.println();  
delay(200);  
analogWrite(enA, 1);  
digitalWrite(in1, HIGH);  
digitalWrite(in2, HIGH);  
delay(1000);
```

Figure 5.22: Unauthorized Access

CHAPTER 6: CONCLUSION

ILM has been successfully implemented using RFID and IoT technology. A user-friendly mobile application has been successfully developed for the system. For testing purposes, we have worked with a database of 10 books however any number of books can be entered into the database from the application. The availability of books can be checked from anywhere. An RFID sensed prototype gate is implemented whose operation is dependent on valid RFID cards. The issuance and return of the books have been successfully achieved. The record of issued and returned books is also kept. Thus, the system reduces the burdens of a librarian. His tasks of record-keeping are now all automated. Even the fines are automatically generated in case of a late return.

We can conclude that all the objectives of ILM as a prototype have been successfully achieved. The system has been thoroughly tested and its reliability is guaranteed.

6.1 Future Work

In the future, the basic design of the library management system can be improved by the addition of certain new features for this library. This may include the location of books in racks and adding certain other features to the mobile application. The system can also be implemented for warehouse and inventory management with certain minor modifications. So instead of books, many products in a warehouse can be managed. Another future is to launch ILM as a startup for digital Pakistan.

APPENDIX A

Main. Dart Code

```
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/material.dart';
import 'package:library_app/splash.dart';
import 'package:library_app/utils/colors.dart';

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  // Code Responsible to Run App.

  @override
  Widget build(BuildContext context) {
    Firebase.initializeApp().then((value) {});
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Flutter Demo',
      theme: ThemeData(
        primarySwatch: navyBlue,
        visualDensity: VisualDensity.adaptivePlatformDensity,
      ),
      home: SplashScreen(),
    );
  }
}
```

APPENDIX B

Main Micro-Python Script of ESP-32

```
# Connect to Wireless
import ConnectWifi
ConnectWifi.connect()

# Setting time and Date using NTP servie GMT+5
import time
import ntptime

[year,month,date,hours,minutes,seconds,ms,us] = time.localtime()

print("Current          time:           %s:%s:%s,          Date:%s/%s/%s"
%(str(hours),str(minutes),str(seconds),str(date),str(month),str(year)))

try:
    ntptime.settime()
except:
    print("Internet Connectivity Problem")

print("After NTP setting:")

[year,month,date,hours,minutes,seconds,ms,us] = time.localtime()

print("Current          time:           %s:%s:%s,          Date:%s/%s/%s"
%(str(hours),str(minutes),str(seconds),str(date),str(month),str(year)))

import read
import json
import urequests as requests

base_url      =      'https://firestore.googleapis.com/v1/projects/library-app-
d05a5/databases/(default)/documents/books/'

students = {
    "57215131131238" : "Hamza",
    "13121318564175" : "Wasi",
    "18572133131247" : "Ali",
    "731414613186"   : "Mubashir",
    "215731932570"   : "Ahmed",
    "1441213550248"  : "Naqash",
    "21747137131252" : "Talha",
    "163661746415"   : "Abdul Rehman"
```

```

}
books = {
    "136411326231" : "FiE7T9r40Y4086tcwG8Q",
    "13641372631" : "HeRv7DuzqFfsBVQWhQqD",
    "13641842646" : "N1fWxzhvjoqBpfdOKHuu",
    "136412126239" : "Qz7viKJUr6fR7vqBAtVR",
    "136410326241" : "VI3ZFsHNm4pI4kaT27jr",
    "13641562610" : "Yxp8ZWjruhKKwkulyLHw",
    "13641292623" : "ezvhZ10xBkD8TcMoUqqK",
    "13641722658" : "gy4t9ercowNXoKk8HMO1",
    "13641642650" : "o2BTM00a4ib8Jz4aAyE3",
    "1364146264" : "uVq1Joi1UCJ2v9yfJEiD"
}

if __name__ == "__main__":
    while True:
        rfid_book = "empty"
        rfid_student = "empty"
        print("Scan Book")
        while rfid_book == "empty":
            rfid_book = read.do_read()
            print(rfid_book, type(rfid_book))
            time.sleep_ms(3000)
            print("Scan Student ID")
            while rfid_student == "empty":
                rfid_student = read.do_read()
                print(rfid_student, type(rfid_student))
                time.sleep_ms(3000)
            request_url = base_url + books[rfid_book]
            print(request_url)
            response = requests.get(request_url)

```

```

if response.status_code == 200:
    print("Request Successfull. Waiting for data")
    data = response.json()
    issued = data['fields']['issued']
    issued_value = list(issued.values())[0]
    issued_key = list(issued.keys())[0]

    issuedOn = data['fields']['issued_on']
    issuedOn_value = list(issuedOn.values())[0]
    issuedOn_key = list(issuedOn.keys())[0]
    if not(issued_value == ''):
        print("Book is issued to: ", issued_value)
        #Clearing the issued field
        patch_url = base_url +
books[rfid_book]+"?currentDocument.exists=true&updateMask.fieldPaths=issued"
        print(patch_url)
        body = {
            "fields" : {
                "issued" : {
                    issued_key : ""
                }
            }
        }
        data = json.dumps(body)
        #print(data)
        patch_request = requests.patch(patch_url, data=data).json()
        #Setting the returned on field
        [year,month,date,hours,minutes,seconds,ms,us] = time.localtime()
        dt_string = str(year)+"-"+str(month)+"-
"+str(date)+"T"+str(hours)+":"+str(minutes)+":"+str(seconds)+"."+str(ms)+"Z"
        patch_url = base_url +
books[rfid_book]+"?currentDocument.exists=true&updateMask.fieldPaths=return_on"

```

```

print(patch_url)
body = {
    "fields" : {
        "return_on" : {
            "timestampValue" : dt_string
        }
    }
}
data = json.dumps(body)
#print(data)
patch_request = requests.patch(patch_url, data=data).json()
else:
    print("Book is not issued yet")
    # Updating the issued field
    patch_url = base_url +
books[rfid_book]+"?currentDocument.exists=true&updateMask.fieldPaths=issued"
    print(patch_url)
    body = {
        "fields" : {
            "issued" : {
                issued_key : students[rfid_student]
            }
        }
    }
}
data = json.dumps(body)
print(data)
patch_request = requests.patch(patch_url, data=data).json()
# Updating the issued_on field
[year,month,date,hours,minutes,seconds,ms,us] = time.localtime()
dt_string = str(year)+"-"+str(month)+"-
"+str(date)+"T"+str(hours)+":"+str(minutes)+":"+str(seconds)+"."+str(ms)+"Z"

```

```

        #time.sleep(2)
        patch_url = base_url +
books[rfid_book]+"?currentDocument.exists=true&updateMask.fieldPaths=issued_on"
        print(patch_url)
        body = {
            "fields" : {
                "issued_on" : {
                    "timestampValue" : dt_string
                }
            }
        }
        data = json.dumps(body)
        print(data)
        patch_request = requests.patch(patch_url, data=data).json()
    else:
        print("Request timeout or failed")

```

REFERENCES

- [1] J. W. Choi, D. I. Oh, and I. Y. Song, "R-LIM: An affordable library search system based on RFID," in *Proceedings - 2006 International Conference on Hybrid Information Technology, ICHIT 2006*, 2006, vol. 1, pp. 103–108, doi: 10.1109/ICHIT.2006.253472.
- [2] M. I. Younis, "SLMS: A smart library management system based on an RFID technology Design and Implementation of a Contactless Smart House Network System View project Current Projects View project SLMS: a smart library management system based on an RFID technology," *Int. J. Reason. Intell. Syst.*, vol. 4, no. 4, pp. 4–5, 2012, doi: 10.1504/IJRIS.2012.051717.
- [3] J. Pandey, S. I. A. Kazmi, M. S. Hayat, and I. Ahmed, "A study on implementation of smart library systems using IoT," in *2017 International Conference on Infocom Technologies and Unmanned Systems: Trends and Future Directions, ICTUS 2017*, 2018, vol. 2018-January, pp. 193–197, doi: 10.1109/ICTUS.2017.8286003.
- [4] E. Liaquat, A. Rahoo, S. I. T. Assistant, and A. Mukhtiar, "Design and Development of an Automated Library Management System for Mehran University Library, Jamshoro."
- [5] A. P. Renold and R. J. Rani, "An internet based RFID library management system," in *2013 IEEE Conference on Information and Communication Technologies, ICT 2013*, 2013, pp. 932–936, doi: 10.1109/CICT.2013.6558229.
- [6] M. Mohammadi and M. Yegane, "IOT: Applied New Technology in Academic Libraries."
- [7] X. Liang and Y. Chen, "Libraries in Internet of Things (IoT) era," *Libr. Hi Tech*, vol. 38, no. 1, pp. 79–93, Apr. 2020, doi: 10.1108/LHT-11-2017-0233.
- [8] D. Y. Li, S. D. Xie, R. J. Chen, and H. Z. Tan, "Design of Internet of Things System for Library Materials Management using UHF RFID," in *2016 IEEE International Conference on RFID Technology and Applications, RFID-TA 2016*, 2016, pp. 44–48, doi: 10.1109/RFID-TA.2016.7750755.
- [9] V. S. Prabhu, R. M. Abinaya, G. Archana, and R. Aishwarya, "IoT-based automatic library management robot," in *Advances in Intelligent Systems and Computing*, 2021, vol. 1163, pp. 567–575, doi: 10.1007/978-981-15-5029-4_46.
- [10] Y.-N. Sun and H.-S. Gou, *A New IoT Method for Economical Library Management*, vol. 0, no. wcne. 2016.
- [11] "ESP32 Wi-Fi & Bluetooth MCU | Espressif Systems." [Online]. Available: <https://www.espressif.com/en/products/socs/esp32>. [Accessed: 14-Jun-2021].
- [12] "Smart monitor of pacemaker patient by using IoT cloud... - Google Scholar." [Online]. Available: https://scholar.google.com/scholar?hl=en&as_sdt=0%2C5&q=Smart+monitor+of+pacemaker+patient+by+using+IoT+cloud+in+real+time&btnG=. [Accessed: 14-Jun-2021].
- [13] "RC522 RFID Module Pinout, Features, Specs & How to Use It." [Online]. Available: <https://components101.com/wireless/rc522-rfid-module>. [Accessed: 15-Jun-2021].
- [14] "Introduction to Arduino Uno - The Engineering Projects." [Online]. Available: <https://www.theengineeringprojects.com/2018/06/introduction-to-arduino-uno.html>.

[Accessed: 15-Jun-2021].

- [15] "How motors work and how to choose the right motor for any project." [Online]. Available: <https://www.jameco.com/Jameco/workshop/ProductNews/motor-buyers-guide.html>. [Accessed: 15-Jun-2021].
- [16] "L298N Motor Driver Module Pinout, Datasheet, Features & Specs." [Online]. Available: <https://components101.com/modules/l293n-motor-driver-module>. [Accessed: 15-Jun-2021].
- [17] "Thonny: The Beginner-Friendly Python Editor (Overview) – Real Python." [Online]. Available: <https://realpython.com/lessons/thonny-overview/>. [Accessed: 15-Jun-2021].
- [18] "What is Firebase?. Firebase is a Backend-as-a-Service —... | by Chris Esplin | How To Firebase." [Online]. Available: <https://howtofirebase.com/what-is-firebase-fcb8614ba442>. [Accessed: 15-Jun-2021].
- [19] "What is RFID and how does it work?" [Online]. Available: <https://internetofthingsagenda.techtarget.com/definition/RFID-radio-frequency-identification>. [Accessed: 15-Jun-2021].
- [20] "(14) (PDF) THE RFID TECHNOLOGY AND ITS APPLICATIONS: A REVIEW." [Online]. Available: https://www.researchgate.net/publication/232575248_THE_RFID_TECHNOLOGY_AND_ITS_APPLICATIONS_A_REVIEW. [Accessed: 15-Jun-2021].