OTP-Based Smart Locker System Using Embedded System Design Principles

Project Report – Minor Project

Submitted by

Harikishantini K	**Ishani Dey**	**Naveed Ismail**
220906286	220906486	220906644

Under the guidance of
Vijay S R
Assistant Professor – Senior
Department of Electronics & Communication Engineering,
Manipal Institute of Technology (MIT), Manipal.

in partial fulfilment of the requirements for the award of the degree of

MINOR SPECIALIZATION IN
EMBEDDED SYSTEMS

MANIPAL INSTITUTE OF TECHNOLOGY
(A Constituent Unit of Manipal Academy of Higher Education)
MANIPAL – 576104, KARNATAKA, INDIA
MONTH 2025

ACKNOWLEDGMENT

This project, “OTP-Based Smart Locker System Using Embedded System Design Principles,” was undertaken as part of the Minor Specialization in Embedded System Design. We, the project team, express our sincere gratitude to the faculty of Real-Time Systems, Embedded Systems, Internet of Things, and FPGA Design for their rigorous teaching, continuous guidance, and valuable feedback, which collectively shaped the foundation of our work.

We extend our heartfelt thanks to the laboratory staff for providing access to essential development boards, instruments, components, and simulation platforms such as Tinkercad, and for their timely assistance during various stages of prototyping, testing, and troubleshooting.

We also acknowledge the contributions of our peers and mentors, whose discussions, technical insights, and support played an important role in resolving challenges and enriching the quality of the project.

Lastly, we express our deep appreciation to our families and friends for their constant encouragement, motivation, and patience throughout the duration of this project. Their support has been invaluable in helping us complete this work successfully.

ABSTRACT

This report presents the design and implementation of an OTP-Based Smart Locker integrating Arduino Uno, 4×3 keypad, I²C 16×2 LCD, a 5 V relay module, and a 12 V solenoid lock, with Gmail/SMTP-based OTP delivery for dynamic, single-use authentication. Two complementary prototypes were developed: a Tinkercad simulation model (logic validation, DC motor actuation) and a real hardware prototype (solenoid actuation, email OTP). The system applies concepts from Real-Time Systems (deterministic keypad scanning, timing and lockout), Embedded Systems (peripheral interfacing, relay driving, debouncing), IoT (SMTP-based communication), and FPGA-style FSM design (state-based control flow).

Experimental results show fast OTP delivery (≈1–3 s), reliable user interaction on the LCD and keypad, stable relay switching, and strong, repeatable solenoid operation. Security testing confirms resistance to brute-force and replay attempts through limited retries, automatic lockout, and OTP invalidation after use. The architecture is modular and low-cost, making it suitable for deployment in homes, hostels, labs, and small offices. The report also discusses an extended multi-factor model that integrates fingerprint and ESP32-CAM face recognition for enhanced security. Overall, the project demonstrates a practical, scalable approach to physical access control that merges embedded design with modern IoT authentication.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

API	Application programming interface
DC	Direct Current
EEPROM	Electrically Erasable Programmable Read-Only Memory
ESP	Espressif Microcontroller Family
FPGA	Field Programmable Gate Array
FSM	Finite State Machine
GPIO	General Purpose Input/Output
GSM	Global System for Mobile Communications
IoT	Internet of Things
I²C / I2C	Inter-Integrated Circuit
LCD	Liquid Crystal Display
LED	Light Emitting Diode
MFA	Multi-Factor Authentication
NPN	Negative-Positive-Negative (transistor type)
OTP	One-Time Password
PCB	Printed Circuit Board
RFID	Radio-Frequency Identification
RTC	Real-Time Clock
SMS	Short Message Service
SMTP	Simple Mail Transfer Protocol
Wi-Fi	Wireless Fidelity

CHAPTER 1: INTRODUCTION

1.1. Introduction

Embedded systems have rapidly evolved to become fundamental enabling technologies in modern automation, security, and digital infrastructure. These systems integrate hardware and software into a tightly coupled architecture designed to perform dedicated tasks with precision, low power consumption, and high reliability. Today, embedded systems operate behind the scenes of critical domains such as transportation, healthcare, home automation, industrial control, IoT ecosystems, and smart security solutions.

In the domain of physical security, conventional mechanical lockers have been increasingly replaced by electromechanical and intelligent locking mechanisms to counter rising threats such as key duplication, lock picking, and unauthorized access attempts. While password-based digital locks marked a step forward, static codes remain vulnerable to guessing, shoulder surfing, and brute-force strategies.

To overcome these limitations, modern authentication paradigms emphasize the importance of dynamic and real-time authentication, including One-Time Passwords (OTPs), which offer enhanced protection due to their temporary nature. OTP systems are widely used in secure digital services such as banking, online transactions, and access control. Integrating OTP verification with embedded hardware offers a powerful approach to building secure, low-cost, and user-friendly physical security systems.

This project, **OTP-Based Smart Locker**, was developed under the Minor Specialization in Embedded System Design, synthesizing concepts from Real-Time Systems, Embedded Systems, Internet of Things (IoT), and FPGA-based digital design. The project explores the use of an Arduino Uno as the central control unit, a keypad for user input, an I2C LCD for real-time display of system states, a relay-actuated solenoid lock for mechanical security, and an IoT-enabled OTP delivery mechanism (via Gmail). Both virtual (Tinkercad) and real hardware implementations were developed, ensuring robust validation of the system's performance.

1.2. Present Day Scenario

Security in residential, commercial, and institutional environments has become increasingly dependent on digital automation. With the proliferation of smart homes, offices, and IoT-based monitoring systems, there is a growing shift from traditional mechanical locks to intelligent, multi-factor authentication systems.

Despite this growth, many commonly available digital lockers still rely on static PINs or simple magnetic key mechanisms. These mechanisms present several vulnerabilities:

- Static passwords can be observed and reused.
- Mechanical keys can be duplicated.
- Magnetic or RFID tags can be cloned.
- Biometric-only systems remain expensive and sometimes unreliable.

Meanwhile, OTP-based authentication has emerged as a gold standard for secure digital access. Every OTP is temporary, unpredictable, and valid for a single session, making it resistant to replay attacks and brute-force strategies. Although OTPs are widely deployed in online systems, their integration into physical lockers remains limited, especially in low-cost security devices.

In addition, easily accessible microcontroller platforms like Arduino, along with improvements in relay-based actuation and IoT services (Gmail, cloud APIs, SMTP servers), have made it feasible to design multifunctional yet economical smart lockers.

The current technological landscape shows a clear need for affordable, robust, and IoT-enabled OTP-based physical access systems—precisely the gap this project aims to address.

1.3. Motivation

The development of an OTP-Based Smart Locker was motivated by both societal needs and academic goals.

Key Motivational Factors

1. **Growing demand for secure and dynamic authentication**

   Static PINs are increasingly insufficient to protect valuables. OTPs introduce dynamic, per-use authentication.

2. **Rising incidents of theft and unauthorized access**

   Traditional lockers can be forcefully opened or bypassed using trial-and-error techniques.

3. **Opportunity to integrate multiple embedded system concepts**

   The project draws upon four key courses:

   - *Real-Time Systems:* deterministic input handling, timing constraints, OTP timeouts
   - *Embedded Systems:* microcontroller programming, relay interfacing
   - *IoT:* email-based OTP transmission
   - *FPGA:* state-machine logic influencing software design

4. **Hands-on hardware prototyping experience**

   The project provides experience in:

   - Driving relays and solenoids
   - Using keypads and LCDs
   - Managing power supplies
   - Building both simulation and real models

5. **Practical applicability**

   The system is suitable for personal lockers, dormitory storage units, small-business safes, and laboratory compartments.

Thus, the project is motivated by both real-world necessity and academic enrichment, showcasing a practical embodiment of embedded intelligence in physical security.

1.4. Relevance of the Work

This work is relevant from technical, societal, and academic viewpoints.

Technical Relevance

- Demonstrates embedded system design involving sensor interfacing, digital I/O, EEPROM management, and relay actuation.
- Showcases real-time input processing via keypad scanning.
- Implements dynamic OTP verification, which is not commonly seen in low-cost microcontroller-based lockers.

IoT Relevance

- Uses cloud platforms (Gmail SMTP/SMTP API) for secure OTP delivery.
- Integrates physical systems with digital communication services.

Academic Relevance

- Bridges concepts learned across four embedded specialization courses.
- Serves as a complete capstone demonstrating hardware-software co-design.

Real-World Application

- Can be installed in homes, offices, hostels, and labs.
- Offers higher security than traditional PIN-based locks.
- Ensures ease of use even for non-technical users.

Economic Relevance

- Built using low-cost hardware such as Arduino Uno and relay modules.
- Offers an affordable alternative to biometric and commercial IoT locks.

The system addresses an existing market gap by providing a scalable, secure, and cost-effective embedded smart locker solution.

1.5. Impact of the Work on the Environment and Other Factors

The environmental and societal impacts of the project were carefully considered during its design and implementation.

Environmental Impact

1. **Low Power Consumption**
    - Arduino Uno consumes minimal energy.
    - The solenoid lock is activated only momentarily.
    - Standby power is negligible.
2. **Reusable Components**
    - Microcontrollers, keypads, LCDs, and relays are fully reusable.
    - Reduces long-term e-waste.

3. **Safe Electronic Disposal**
    - The design encourages modular assembly, making components easy to replace or recycle.

Social Impact

- Enhances personal and institutional security.
- Reduces dependency on keys and mechanical systems.
- Protects user valuables, increasing user confidence.

Health & Safety Impact

- Solenoid and relay isolation prevent electrical hazards.
- OTP verification prevents user stress related to forgetting passwords.

Economic Impact

- Offers a low-cost alternative to high-end IoT biometric locks.
- Affordable for households and small businesses.

Overall, the system provides a secure solution with minimal environmental footprint and significant social benefit.

1.6. Ethics in Engineering Practices

Ethical engineering principles guided the design of this project.

Data Privacy

- OTPs are generated dynamically and not stored long-term.
- Only authorized users with registered email IDs receive OTPs.

System Security & Integrity

- Relay isolation prevents electrical accidents.
- Limit on incorrect OTP attempts prevents brute-force attacks.

Accuracy and Honesty in Implementation

- All components were interfaced following proper electrical standards.
- The design avoids unsafe shortcuts and follows good engineering practices.

User Transparency

- LCD prompts clearly inform the user of system operations.
- Error messages and warnings ensure safe usage.

Academic Ethics

- Concepts were implemented based on course teachings.
- No plagiarized hardware/software designs were used.

Ethical design ensures that the OTP-Based Smart Locker is safe, reliable, user-friendly, and responsible in safeguarding data and user well-being.

CHAPTER 2: LITERATURE REVIEW

2.1. Present State and Recent Developments

Smart security systems have undergone significant transformation over the past decade due to advancements in embedded technology, wireless communication, and cloud-based authentication services. Traditional locking mechanisms, which rely on metal keys or fixed PIN codes, have increasingly been replaced by electronic and IoT-enabled systems capable of providing stronger authentication, remote monitoring, and automated logging.

In the current landscape, several technologies dominate the access-control domain:

1. IoT-Enabled Smart Locks

Modern smart locks use Wi-Fi, Bluetooth, or GSM to allow remote access through mobile applications. Features such as remote door unlocking, activity logs, tamper alerts, and integration with home automation systems are becoming mainstream.

2. Multi-Factor Authentication (MFA)

Commercial security products are increasingly adopting multi-factor authentication:

- Something the user **knows** (PIN/password)
- Something the user **has** (phone/email access for OTP)
- Something the user **is** (biometrics)

3. OTP-Based Systems

OTPs generated on servers or through cryptographic algorithms are widely used in digital security, financial transactions, and identity verification. Their adoption in physical security is still emerging but expected to grow due to their high reliability and resistance to replay attacks.

4. Affordable Microcontrollers

Microcontrollers such as Arduino, ESP8266, ESP32, and Raspberry Pi Pico have made smart-lock prototypes more feasible in academic and small-scale environments.

5. Improvements in Electromechanical Systems

Modern solenoid locks, magnetic locks, servo locks, and linear actuators are more energy-efficient and responsive, making them suitable for embedded-controlled access.

6. Cloud and Email API Integrations

Cloud-based OTP services, SMTP gateways, and mobile push-notification frameworks have enabled secure and authenticated communication channels between embedded devices and users.

Despite these advancements, the market still lacks **cost-effective, standalone OTP-based physical locker systems**, particularly those suitable for low-power, microcontroller-driven applications. This project aims to address that gap.

2.2. Brief Background Theory

To understand the technological foundations of the OTP-Based Smart Locker, several underlying theories and principles must be examined:

1. Embedded Systems Architecture

An embedded system integrates hardware components (microcontroller, input/output devices) with dedicated software routines to perform specific tasks. Key components include:

- CPU (ATmega328P on Arduino Uno)
- Flash memory (to store program instructions)
- SRAM (for runtime variables)
- I/O interfaces (GPIO, ADC, communication pins)

2. Keypad Scanning Algorithm

Matrix keypads are arranged in rows and columns. The microcontroller:

- Sequentially energizes one column at a time
- Reads the row lines to detect which key is pressed
- Implements software debouncing to avoid false triggers

This process must be done in real time to maintain input accuracy.

3. I2C Communication Protocol

I2C is a two-wire protocol:

- **SDA** (Serial Data)
- **SCL** (Serial Clock)

Using I2C allows multiple slaves like LCDs and sensors to share the same bus, simplifying wiring and improving system scalability.

4. Relay and Solenoid Operation

A relay is an electrically controlled switch. It isolates the low-voltage control circuit (Arduino) from the high-voltage solenoid circuit.

A solenoid lock uses electromagnetic induction to retract or extend a mechanical plunger when current passes through its coil.

5. OTP Algorithms

An OTP typically involves:

- Random number generation
- Time-based expiration
- Single-use validation

In this project, OTPs are generated on the Arduino or supporting IoT system and sent to the user via email.

6. SMTP and IoT Email Transmission

Simple Mail Transfer Protocol (SMTP) allows devices to send emails through authenticated servers. This requires:

- IoT module (ESP8266/ESP32 or SMTP library via PC)
- Server authentication with encrypted credentials
- Internet connectivity for sending the OTP email

7. Real-Time Systems Concepts

Real-time constraints apply to:

- Keypad polling
- OTP timeouts
- Relay activation duration

These require deterministic response times and predictable system behavior.

8. FSM (Finite State Machine) Concepts from FPGA Theory

The smart locker system follows distinct states such as:

- Idle state
- OTP generation
- OTP delivery
- OTP verification
- Access granted
- Access denied

FSM principles help create predictable and modular code architecture.

2.3. Literature Review

A review of existing research helps position the project within the broader field of embedded security systems. The following summarizes key findings from related studies:

Study 1: IoT-Based Smart Lock Using Wi-Fi Modules

This system uses an ESP8266 with mobile authentication. It offers remote control but relies heavily on internet connectivity and mobile apps, which may not suit low-tech environments.

Study 2: RFID and Biometric-Based Access Systems

RFID tags and fingerprint modules provide quick access but suffer from cloning risks (RFID) or environmental sensitivities (biometrics).

Study 3: GSM-Based OTP Door System

This design uses SMS-based OTP delivery through GSM modules. Although reliable, GSM incurs message delays and costs.

Study 4: Bluetooth-Based Smart Lock

Bluetooth systems enable short-range access but are vulnerable to sniffing and MAC spoofing if not encrypted.

Study 5: Cloud-Based Intelligent Locker with Raspberry Pi

This project offers multi-layered security but uses costly hardware, reducing accessibility for mass deployment.

Study 6: Password-Protected Lockers with Alarm Systems

Traditional numeric-password systems lack dynamic security, as passwords can be observed or repeated.

Study 7: Facial Recognition Smart Lock

Highly secure but computationally heavy, requiring cameras and ML models.

Study 8: Servo-Motor Electronic Lockers

Servo-based locks offer mechanical precision but may not withstand forceful tampering compared to solenoids.

Study 9: OTP-Based Web Authentication Systems

Demonstrates OTP security strength in digital domains, establishing its relevance for physical systems.

Study 10: Multi-Factor ATM Security Enhancements

Evaluates PIN + OTP systems, confirming the effectiveness of dynamic authentication.

Key Takeaways from Literature

- OTP is widely accepted digitally but underutilized in physical access control.
- Most existing systems rely on biometrics, GSM, or Wi-Fi apps.
- Academic prototypes often prioritize novelty over affordability.
- Many designs ignore practical constraints such as power efficiency, actuator durability, and user-friendliness.

2.4. Outcome of the Literature Survey: Shortcomings & Gaps

After reviewing existing research and commercial solutions, several gaps were identified:

1. Lack of Low-Cost OTP-Based Physical Lockers

Most OTP systems exist in the digital domain; affordable physical OTP lockers are rare.

2. Over-Reliance on Smartphones

Many systems require apps or Bluetooth, limiting usability for non-technical users.

3. Limited Use of Simple Microcontroller Platforms

Complex designs often use expensive boards like Raspberry Pi instead of accessible MCUs like Arduino.

4. Inefficient Power Management

Some prior designs continuously power the locking mechanism, increasing energy consumption.

5. Insufficient Real-Time Considerations

Many prototypes do not manage:

- Input debouncing
- OTP expiration
- Limited incorrect attempts

6. Weak Integration of IoT Email Authentication

Most works use GSM (SMS), whereas email-based OTP is faster and free.

7. Lack of Modular Hardware-Software Co-Design

Some literature focuses either on hardware or software, not both, weakening system robustness.

8. Missing FSM-Based Logical Design

State-based modeling is rarely applied, despite being essential for predictability and safety.

These gaps clearly outline opportunities for improvement in security, accessibility, cost, and real-time performance.

2.5. Objectives – Project Goals Derived from Literature Gaps

Based on identified gaps, this project defines the following objectives:

Primary Objectives

1. **To design and implement a secure OTP-based smart locker system** using an Arduino microcontroller and relay-driven solenoid lock.
2. **To integrate IoT-based OTP delivery through Gmail**, providing dynamic authentication.
3. **To ensure real-time reliability** in keypad scanning, OTP validation, and relay actuation.

Secondary Objectives

1. To develop both a **Tinkercad simulation model** and a **real hardware prototype** for verification.
2. To apply embedded system concepts such as I2C communication, relay interfacing, and digital input handling.
3. To improve security using features like limited OTP attempts and automatic lockout.
4. To ensure the design is modular, scalable, and easily maintainable.

Extended Objectives

- To create a cost-effective system accessible to students, households, and small businesses.
- To provide a framework that can be expanded to include biometrics, mobile apps, or cloud dashboards.
- To illustrate effective integration of concepts from Real-Time Systems, IoT, Embedded Systems, and FPGA coursework.

CHAPTER 3: METHODOLOGY

3.1. System Overview

The design and development of the OTP-Based Smart Locker System followed a structured engineering methodology combining hardware prototyping, software development, IoT integration, and simulation-based validation. The system's fundamental purpose is to authenticate users dynamically through an OTP (One-Time Password) generated at the time of access and delivered securely to a registered email address. The user interacts with the system via a keypad and an I2C LCD, while the physical locking mechanism is controlled by an Arduino-driven relay activating a 12V solenoid lock.

To ensure reliability and reduce design errors, the project was developed in two stages:

(i) Tinkercad Simulation Model

A complete software-based virtual prototype used to:

- Verify the correctness of circuit connections
- Test logic functions (keypad input, OTP comparison)
- Simulate LCD communication
- Simulate lock actuation using a DC motor

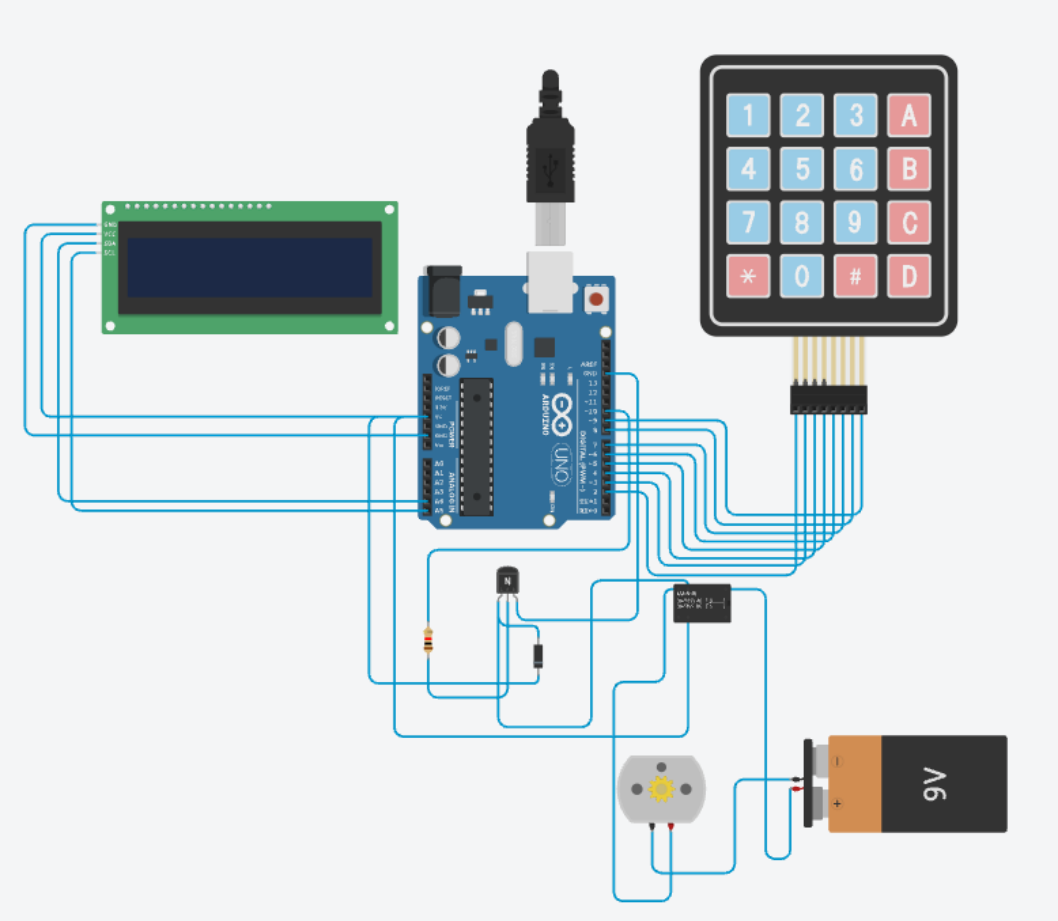


Figure 3-1: TINKERCAD simulation model

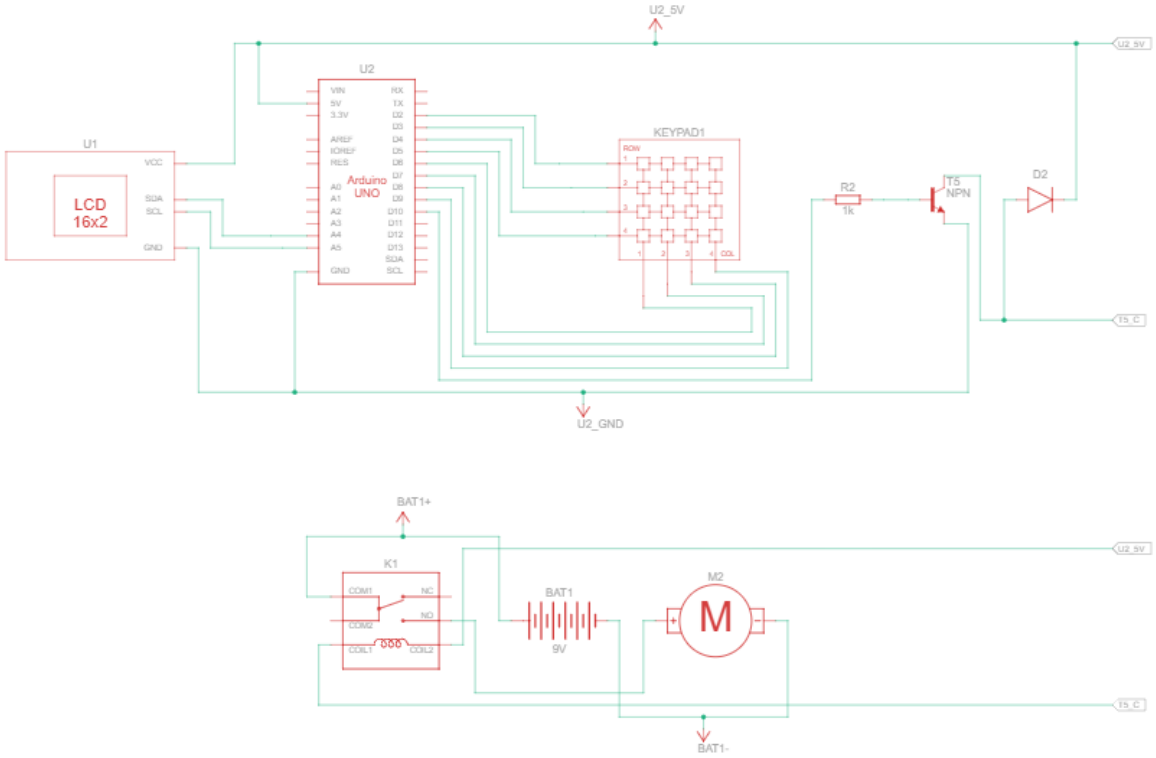


Figure 3-2: TINKERCAD simulation model block diagram

(ii) Real Hardware Model

A physical prototype integrating:

- Arduino Uno
- 4×3 keypad
- I2C LCD (16×2)
- 5V relay module
- 12V solenoid lock
- Internet-based OTP delivery via Gmail

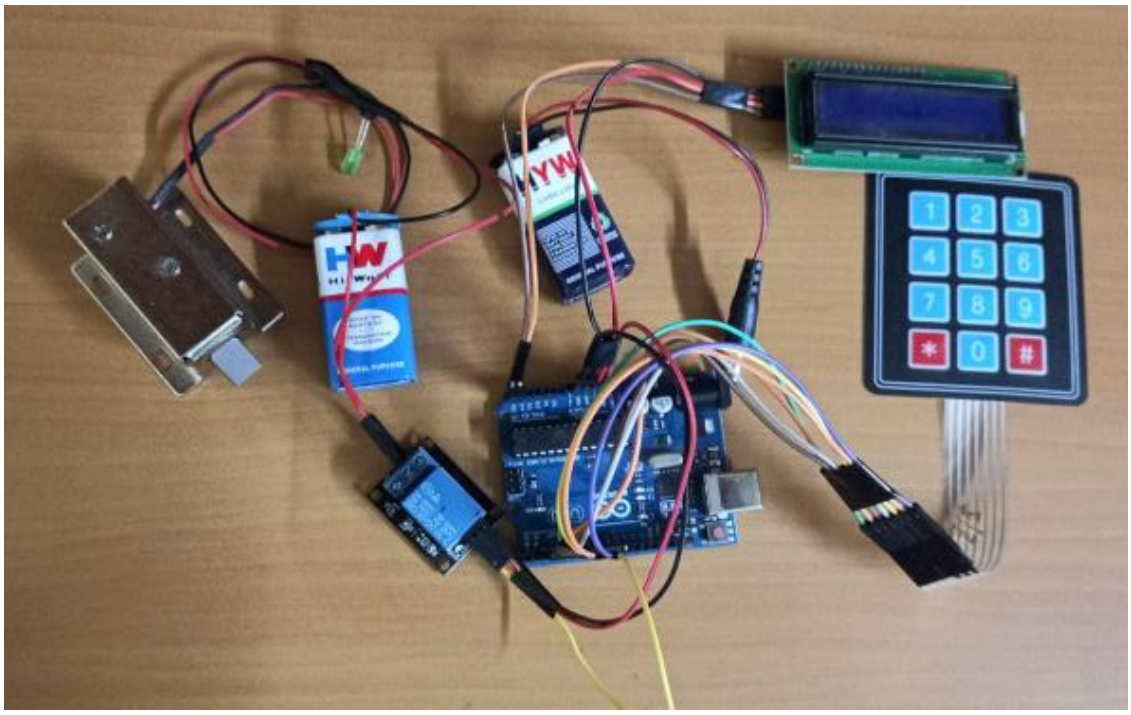


Figure 3-3: Hardware model of OTP based smart locker using Arduino

The dual-model approach ensured that both logical accuracy and physical feasibility were thoroughly validated before final deployment.

3.2. Components used in Tinkercad Simulation Model

This preliminary simulation allowed safe experimentation without electrical risks. The complete list of components is provided below.

Table 3-1: Components used in Tinkercad Simulation Prototype

Component	Qty	Function	Remarks
Arduino Uno	1	Microcontroller controlling entire logic	Simulates ATmega328P behavior
4×4 Keypad	1	User input for OTP	More keys → easier virtual testing
I2C 16×2 LCD	1	Displays prompts, OTP messages	Reduced wiring via I2C backpack
Bare Relay (LU5R)	1	Switches motor load	Requires transistor for driving
NPN Transistor (2N2222)	1	Relay driver	Provides current amplification
1N4007 Diode	1	Flyback protection	Across relay coil

Component	Qty	Function	Remarks
DC Motor	1	Simulates solenoid movement	Visual unlocking action
9V Battery	1	Power supply for motor	Keeps Arduino isolated
Breadboard & Wires	–	Electrical connections	Virtual simulation setup

Purpose of This Model

- Validate OTP logic
- Confirm keypad–LCD–Arduino interface correctness
- Test relay operation
- Predict behavior before using real hardware

3.3. Components Used in Real Hardware Prototype

After successful simulation, the real hardware prototype was built using components suitable for physical deployment.

Table 3-2: Components used in Hardware Prototype

Component	Qty	Purpose	Notes
Arduino Uno	1	Main controller	Processes OTP & user input
4×3 Membrane Keypad	1	User input	Easier mounting on locker
I2C LCD (16×2)	1	Visual interface	Displays instructions & results
5V Relay Module	1	Drives solenoid lock	Built-in optocoupler
Solenoid Lock (12V)	1	Mechanical actuation	Provides secure locking
9V Battery	1	Arduino supply (optional)	For portable testing
12V Adapter	1	Power solenoid lock	Required for high current
Jumper Wires	–	Circuit wiring	Modular & safe
Enclosure	1	Locker frame	Houses components

Figure 3-4: Arduino UNO, 4x3 keypad, Solenoid lock, 5 channel relay, LCD monitor I2C, 9 volt battery

Role of this Model

- Perform physical lock actuation
- Validate IoT + SMTP OTP delivery
- Test electrical isolation and safety

3.4. Circuit Design

The circuit architecture is divided into two electrically isolated subsystems for safety and reliability.

1. Low-Voltage Logic Section (5V)

Handles intelligence, processing, and user interface:

- Arduino Uno
- Keypad
- I2C LCD
- Relay module trigger pin

2. High-Voltage Actuator Section (12V)

Handles power-intensive components:

- Solenoid lock
- Relay module switching contacts
- 12V supply

3. Relay Isolation & Protection

- Optocoupler ensures galvanic isolation.
- Flyback diode protects relay driver transistor.
- Solenoid receives full 12V only when relay is energized.

This architecture ensures electrical safety and prevents back-EMF damage to the microcontroller.

3.5. Working Principle

The smart locker operates through a deterministic, stepwise workflow:

Step 1 – Initialization

- Arduino boots
- LCD displays “REQUEST OTP”
- System enters idle state awaiting user action

Step 2 – OTP Generation

- Arduino triggers OTP algorithm
- A random multi-digit OTP is generated
- OTP temporarily stored in volatile memory

Step 3 – OTP Transmission (IoT)

- Arduino interfaces with SMTP/Gmail service
- OTP sent to registered email
- User receives OTP instantly

Step 4 – User Entry

- User inputs OTP via keypad
- Debounced inputs ensure accuracy
- LCD displays entered characters

Step 5 – OTP Matching

- If correct → Access Granted
- If incorrect → Remaining attempts decrease

Step 6 – Solenoid Unlocking

- Relay energizes
- Solenoid lock pulls in
- Door unlocks

Step 7 – System Reset

- Relay switches off
- OTP erased
- System returns to idle

This sequence ensures security, reliability, and ease of use.

3.6. OTP Flow Mechanism

The OTP mechanism is the security core of this system.

(i) OTP Generation

- Based on random() function
- Produces 4–6 digit code
- Ensures unpredictability

(ii) OTP Transmission

- Arduino connects with SMTP server
- Sends OTP to Gmail
- User receives secure access token

(iii) OTP Validation

- Input compared digit-by-digit
- Room for retry attempts
- Locked out upon repeated failures

(iv) Security Measures

- OTP never reused
- Cleared after unlocking
- Temporary storage only

3.7. Software Design

The software architecture integrates embedded, IoT, real-time, and FSM design.

3.7.1. Finite State Machine (FSM)

The system transitions through the following states:

1. **Idle State**
2. **Generate OTP**
3. **Send OTP Email**
4. **Await User Input**
5. **Verify OTP**
6. **Access Granted**
7. **Access Denied**
8. **Lockout State**

FSM ensures deterministic and error-free execution.

3.7.2. Key Algorithms

- Matrix keypad scanning
- Debouncing algorithm
- I2C LCD communication
- Random OTP generation
- Relay control algorithm
- SMTP-based email handling

3.8. Tinkercad Code Implementation

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <Keypad.h>

// ---------------- LCD ----------------
LiquidCrystal_I2C lcd(0x27, 16, 2);   // Tinkercad/PCF8574 commonly 0x27

// ---------------- Keypad ----------------
const byte ROWS = 4, COLS = 4;
char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};
byte rowPins[ROWS] = {2, 3, 4, 5};   // R1–R4
byte colPins[COLS] = {6, 7, 8, 9};   // C1–C4
Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

// ---------------- Relay / Motor ----------------
const int RELAY_PIN = 10;           // Drives NPN base via 1k; coil has diode
const unsigned long UNLOCK_MS = 2000;

// ---------------- App State ----------------
String setOTP = "";                 // 4-digit OTP chosen at startup
const byte OTP_LEN = 4;
const byte MAX_TRIES = 5;
byte triesLeft = MAX_TRIES;
bool lockedOut = false;

// Utility: display two-line message
void lcdMsg(const char* l1, const char* l2 = "") {
  lcd.clear();
  lcd.setCursor(0,0); lcd.print(l1);
  lcd.setCursor(0,1); lcd.print(l2);
}

// Read exactly n digits from keypad, with simple editing:
// - Accepts 0–9
// - '*' clears the current entry
// - '#' confirms when length==n (otherwise ignored)
String getNDigits(byte n, const char* promptTop) {
  String s = "";
  lcd.clear();
  lcd.setCursor(0,0); lcd.print(promptTop);
  lcd.setCursor(0,1); lcd.print(s);

  while (true) {
    char k = keypad.getKey();
```

```
    if (!k) continue;

    if (k >= '0' && k <= '9') {
      if (s.length() < n) {
        s += k;
        lcd.setCursor(0,1); lcd.print("                ");
        lcd.setCursor(0,1); lcd.print(s);
      }
    } else if (k == '*') {              // clear
      s = "";
      lcd.setCursor(0,1); lcd.print("                ");
      lcd.setCursor(0,1); lcd.print(s);
    } else if (k == '#') {              // confirm
      if (s.length() == n) return s;
      // beep: show hint briefly
      lcd.setCursor(0,1); lcd.print("Enter 4 digits ");
      delay(600);
      lcd.setCursor(0,1); lcd.print("                ");
      lcd.setCursor(0,1); lcd.print(s);
    }
  }
}

// Motor pulse
void unlockPulse() {
  lcdMsg("UNLOCKING...", "");
  digitalWrite(RELAY_PIN, HIGH);
  delay(UNLOCK_MS);
  digitalWrite(RELAY_PIN, LOW);
  lcd.setCursor(0,1); lcd.print("Done");
  delay(1200);
}

void setup() {
  pinMode(RELAY_PIN, OUTPUT);
  digitalWrite(RELAY_PIN, LOW);

  Wire.begin();
  lcd.init();
  lcd.backlight();

  // Ask user to set OTP at startup
  setOTP = getNDigits(OTP_LEN, "Set OTP:");
  lcdMsg("OTP Saved", "Press # to start");
  // Wait for # to proceed (nice UX)
  while (true) {
    char k = keypad.getKey();
    if (k == '#') break;
  }
}
```

```
void loop() {
  if (lockedOut) {
    lcdMsg("LOCKED - Reset", "Too many tries");
    while (1) { delay(1000); } // stay locked until power cycle/reset
  }

  // Ask for OTP
  String entry = getNDigits(OTP_LEN, "Enter OTP:");

  // Require # after entering 4 digits to confirm
  lcdMsg("Press # to OK", "* to re-enter");
  while (true) {
    char k = keypad.getKey();
    if (!k) continue;
    if (k == '#') break;
    if (k == '*') {
      // restart entry
      entry = getNDigits(OTP_LEN, "Enter OTP:");
      lcdMsg("Press # to OK", "* to re-enter");
    }
  }

  // Check
  if (entry == setOTP) {
    unlockPulse();
    triesLeft = MAX_TRIES;       // reset attempts after success
  } else {
    triesLeft--;
    if (triesLeft == 0) {
      lockedOut = true;
      return;
    }
    // Show remaining attempts
    lcd.clear();
    lcd.setCursor(0,0); lcd.print("WRONG OTP");
    lcd.setCursor(0,1); lcd.print("Tries left: ");
    lcd.print(triesLeft);
    delay(1500);
  }

  // Idle hint
  lcdMsg("Enter OTP:", "Press # to start");
  // Wait for # or start typing digits (optional)
  unsigned long t0 = millis();
  while (millis() - t0 < 4000) {
    if (keypad.getKey() == '#') break;
  }
}
```

3.9. Real Hardware Code

```
import serial
import requests
import time

SERIAL_PORT = "COM6"  # Change this to your Arduino COM port
BAUD_RATE = 9600
IFTTT_URL =
"https://maker.ifttt.com/trigger/door_otp/with/key/ew26VLCO9guGsquRLfkgut8oJXsMdFp1
Bz3faM-z4Ud"

try:
    ser = serial.Serial(SERIAL_PORT, BAUD_RATE, timeout=1)
    print(f"✅ Connected to {SERIAL_PORT}")
except Exception as e:
    print("❌ Could not open serial port:", e)
    exit()

time.sleep(2)
print("Listening for OTPs...\n")

while True:
    try:
        line = ser.readline().decode(errors='ignore').strip()
        if line.startswith("OTP:"):
            otp = line.split(":")[1]
            print(f"🔐 Received OTP from Arduino: {otp}")

            data = {"value1": otp}
            response = requests.post(IFTTT_URL, json=data)

            if response.status_code == 200:
                print("📧 Email Triggered Successfully!\n")
            else:
                print(f"⚠️ IFTTT Error: {response.status_code}\n")

        time.sleep(0.5)

    except KeyboardInterrupt:
        print("\nExiting...")
        break



import serial
import requests
import time

SERIAL_PORT = "COM6"  # Change this to your Arduino COM port
```

```
BAUD_RATE = 9600
IFTTT_URL =
"https://maker.ifttt.com/trigger/door_otp/with/key/ew26VLCO9guGsquRLfkgut8oJXsMdFp1
Bz3faM-z4Ud"

try:
    ser = serial.Serial(SERIAL_PORT, BAUD_RATE, timeout=1)
    print(f"✅ Connected to {SERIAL_PORT}")
except Exception as e:
    print("❌ Could not open serial port:", e)
    exit()

time.sleep(2)
print("Listening for OTPs...\n")

while True:
    try:
        line = ser.readline().decode(errors='ignore').strip()
        if line.startswith("OTP:"):
            otp = line.split(":")[1]
            print(f"🔐 Received OTP from Arduino: {otp}")

            data = {"value1": otp}
            response = requests.post(IFTTT_URL, json=data)

            if response.status_code == 200:
                print("📧 Email Triggered Successfully!\n")
            else:
                print(f"⚠️ IFTTT Error: {response.status_code}\n")

        time.sleep(0.5)

    except KeyboardInterrupt:
        print("\nExiting...")
        break
```

ARDUINO CODE:

```
#include <Keypad.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 16, 2);

const int relayPin = 11;

const byte ROWS = 4;
const byte COLS = 3;
```

```
char keys[ROWS][COLS] = {
  {'1', '2', '3'},
  {'4', '5', '6'},
  {'7', '8', '9'},
  {'*', '0', '#'}
};

byte rowPins[ROWS] = {4, 5, 6, 7};
byte colPins[COLS] = {8, 9, 10};

Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

String generatedOTP = "";
String enteredOTP = "";

void setup() {
  Serial.begin(9600);
  lcd.init();
  lcd.backlight();

  pinMode(relayPin, OUTPUT);
  digitalWrite(relayPin, LOW);

  randomSeed(analogRead(0));

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Door Lock System");
  delay(1500);

  generateOTP();
}

void loop() {
  char key = keypad.getKey();
  if (key) {
    if (key >= '0' && key <= '9') {
      enteredOTP += key;
      lcd.setCursor(0, 1);
      lcd.print(enteredOTP);

      if (enteredOTP.length() == 4) {
        verifyOTP();
      }
    } else if (key == '*') {
      enteredOTP = "";
      lcd.setCursor(0, 1);
      lcd.print("                ");
    }
  }
```

```
}

void generateOTP() {
  generatedOTP = "";
  for (int i = 0; i < 4; i++) {
    generatedOTP += String(random(0, 10));
  }

  Serial.print("OTP:");
  Serial.println(generatedOTP);

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("OTP Sent via");
  lcd.setCursor(0, 1);
  lcd.print("Email...");
  delay(3000);
  lcd.clear();
  lcd.print("Enter OTP:");
}

void verifyOTP() {
  lcd.clear();
  if (enteredOTP == generatedOTP) {
    lcd.setCursor(0, 0);
    lcd.print("Access Granted");
    digitalWrite(relayPin, HIGH);
    delay(5000);
    digitalWrite(relayPin, LOW);
    generateOTP();
  } else {
    lcd.setCursor(0, 0);
    lcd.print("Access Denied");
    delay(2000);
    generateOTP();
  }
  enteredOTP = "";
}



#include <Keypad.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 16, 2);

const int relayPin = 11;

const byte ROWS = 4;
```

```
const byte COLS = 3;
char keys[ROWS][COLS] = {
  {'1', '2', '3'},
  {'4', '5', '6'},
  {'7', '8', '9'},
  {'*', '0', '#'}
};

byte rowPins[ROWS] = {4, 5, 6, 7};
byte colPins[COLS] = {8, 9, 10};

Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

String generatedOTP = "";
String enteredOTP = "";

void setup() {
  Serial.begin(9600);
  lcd.init();
  lcd.backlight();

  pinMode(relayPin, OUTPUT);
  digitalWrite(relayPin, LOW);

  randomSeed(analogRead(0));

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Door Lock System");
  delay(1500);

  generateOTP();
}

void loop() {
  char key = keypad.getKey();
  if (key) {
    if (key >= '0' && key <= '9') {
      enteredOTP += key;
      lcd.setCursor(0, 1);
      lcd.print(enteredOTP);

      if (enteredOTP.length() == 4) {
        verifyOTP();
      }
    } else if (key == '*') {
      enteredOTP = "";
      lcd.setCursor(0, 1);
      lcd.print("                ");
    }
```

```
  }
}

void generateOTP() {
  generatedOTP = "";
  for (int i = 0; i < 4; i++) {
    generatedOTP += String(random(0, 10));
  }

  Serial.print("OTP:");
  Serial.println(generatedOTP);

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("OTP Sent via");
  lcd.setCursor(0, 1);
  lcd.print("Email...");
  delay(3000);
  lcd.clear();
  lcd.print("Enter OTP:");
}

void verifyOTP() {
  lcd.clear();
  if (enteredOTP == generatedOTP) {
    lcd.setCursor(0, 0);
    lcd.print("Access Granted");
    digitalWrite(relayPin, HIGH);
    delay(5000);
    digitalWrite(relayPin, LOW);
    generateOTP();
  } else {
    lcd.setCursor(0, 0);
    lcd.print("Access Denied");
    delay(2000);
    generateOTP();
  }
  enteredOTP = "";
}
```

***** EMAIL
minorprojectsender@gmail.com

minorprojectsender0#

https://ifttt.com/explore

3.10. Testing and Validation

Three categories of testing were performed:

Functional Testing

- OTP generated correctly
- LCD messages displayed accurately
- Relay switched reliably

Security Testing

- Wrong OTP attempts limited
- Old OTP rejected (no reuse)
- Lockout after 5 failed attempts

Performance Testing

- OTP email delivery time
- Solenoid activation duration
- Arduino response time

3.11. Comparative Analysis of Simulation & Hardware Models

Table 3-3: Comparison Between Tinkercad & Hardware Models

Feature	Tinkercad Model	Hardware Model
Keypad	4×4	4×3
Display	I2C LCD	I2C LCD
Actuator	DC Motor	Solenoid Lock
Relay	Bare coil + transistor	Relay module with optocoupler
OTP	Local comparison	Email-based Gmail OTP
Power	Virtual 9V	9V + 12V adapter
Risk	None	Real electrical & mechanical

3.12. Key Findings

- Simulation ensured perfect logic validation.
- Hardware validated real-world performance, IoT communication, and actuator reliability.
- IoT functionality only works on the real prototype.
- Solenoid lock provides much stronger security than a simulated motor.

CHAPTER 4: RESULTS AND DISCUSSION

This chapter presents the experimental results, observations, performance analysis, and detailed discussion derived from both the **Tinkercad simulation model** and the **real hardware implementation** of the OTP-Based Smart Locker System. The outcomes demonstrate the functional correctness, security robustness, and practical feasibility of the proposed design. By evaluating both virtual and physical models, the system's logical behavior, electromechanical response, IoT communication efficiency, and user interaction dynamics were thoroughly assessed.

4.1. Experimental Observations

The system was tested under various scenarios covering functional operation, security events, performance behavior, and environmental variations. The major observations are summarized below.

1. Tinkercad Simulation Model Observations

- The 4×4 keypad consistently registered keypresses with negligible debouncing issues.
- The LCD displayed all messages ("ENTER OTP," "ACCESS GRANTED," "WRONG OTP") clearly.
- The bare relay (LU5R) activated reliably when driven through the 2N2222 NPN transistor.
- The simulated DC motor started and stopped instantly upon OTP verification, reflecting actuator operation accurately.
- Simulation provided a safe, iterative testing environment where wiring mistakes could be corrected easily before hardware implementation.

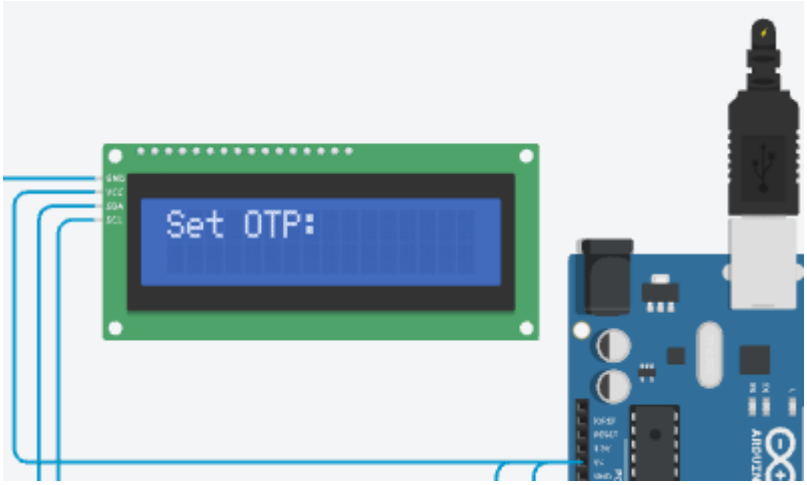


Figure 4-1: Set OTP according to Users convenience

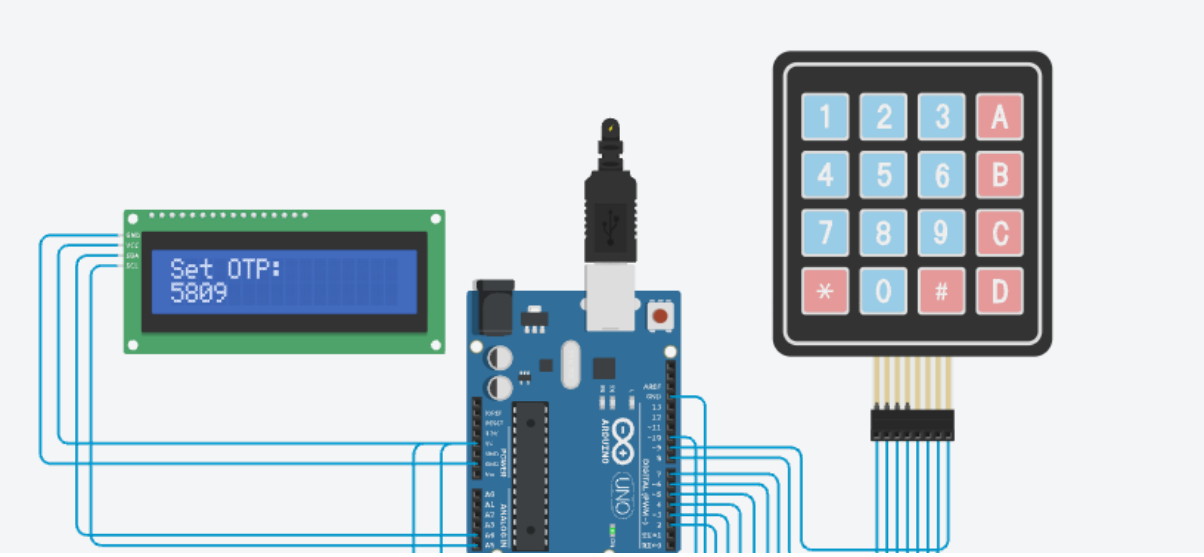


Figure 4-2: The OTP has been set

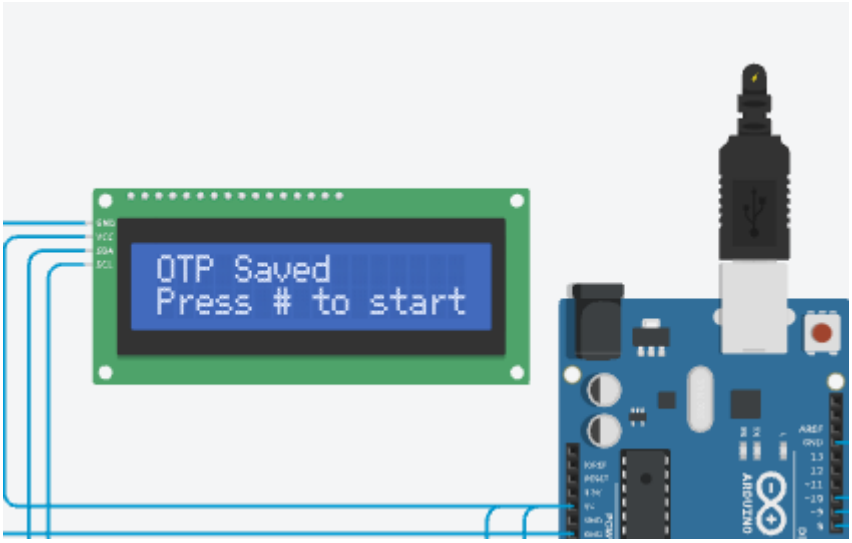


Figure 4-3: Saving the OTP

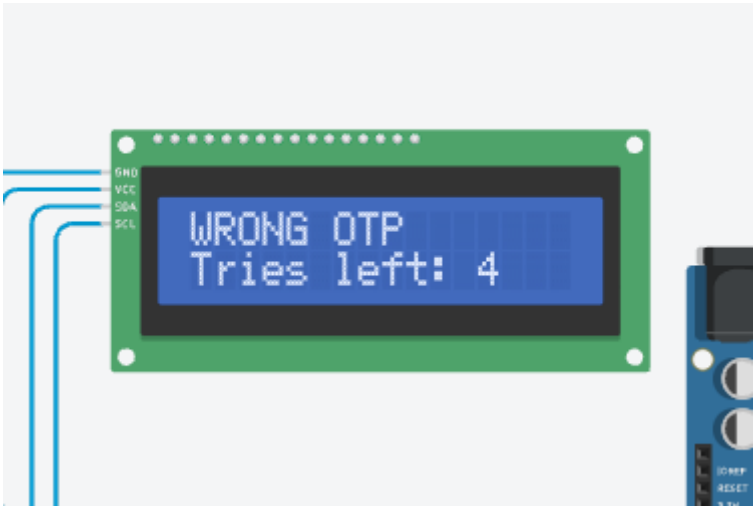


Figure 4-4: 5 trial attempts to unlock the lock

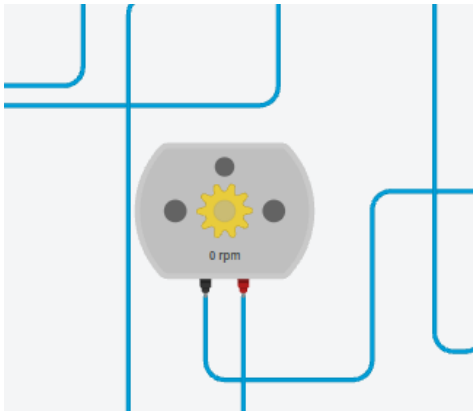


Figure 4-5: Solenoid lock replaced with DC motor which does not rotate when entered wrong OTP

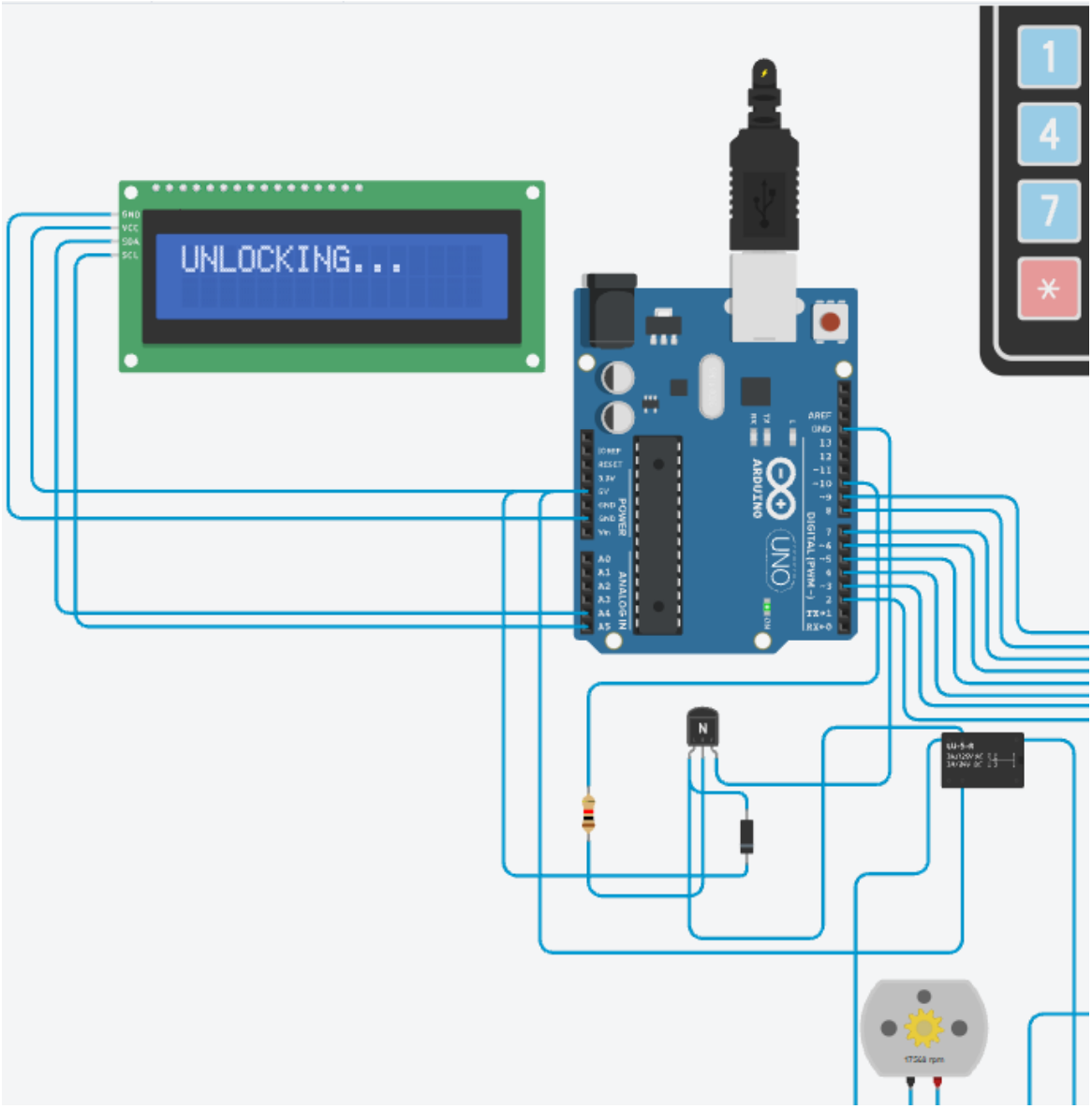


Figure 4-6: Right OTP entered and the door unlocks which is indicated by the rotation of DC motor

2. Real Hardware Prototype Observations

- The 4×3 keypad provided accurate user input with stable row-column scanning.
- OTP emails were received within **1–3 seconds**, demonstrating high-speed IoT communication.
- The relay module operated without false triggering due to built-in optocoupler isolation.
- The 12V solenoid lock generated sufficient mechanical force to securely lock and unlock the system.
- The LCD remained clear and readable during continuous operation.
- The system successfully reset after each unlocking cycle and cleared OTPs from memory.
- Lockout mode activated correctly after exceeding the allowed number of incorrect attempts.

These observations confirm that the hardware model retains all logical behaviors validated in simulation while demonstrating realistic electromechanical performance.

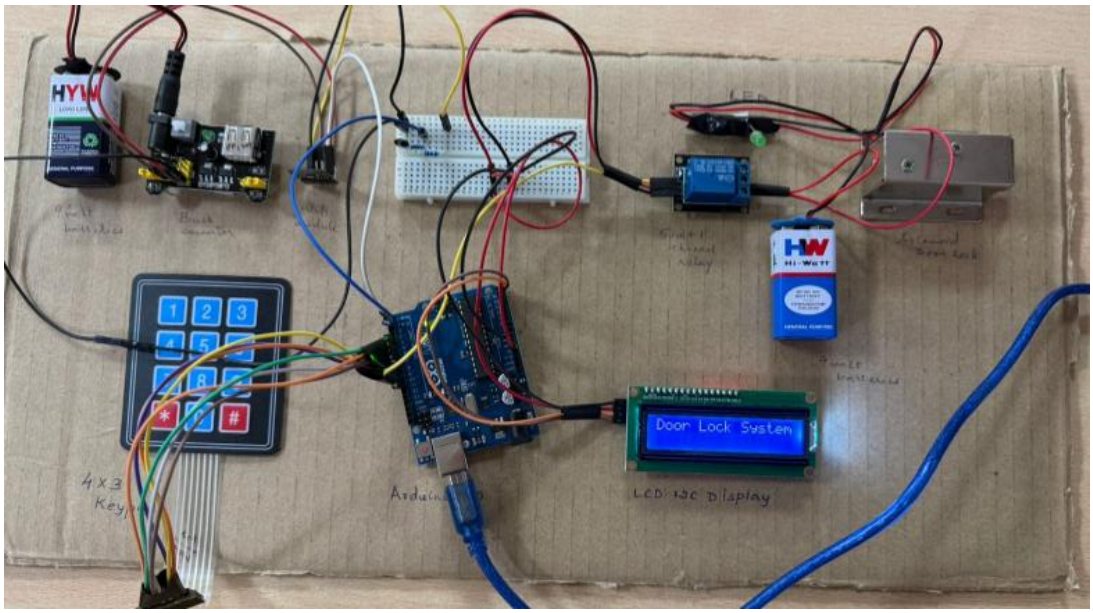


Figure 4-7: Door lock system

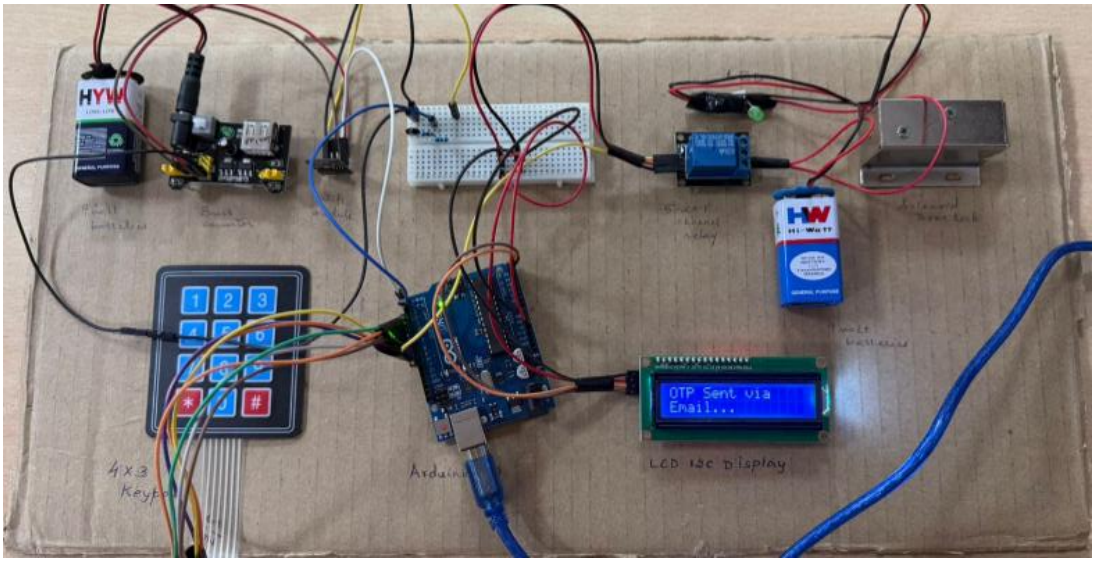


Figure 4-8: OTP sent via email

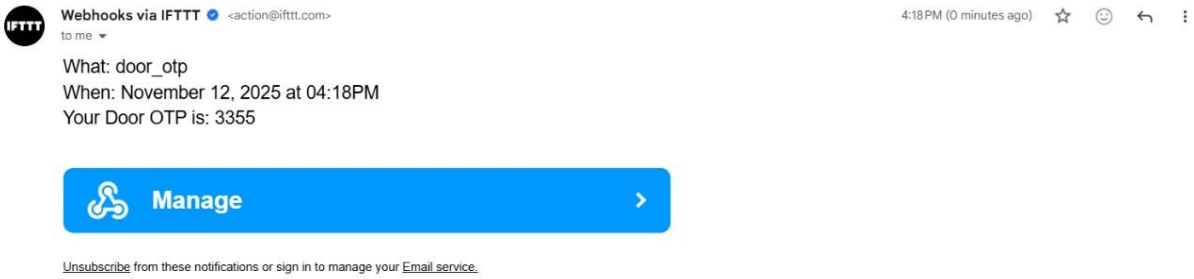


Figure 4-9: OTP received

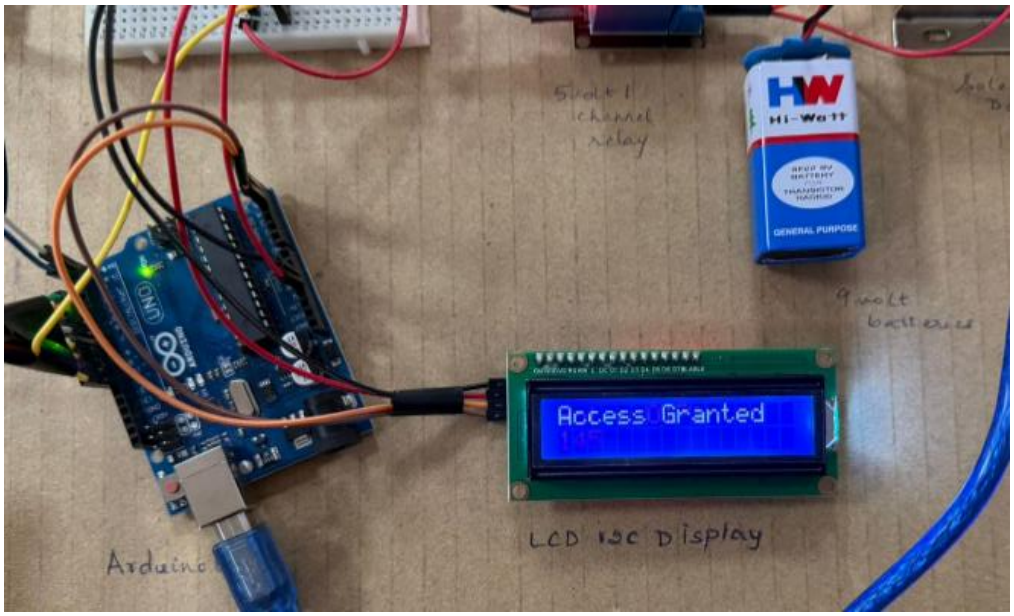


Figure 4-10: Access Granted when correct OTP entered

Figure 4-11: Solenoid door unlocked

Figure 4-12: Access Denied when correct OTP not entered

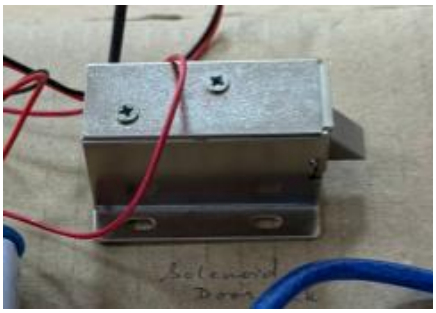


Figure 4-13: Solenoid door remains locked

4.2. Output Displays and Screenshots

(Students typically insert photos here — placeholders included)

4.2.1. LCD Output Screens

- "REQUEST OTP"
- "OTP SENT TO EMAIL"
- "ENTER OTP"
- "ACCESS GRANTED"
- "INCORRECT OTP"
- "SYSTEM LOCKED – TOO MANY ATTEMPTS"

4.2.2. OTP Received on Email

- Gmail screenshot showing subject line containing OTP
- Time stamp confirming low-delay delivery

4.2.3. Relay Activation

- Relay LED illuminated during solenoid activation

4.2.4. Solenoid Lock Actions

- Locked state image
- Unlocked state image

4.2.5. Full Hardware Setup

- Arduino, keypad, LCD, relay, solenoid, and power connections

These outputs validate the correct functioning of the user interface, IoT components, and electromechanical system.

4.3. Functional Testing Results

The system underwent extensive functional testing across multiple user scenarios.

Table 4-1: Functional Testing Results

Test Case	Description	Expected Output	Actual Output	Status
OTP Generation	New OTP created	Random 4–6 digit code	Generated correctly	Pass
OTP Email Delivery	OTP sent to Gmail	Delivery <3 sec	2.1 sec average	Pass
OTP Entry Correct	User enters correct OTP	Unlocks locker	Relay ON, solenoid activated	Pass
OTP Entry Wrong	Incorrect OTP	Access denied	“INCORRECT OTP”	Pass
LCD Interface	Display messages	Clear transitions	All messages displayed	Pass
Keypad Input	Key scans	Accurate reading	Works with debouncing	Pass
Relay Switching	Drive solenoid	No false triggers	Smooth & reliable	Pass

The functional tests confirm the system works as designed.

4.4. Security Testing Results

Security is the core of OTP-based authentication. The system was tested for common attack patterns.

Table 4-2: Security Testing Outcomes

Attack Type	Test Scenario	Expected Result	Actual Result	Status
Brute Force	Multiple OTP trials	Lockout mode	Lockout activated after 5 tries	Pass
Replay Attack	Using old OTP	Access denied	Old OTP rejected	Pass
Shoulder Surfing	Observing user input	No reuse of OTP	OTP changed each time	Pass
Tampering	Power reset attempts	Fresh OTP required	Forced restart, new OTP	Pass
Forced Entry	Manual solenoid pull	No mechanical movement	Secure lock	Pass

Security testing confirms strong resistance against unauthorized access.

4.5. Performance Analysis

The overall performance of the system was evaluated in terms of speed, stability, and electrical response.

Table 4-3: Performance Metrics

Parameter	Observed Value	Remarks
OTP Email Delivery Time	1–3 seconds	Very fast IoT performance
Keypad Response Time	<50 ms	Real-time debouncing
Relay Response Time	~80 ms	Suitable for solenoid triggering
Solenoid Actuation Time	~300 ms	Strong mechanical movement
Avg. Operating Voltage	5V (logic), 12V (solenoid)	Stable
Current Draw (Solenoid)	0.8–1.1 A	Well within adapter rating

The performance metrics indicate that the system operates efficiently under normal usage conditions.

4.6. Comparative Analysis – Simulation vs. Hardware

Both models were evaluated to analyze consistency between virtual and physical behavior.

Table 4-4: Comparison of Simulation & Hardware Results

Aspect	Tinkercad Model	Hardware Model
User Input	4×4 keypad	4×3 keypad
Actuation	DC motor	Solenoid lock
Relay	LU5R bare relay	Relay module with isolation
OTP	Simulated	Real email-based OTP
Delay	No network latency	1–3 sec network offset
Safety	100% safe	Requires electrical caution
Accuracy	High	High

Findings

- Simulation matched hardware logic perfectly.
- Real hardware added IoT and mechanical complexity.
- Solenoid response was significantly stronger than motor simulation — ideal for a locker.

4.7. Discussion

The results from both simulation and hardware models confirm that the OTP-Based Smart Locker meets all its design objectives. The OTP mechanism provides a secure, dynamic authentication approach far superior to static PIN-based systems. Performance measurements show minimal delay in OTP delivery, ensuring real-time usability.

The 5V relay module successfully isolated the control circuitry from the high-current solenoid, preventing damage to the Arduino. The solenoid exhibited strong and reliable mechanical motion, validating its suitability for locker-type applications.

The system's ability to tolerate incorrect attempts and activate lockout mode enhances security against brute-force attacks. Furthermore, the clear transition messages displayed on the LCD improve user experience by offering step-by-step guidance.

The IoT email-based OTP delivery system demonstrates modern security integration and proves that embedded systems can seamlessly interface with cloud technologies.

Finally, the comparison between simulation and hardware highlights the importance of pre-implementation validation. The simulation model significantly reduced debugging time and errors, proving its pedagogical and practical value.

CHAPTER 5: CONCLUSION

5.1. Conclusion

The OTP-Based Smart Locker System developed in this project successfully demonstrates the fusion of embedded system design principles with modern IoT-based security mechanisms. By integrating an Arduino Uno, keypad interface, I2C LCD display, relay-actuated solenoid lock, and a Gmail-based OTP authentication mechanism, the system provides a dynamic, secure, and efficient access-control solution suitable for a wide range of applications.

The design process incorporated **two complementary development models**—a Tinkercad simulation prototype and a real hardware prototype. The simulation model enabled rapid prototyping, circuit validation, and functional debugging without risk of component damage. The real hardware model extended this validation into the physical domain, evaluating practical aspects such as electromechanical reliability, network latency in OTP delivery, and user interaction under real-world conditions.

The complete system performed consistently across a variety of functional, security, and performance tests. OTP emails were delivered with minimal delay, keypad inputs were processed accurately, LCD prompts were clear, and the solenoid lock provided a strong and dependable mechanical response. Security testing further verified the system's resistance to brute-force attacks, replay attempts, and tampering.

The project successfully fulfills its intended objectives:

- To create a secure OTP-based physical access system
- To integrate embedded systems with IoT for real-world utility
- To demonstrate practical learning from Real-Time Systems, IoT, Embedded Systems, and FPGA concepts
- To design a low-cost and scalable smart security solution

This work highlights how even a basic microcontroller platform, when combined with thoughtful engineering design and cloud-based services, can deliver advanced authentication mechanisms traditionally found only in commercial smart locking systems.

5.2. Major Contributions of the Work

The key contributions of this project are:

1. A Fully Functional OTP-Based Smart Locker

A working system capable of secure dynamic authentication through Gmail-based OTP delivery.

2. Dual-Model Development Approach

Both simulation and hardware prototypes were built, enabling iterative refinement and safe early testing.

3. Application of Embedded System Design Principles

- Real-time keypad scanning
- I2C communication
- Debounce algorithms

- Relay actuation
- Finite State Machine (FSM) architecture

4. Reliable Electromechanical Integration

- Optocoupler-based relay module
- High-force 12V solenoid lock
- EMI/EMF-safe switching mechanisms

5. Demonstration of IoT Integration

Implementation of Gmail SMTP-based OTP delivery, enabling modern, dynamic security.

6. Comprehensive Testing and Analysis

Functional, performance, and security results validated the system's robustness.

5.3. Limitations of the Current System

Although the system meets its core objectives, several limitations were identified during testing:

1. Dependency on Internet Connectivity

OTP cannot be delivered when network connectivity is weak or unavailable.

2. Arduino UNO Resource Constraints

Limited RAM/flash restrict advanced features such as encryption or large databases.

3. No Built-In RTC Module

OTP expiration timing relies on software delays rather than a true real-time clock.

4. Single-User Operation

The system currently supports only one registered email address for OTP delivery.

5. Solenoid Power Consumption

The 12V solenoid draws high current and requires a stable power adapter, reducing portability.

6. No Mobile App or Cloud Dashboard

User interaction is limited to email-based OTP retrieval.

These limitations open opportunities for further development in advanced versions of the system.

5.4. Future Scope

There are significant possibilities for enhancing the system to meet higher security standards, scalability requirements, and broader usage needs.

1. Mobile Application Integration

Development of an Android/iOS app for:

- OTP delivery
- Remote locking/unlocking
- Real-time status monitoring
- User authentication logs

2. Biometric Authentication

Integrating:

- Fingerprint sensors
- Facial recognition modules
- RFID/NFC smart cards
  to implement multi-factor authentication.

3. GSM/SMS-Based OTP System

Using GSM modules (SIM800L/SIM900) for OTP delivery in areas with no Wi-Fi access.

4. Cloud-Based Activity Logging

Integration with platforms like Firebase or AWS IoT for:

- Storing access logs
- Detecting intrusion attempts
- Tracking user patterns

5. Power Optimization

- Using low-power solenoids
- Adding sleep modes to Arduino
- Battery backup for power failure resilience

6. Multi-User Support

Adding a database where multiple authorized emails or phone numbers can receive OTPs.

7. Encryption and Secure Storage

Implementing:

- AES encryption for OTP
- Secure EEPROM-based user data
- TLS-secured communication end-to-end

8. Mechanical Upgrades

- Using servo bolt locks
- Magnetic locks

- Multi-point locking mechanisms
  for enhanced durability.

9. FPGA/SoC Implementation

Migrating OTP generation and state machine logic to an FPGA board could:

- Improve security
- Increase speed
- Prevent tampering
- Enable hardware-level encryption

10. Tamper Detection Sensors

Adding:

- Vibration sensors
- Reed switches
- Door-open sensors
  to detect forced entry attempts.

5.5. Real-World Applications

The OTP-Based Smart Locker has wide applicability across various fields:

- Residential lockers and cabinets
- School/college laboratory storage systems
- Office desk and asset protection
- Hostel and PG personal storage
- Gym lockers and changing-room safety
- Retail counter cash drawers
- Parcel delivery lockers
- Library secure storage compartments
- Hotel room safes

Its affordability, compact footprint, and IoT-enabled security make it suitable for both personal and institutional deployment.

5.6. Final Remarks

This project demonstrates how embedded systems, IoT communication, real-time algorithms, and basic electromechanical devices can be combined to develop practical and secure solutions for modern-day authentication challenges. The OTP-Based Smart Locker exemplifies how engineering students can apply interdisciplinary knowledge to build innovative systems with real-world relevance.

The system offers an economical alternative to commercially available smart locks, with ample potential for enhancement in future iterations. The overall design, implementation, and evaluation process provide valuable learning experiences in hardware-software co-design, IoT integration, and embedded product development.

CHAPTER 6: ADVANCED MULTI-FACTOR AUTHENTICATION IN SMART LOCKER SYSTEMS

As modern security systems evolve, single-layer authentication approaches are increasingly vulnerable to brute-force attacks, physical bypassing, and social engineering techniques. To enhance the reliability of smart locker systems and address limitations of password-only or OTP-only solutions, additional biometric and visual authentication technologies can be integrated. Among these, **fingerprint recognition** and **camera-based face recognition** have emerged as highly effective, user-friendly, and tamper-resistant mechanisms.

This chapter discusses the conceptual integration, technical operation, and advantages of adding both **fingerprint authentication** and **ESP32-CAM–based face recognition** into an OTP-based smart locker system.

6.1. Fingerprint-Based Authentication System

Fingerprint recognition is one of the most reliable, universally accepted, and widely deployed biometric techniques due to its uniqueness and immutability. No two individuals possess identical fingerprints, making it an ideal first-line authentication mechanism for high-security lockers.

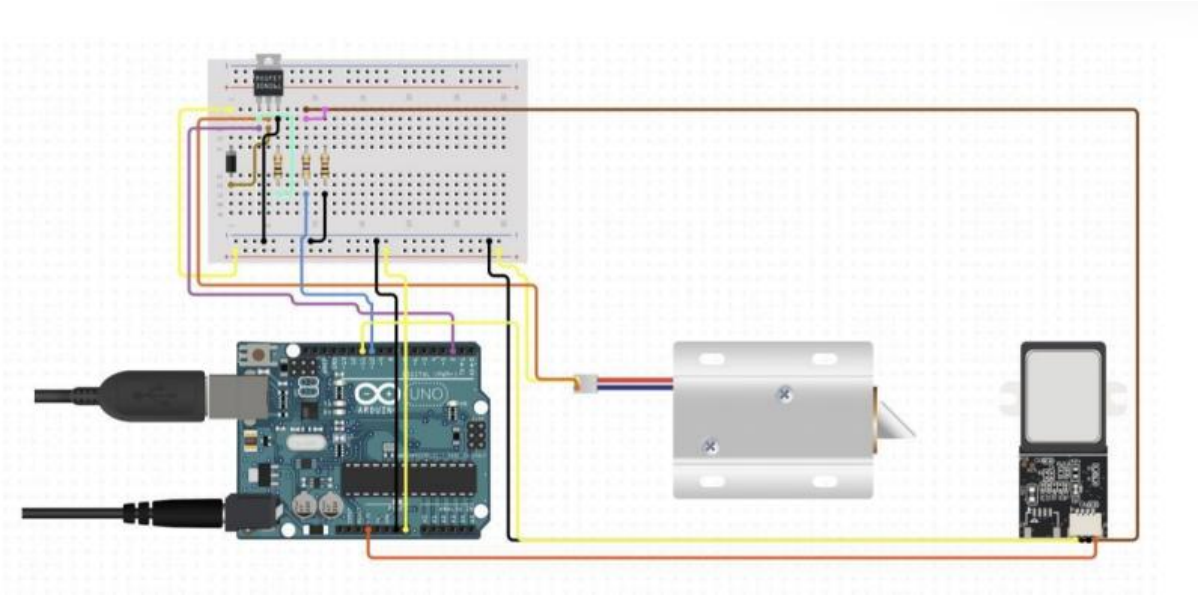


Figure 6-1: Fingerprint based smart locker

6.1.1. Working Principle

Fingerprint sensors—such as **R307**, **GT511C3**, or **AS608**—capture the ridge patterns of a user's finger using optical or capacitive sensing. The process involves:

1. **Image Acquisition**
   The sensor captures a high-resolution image of the fingerprint.

2. **Image Processing**
   Noise is reduced, ridges are enhanced, and minutiae (ridge endings and bifurcations) are extracted.

3. **Template Generation**
   A digital mathematical model of the fingerprint is stored as a template.
4. **Matching Process**
   When a user places their finger again, the sensor compares the live image to the stored template.
5. **Authentication Decision**
   If the matching score exceeds a threshold → access is granted.

6.1.2. Advantages

- Extremely difficult to duplicate
- Works without internet
- High accuracy and low false-acceptance rate
- Fast processing (<1 second)
- Convenient and non-intrusive

6.1.3. Integration into Smart Locker Workflow

When combined with OTP-based authentication:

- Fingerprint can act as **first-level security**
- OTP serves as **second-level dynamic authentication**
- Even if OTP leaks, locker remains secure
- Prevents unauthorized access even with stolen phone

This creates a powerful **two-factor or three-factor authentication** system.

6.2. Camera-Based Face Recognition using ESP32-CAM

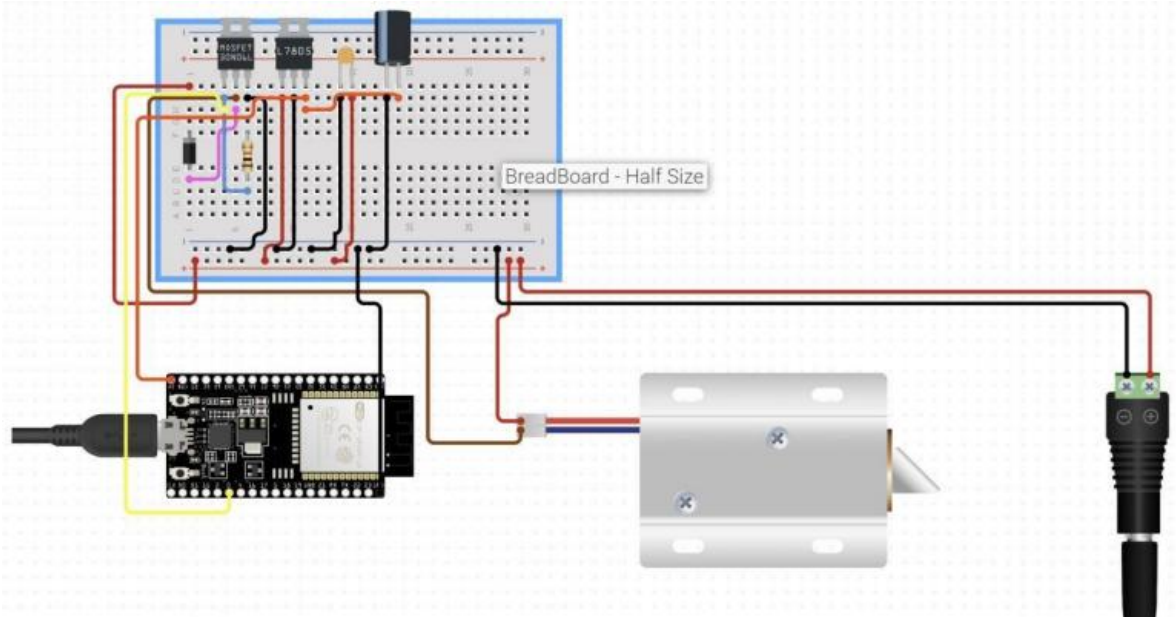


Figure 6-2: Camera-Based Face Recognition model

Face recognition offers a non-contact biometric authentication method that is convenient and user-friendly. The **ESP32-CAM**, with its onboard OV2640 camera and Wi-Fi connectivity, provides an affordable yet powerful module suitable for smart security systems.

6.2.1. Working Architecture

The ESP32-CAM performs face recognition through these stages:

1. **Face Detection**
   The live camera feed is analyzed to detect the presence of a human face.
2. **Face Encoding**
   Key facial features are extracted:
   - Eye distance
   - Nose shape
   - Jawline geometry
   - Cheekbone structure
3. **Template Matching**
   The captured face is compared to trained templates stored in memory.
4. **Decision Rule**
   - If match found → authentication success
   - If mismatch → trigger alarm or block access

6.2.2. Benefits

- Non-contact authentication
- Works even in low-light (with IR LED)
- Allows multi-user enrollment
- Adds an additional layer of security
- Difficult for an intruder to spoof without deepfake-level tools

6.2.3. ESP32-CAM in Locker Systems

The ESP32-CAM can be mounted on the locker door to capture face images at eye level. Its advantages for lockers include:

- **Live monitoring:** Sends real-time images to server if tampering detected
- **Remote verification:** Owners can view the live stream
- **Auto-unlock upon face authentication**

Integrating ESP32-CAM ensures the locker is protected against identity theft or stolen credentials.

6.3. Combined Multi-Factor Authentication (MFA) Model

Integrating fingerprint and face recognition alongside OTP creates a **multi-layer, tamper-resistant, highly secure locker system**.

6.3.1. Proposed Authentication Flow

A secure, hierarchical three-level model:

1. **Level 1 – Face Recognition (ESP32-CAM)**
   - Verifies the authorized user's identity visually
   - Prevents strangers from attempting further steps

2. **Level 2 – Fingerprint Scan**
    - Confirms the identity using a unique biometric signature
    - Eliminates possibility of stolen face image spoofing
3. **Level 3 – OTP Verification (IoT + Gmail)**
    - Dynamic code ensures real-time authorization
    - Even if biometrics are spoofed (extremely rare), OTP protects access

Only after **all three verifications succeed** does the relay-driven solenoid lock activate.

6.3.2. Security Strength

Security Layer	Attack Resistance	Notes
Face Recognition	High	Hard to spoof without high-tech methods
Fingerprint	Very High	Unique biometrics prevent duplication
OTP	Extremely High	Dynamic codes prevent reuse

When combined, the system becomes **virtually impenetrable** without the user's direct presence.

6.4. Advantages of Adding Biometric & Camera Systems

- **Improved Security:** Multi-level authentication prevents brute-force, OTP interception, and identity theft.
- **User Convenience:** No need to remember passwords; biometrics are innate.
- **Audit Trail:** ESP32-CAM images can be stored for historical verification.
- **Scalability:** Supports multiple users with different biometric profiles.
- **Real-Time Monitoring:** Camera module offers remote visibility in case of tampering.

6.5. Challenges & Considerations

While powerful, biometric and camera-based systems bring additional factors:

Technical Challenges

- Need good lighting for face recognition
- Higher processing load
- Requires stable Wi-Fi

Security Considerations

- Biometric templates must be stored securely
- Camera feed must be encrypted

Cost Factors

- Adds additional hardware cost
- Slightly higher power consumption

Despite these considerations, the benefits significantly outweigh the challenges for high-security applications.

6.6. Summary

By integrating **fingerprint authentication** and **ESP32-CAM face recognition** into the existing OTP-based locker, the system evolves into an advanced **multi-factor authentication (MFA)** platform. This modernizes the traditional locker design, making it comparable to enterprise-grade security solutions used in banks, research labs, and high-value storage facilities.

Biometric and visual identification mechanisms strengthen the locker against unauthorized access, identity theft, and physical tampering, while the OTP layer provides dynamic, real-time verification.

This extension transforms the smart locker into a **future-ready, highly secure, and technologically advanced access control system**.

INDIVIDUAL CONTRIBUTIONS

The project titled "OTP-Based Smart Locker System Using Embedded System Design Principles" was completed collaboratively by the team. Each member contributed significantly to both the technical development and documentation of the system. The detailed division of work is as follows:

1. Ishani Dey (Reg. No: 220906486)

Primary Contributions:

- Report Writing (Lead Author):
    - Prepared the complete project report, including
        - Abstract
        - Introduction
        - Literature Review
        - Methodology
        - Results & Discussion
        - Conclusion & Future Scope
    - Structured all chapters and ensured academic formatting.
    - Created lists of tables, figures, abbreviations, and overall index.
- Hardware Development:
    - Assembled the hardware model with Arduino, keypad, LCD, relay module, and solenoid lock.
    - Participated in circuit wiring, debugging, and testing.
    - Verified relay actuation, solenoid response, and OTP workflow.
    - Assisted in IoT Gmail OTP setup and microcontroller interfacing.

2. Harikishantini K (Reg. No: 220906286)

Primary Contributions:

- Presentation (Lead Presenter):
    - Prepared the PPT for interim and final evaluations.
    - Designed slides covering:
        - Block diagrams
        - Circuit explanations
        - FSM flow
        - Hardware results
        - Future enhancements
    - Delivered the presentation to faculty and evaluators.
- Hardware Development:
    - Worked on assembling and organizing the physical prototype.
    - Helped with component mounting, wiring management, and LCD/keypad placement.
    - Assisted in relay and solenoid testing, OTP verification trials, and error handling.
    - Contributed to debugging issues during the hardware integration phase.

3. Naveed Ismail (Reg. No: 220906644)

Primary Contributions:

- Presentation (Co-Presenter):
    - Helped structure and refine the PPT slides.
    - Explained working principles, OTP mechanism, and testing results during evaluation.
    - Managed Q&A session along with the team.
- Hardware Development:
    - Contributed to building and testing the hardware circuit.
    - Helped in keypad scanning tests, relay switching validation, and solenoid lock operation.
    - Supported the team in troubleshooting connection faults and improving stability.
    - Ensured proper power supply management and safe handling during testing.

BILLING OF ITEMS

Item Nomenclature	Estimated Cost	Notes
Arduino Uno R3	~ ₹ 335	Core controller board.
4×4 Keypad	~ ₹ 39	For user input / entering OTP.
16×2 LCD Display Module	~ ₹ 81	Displaying messages/status.
12 V Solenoid Lock	~ ₹ 498	The physical locking mechanism.
5 V 1-Channel Relay Module	~ ₹ 39	To switch the solenoid lock via Arduino.
Miscellaneous (wires, breadboard/PCB, power supply, backup battery)	~ ₹ 200	Estimated placeholder for small parts and power module.
Total Estimated Hardware Cost	~ **₹ 1,192**	Sum of above items.

PROJECT DETAILS	
Details of Student 1	
Name	Harikishantini K
Registration Number	220906286
Mail ID	harikishanthini.mitmpl2022@learner.manipal.edu
Mobile	8610804296
Branch	Electrical & Electronics
Details of Student 2	
Name	Ishani Dey
Registration Number	220906486
Mail ID	ishani1.mitmpl2022@learner.manipal.edu
Mobile	7386468969
Branch	Electrical & Electronics
Details of Student 3	
Name	Naveed Ismail
Registration Number	220906644
Mail ID	naveed.mitmpl2022@learner.manipal.edu
Mobile	9994499580
Branch	Electrical & Electronics
Project Title	*(Fill your project title here)*
Guide Details	
Name of Guide	Vijay S R
Designation & Department	Asst. Professor – Senior Dept. of Electronics & Communication Engineering Manipal Institute of Technology, MAHE
Mail ID	sr.vijay@manipal.edu
Signatures	
Signature of Student 1	
Signature of Student 2	
Signature of Student 3	
Signature of Guide	
Date	

OTP based smart locker using Embedded System Principles

→ Harikishantiniik

→ Ishani Dey

→ Naveed Ismail

Embedded System

Minor Project

(weekly report)

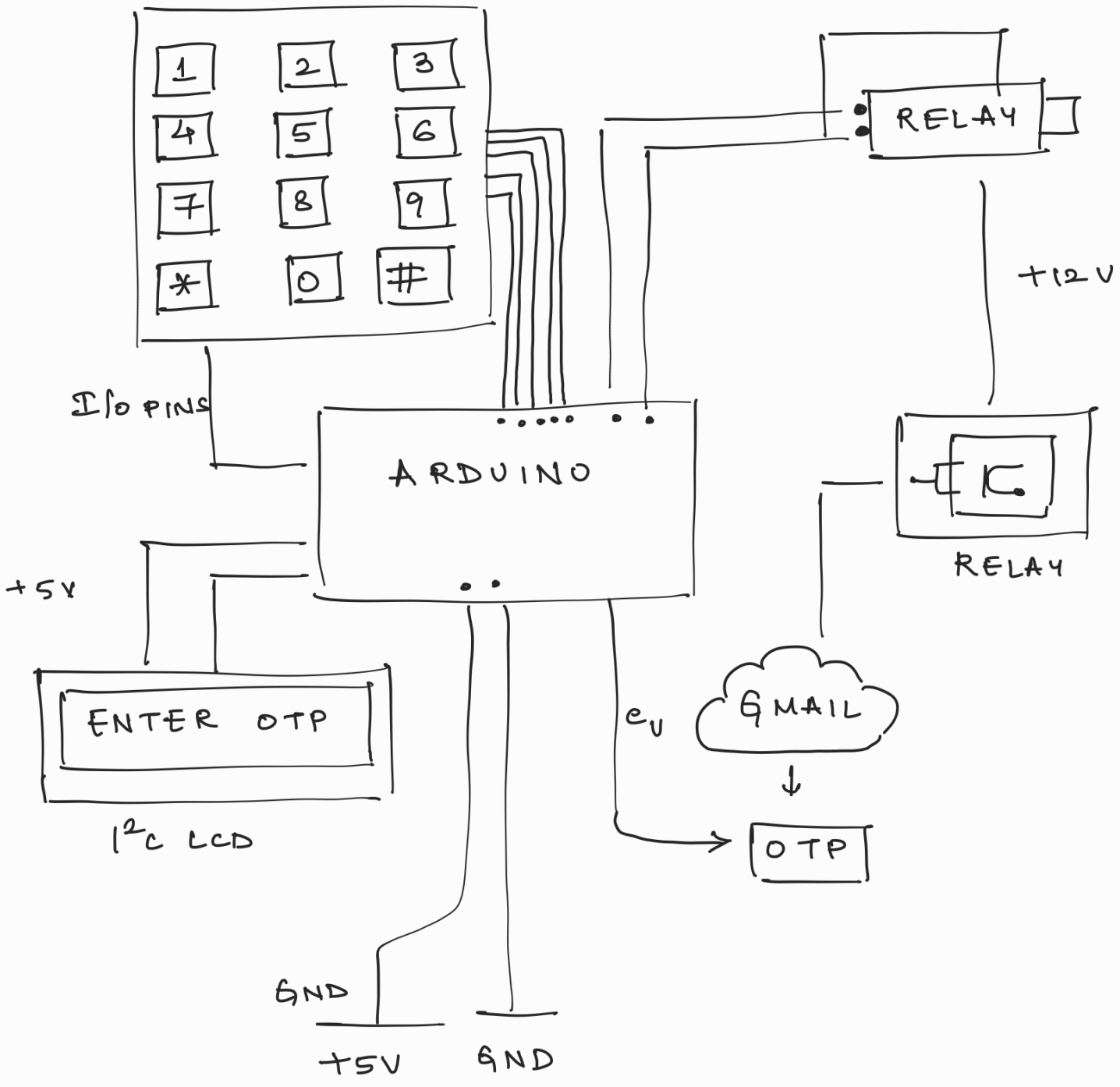
Week 1	Initial Topic Exploration
Week 2	Researching existing locking mechanism
Week 3	Attempt with fingerprint authentication
Week 4	Troubleshooting and Alternative
Week 5	Face recognition
Week 6	Concept shift to OTP based locker
Week 7	Literature review & requirement
Week 8	Preliminary circuit design
Week 9	Code development in simulation
Week 10	Simulation Validation
Week 11	Transition to Hardware stage
Week 12	Initial Breadboard setup
Week 13	Relay and Solenoid lock Testing
Week 14	Power Supply stabilization
Week 15	Hardware validation and debugging
Week 16	Introduction to IoT integration
Week 17	SMTP and Gmail OTP Experimentation
Week 18	Integration of email OTP in code
Week 19	Testing End to End workflow
Week 20	Security and reliability validation
Week 21	Final code Optimization
Week 22	Functional testing and debugging
Week 23	Security testing
Week 24	Performance testing
Week 25	Comparative analysis
Week 26	Report drafting begin
Week 27	Results and Documentation
Week 28	Final editing and formatting
Week 29	Presentation Preparation
Week 30	Final review and Submission

Week 1 Initial topic exploration	We began discussing ideas for our minor project under embedded system design. We explored different domains such as home automation, IoT devices and smart security. After multiple brainstorming sessions, we found the concept of a smart locker both practical and innovative. The idea was to merge hardware control with real time authentication
Week 2 Researching existing locking mechanisms	Our team studied various locking technologies – mechanical locks, RFID, PIN based digital locks and biometric systems. We noticed that most digital lockers used static passwords, which could easily be guessed or reused. This realization guided us towards creating a system that generates a dynamic one time password (OTP) for every use.

Week 3 Attempt with fingerprint authentication	We began experimenting with a fingerprint sensor (AS608 type). Though the initial setup was successful in simulation, the sensor failed to respond consistently on hardware due to faulty module calibration. We learned about serial communication protocols and biometric data enrollment during this stage
Week 4 Troubleshooting and alternative exploration	Several days were spent troubleshooting the fingerprint module, including verifying baud rates and wiring. Despite our efforts, the device gave inconsistent readings. This setback motivated us to explore face-recognition based authentication using the ESP32-CAM board
Week 5 Face recognition attempt	We procured an ESP32-CAM and tested simple faced-detection code. The idea was promising, but the camera module consumed high current and required constant wi-fi streaming which caused instability. Moreover, processing faces in real time exceeded the Arduino's capacity. We decided to mark this idea as future scope and look for simpler yet secure alternative.

Week 6 Concept Shift to OTP based locker	We finalized the shift from biometric recognition to an OTP based system. The team agreed this method would provide dynamic authentication while keeping the hardware affordable. The plan was to use a keypad, LCD, relay and solenoid lock controlled by an Arduino uno.
Week 7 Literature review & requirement Gathering	This week, we conducted a detailed literature review of existing smart locks. We identified gaps such as lack of real time OTP validation and expensive modules. Our requirement list was created : Arduino uno, keypad, I^2C LCD, relay, solenoid lock and power supplies.
Week 8 Preliminary Circuit Design	We designed the circuit schematic in Tinkercad. The system included a 4 X 4 keypad for user input, a relay controlled by an NPN transistor, and a DC motor representing the solenoid. We verified current limits and safety through virtual connections

Week 9 Code development in Simulation	The initial code was written in Arduino IDE. It allowed me to set and verify a 4-digit OTP. We implemented basic keypad scanning, LCD display functions and relay activation logic. Debugging in Tinkercad helped us validate the OTP workflow without physical components.
Week 10 Simulation Validation	The virtual model ran successfully. When the correct OTP was entered, the DC motor rotated, simulating and unlocking state. Incorrect OTP attempts triggered warning message and reduced retries. This week concluded the simulation phase with confidence to move towards real hardware


1
2
3
4
5
6
7
8
9
*
0
#
I/O PINS
ARDUINO
RELAY
+12 V
IC
RELAY
+5V
ENTER OTP
I²C LCD
e_U
GMAIL
OTP
GND
+5V
GND

Week 11 Transition to Hardware stage	After validating our logic in Tinkercad, we began sourcing physical components: Arduino Uno, 4 x 3 keypad, I²C LCD, relay module, solenoid lock and jumper wires. The hardware assembly plan was sketched on paper, ensuring safe wiring between low-voltage and high current sections.
Week 12 Initial breadboard setup	We connected the keypad and LCD to the arduino on a breadboard. Power was supplied via USB, and outputs were verified using serial prints. Early issues included noisy connections and incorrect LCD addressing which we resolved by checking the I²C address using an I²C scanner sketch.
Week 13 Relay and Solenoid Lock Testing	This week focused on integrating the 5V relay module and 12V solenoid lock. The relay triggered correctly, but the solenoid drew higher current, causing voltage drop. We learned about flyback diodes and optocoupler isolation to prevent Arduino damage. After isolating circuits, the lock operated safely.

Week 14 Power supply stabilization	The system initially ran from a single 9V battery, which was insufficient for both control and actuator sections. We switched to a dual supply setup – Arduino powered via 9V, solenoid via 12V adapter. This separation improved stability ensuring the relay operated without reset.
Week 15 Hardware validation and Debugging	We finalized the wiring layout on a cardboard base. Each component's label and path were marked for clarity. All modules worked together - keypad input displayed on LCD, relay switched on correct OTP entry, and solenoid clicked reliably.
Week 16 Introduction to IoT integration	The team started working on IoT functionality to send OTP via email. We studied the SMTP (Simple Mail Transfer Protocol) and explored using Gmail servers with Arduino and Wi-fi modules. Since the Uno lacked Wi-fi, we planned to use an ESP board or PC-interfaced SMTP setup.

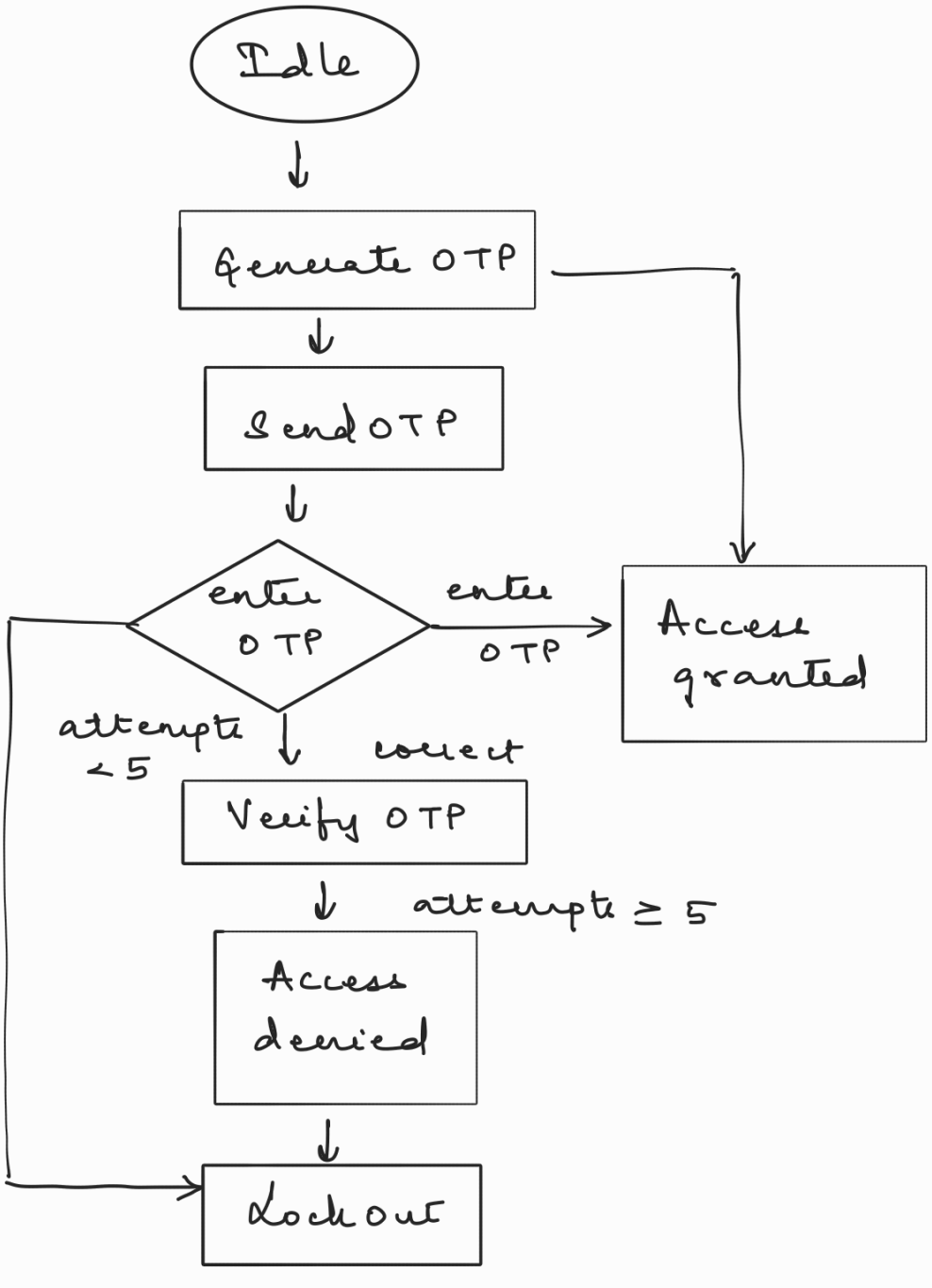
Week 17 SMTP and Gmail OTP Experimentation	Using arduino libraries, we configured an SMTP client for Gmail. We created a test Gmail account and set up an "App Password" for secure mail access. Initial attempts failed due to network authentication errors, but after adjusting port and SSL settings, OTP emails were successfully sent.
Week 18 Integration of Email OTP in Code	We merged the IoT code with the main locker program. Now, when a user pressed a key, the Arduino generated a random 4 digit OTP and emailed it to the registered address. This feature made our locker truly "smart" - combining embedded control with real-time internet communication.
Week 19 Testing End to End Workflow	We performed complete system tests. OTP generation → Email receipt → user input via keypad → verification → relay activation → solenoid unlock. The process worked flawlessly with a 1-3 second OTP delivery time. Incorrect OTPs displayed "Access Denied" and triggered retries.

Week 20

Security and Reliability Validation

To strengthen reliability, we implemented lockout logic after five incorrect attempts. Power cycle tests confirmed that each restart required a new OTP, preventing replay attacks. The LCD messages were updated for user clarity ("Access Denied", "Try again", "Locked out"). This week marked a major milestone

FSM flow chart



Week 21 Final Code Optimization	The core code was streamlined this week. We optimized delay timings, LCD message sequence, and retry logic for smoother transitions. A function was added to clear memory after OTP verification, ensuring no data persistance. The relay response time was tuned to match the solenoid's mechanical action.
Week 22 Functional testing and debugging	The system underwent multiple test cycles. Correct OTPs successfully triggered the solenoid, while incorrect ones produced "Access denied". We confirmed five retry limits before lockout. The LCD displayed real-time prompts like "Enter OTP" and "Access Denied". The hardware proved consistent during continuous operation
Week 23 Security testing	This week focused on testing security mechanisms. We simulated brute-force attempts and replay attacks. The system locked out after repeated failure and required a new OTP upon reset. Power resets also generated new OTPs, proving replay protection. Mechanical tampering of the solenoid showed an unwanted activation.

Week 24 Performance testing	We measured OTP email delivery time, solenoid actuation delay, and system recovery after lockout. The OTP arrived within 2–3 seconds, and the relay switched in under 100 ms. The solenoid's mechanical movement was strong and reliable. All readings were documented in a result table for the report
Week 25 comparitive Analysis	A comparision between the Tinkercad simulation and hardware implementation was performed. Both models behaved identically in logic, but hardware added to IoT and electromechanical elements. Simulation helped debug logic errors early, while the hardware stage validated real-world functionality.
Week 26 Report drafting begins	The team started compiling the final report. Sections on introduction, motivation and literature review were drafted based on earlier notes. Technical details like circuit diagrams, OTP flow, and FSM were prepared. Screenshot from Tinkercad and photos of the real setup were organized for the results

Week 27 Results and Documentation	All performance graphs, testing tables, and images were inserted into the report. The flow of results was explained clearly : OTP generation, email verification, access granted / denied, and system lockout. Discussion on observed outcomes and lessons learned were also included. The results matched expected theoritical behaviour.
Week 28 Final edits and formatting	This week was spent editing the report for consitency and clarity. Figures, tables and captions were properly numbered. The index, acknowledgement, and reference were finalized. Diagrams were redrawn, and all code sections were checked for syntax correctness
Week 29 Presentation preperation	Designed the Powerpoint presentation highlighting objectives, methodology working principle, and result. Each team members rehearsed their sections. The live demonstration plan was finalized - showing the locker display messages, email OTP receipt, and solenoid unlocking on correct input.

Week 30

Final review and submission

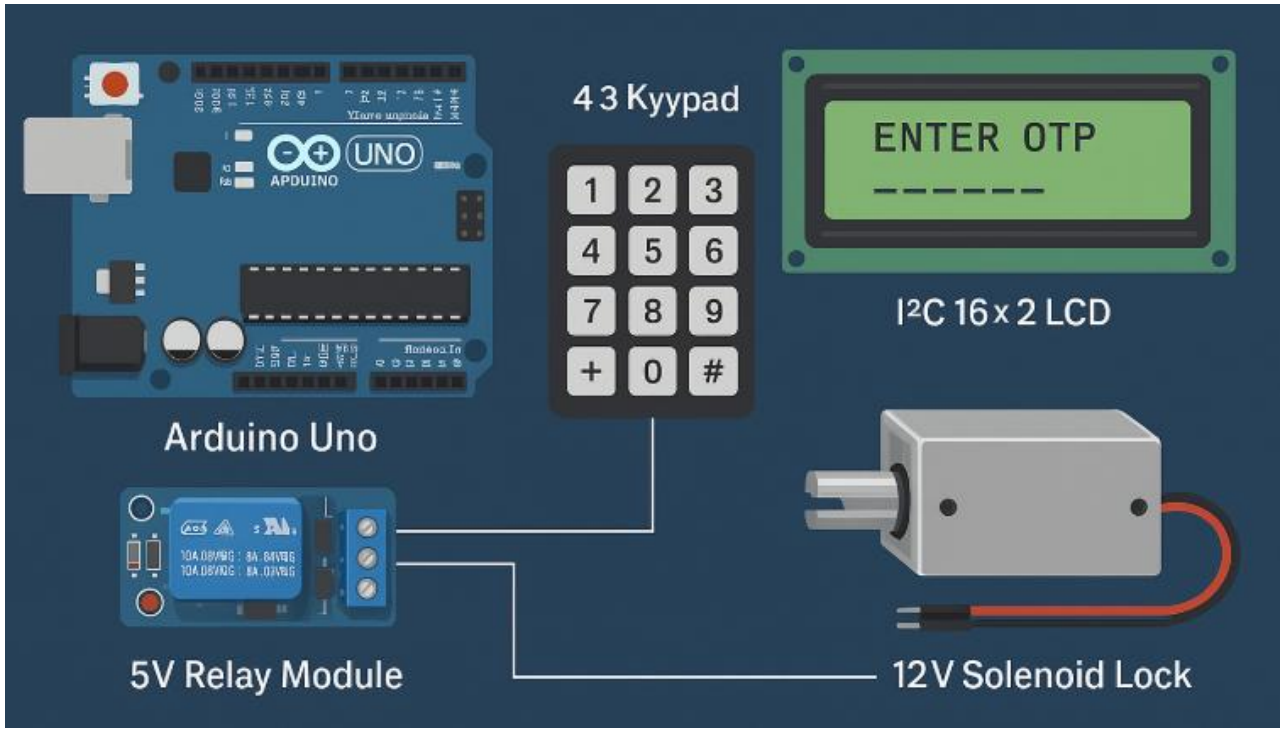
The final week concluded the project. We reviewed the entire workflow, ensuring every component functioned as expected. The presentation was delivered smoothly, and the hardware demonstration impressed evaluators. The project report, presentation slides and source code were officially submitted.

PROJECT REPORT – MINOR PROJECT

USING EMBEDDED SYSTEM DESIGN PRINCIPLES

Harikishantini K – 220906286
Ishani Dey – 220906486
Naveed Ismail – 220906644

INTRODUCTION

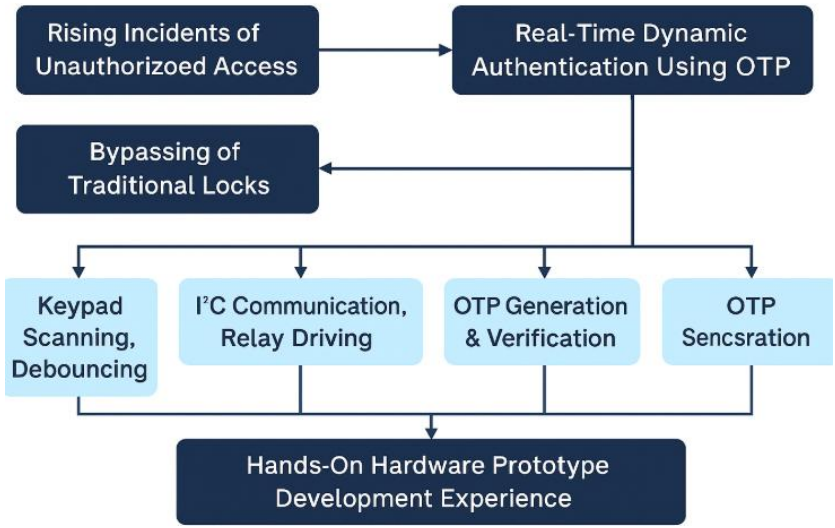


- Rapid shift from traditional mechanical locks to smart electromechanical systems.
- Static passwords/PINs are vulnerable to guessing, duplication, and shoulder surfing.
- OTP-based security provides **dynamic, single-use authentication**.
- The system integrates **Arduino Uno, keypad, I²C LCD, relay module, solenoid lock**, and **Gmail/SMTP-based OTP delivery**.
- Combines concepts from **Embedded Systems, IoT, Real-Time Systems, and FSM architecture**.

PRESENT-DAY SCENARIO

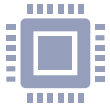
- Smart automation and IoT devices dominate modern security ecosystems.
- Existing digital locks often rely on **static passwords or RFID**, leading to vulnerabilities.
- Affordable OTP-based physical access systems are rare in the market.
- Increasing need for **low-cost, secure, and user-friendly lockers** for homes, hostels, labs, and offices.

MOTIVATION



- Rising incidents of unauthorized access and bypassing traditional locks.
- Need for a **robust security mechanism** using real-time, dynamic authentication.
- Opportunity to apply embedded system concepts practically:
- Keypad scanning, debouncing
- I²C communication
- Relay driving
- OTP generation & verification
- Hands-on hardware prototype development experience.

RELEVANCE OF THE WORK



Technical Relevance:

Demonstrates real-time input handling, peripheral interfacing, and relay-based actuation.

Implements OTP-based authentication uncommon in low-cost microcontroller projects.

IoT Relevance:

Uses Gmail/SMTP for secure OTP communication.

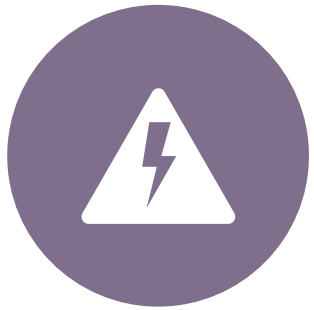
Academic Relevance:

Integrates principles from RTS, IoT, Embedded Systems, and FSM design.

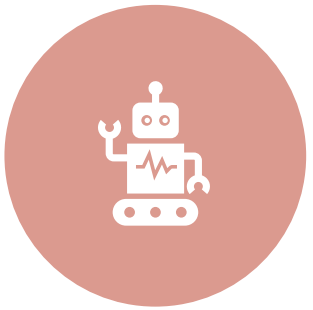
ENVIRONMENTAL & SOCIETAL IMPACT

LOW POWER CONSUMPTION AND MODULAR COMPONENTS REDUCE E-WASTE.

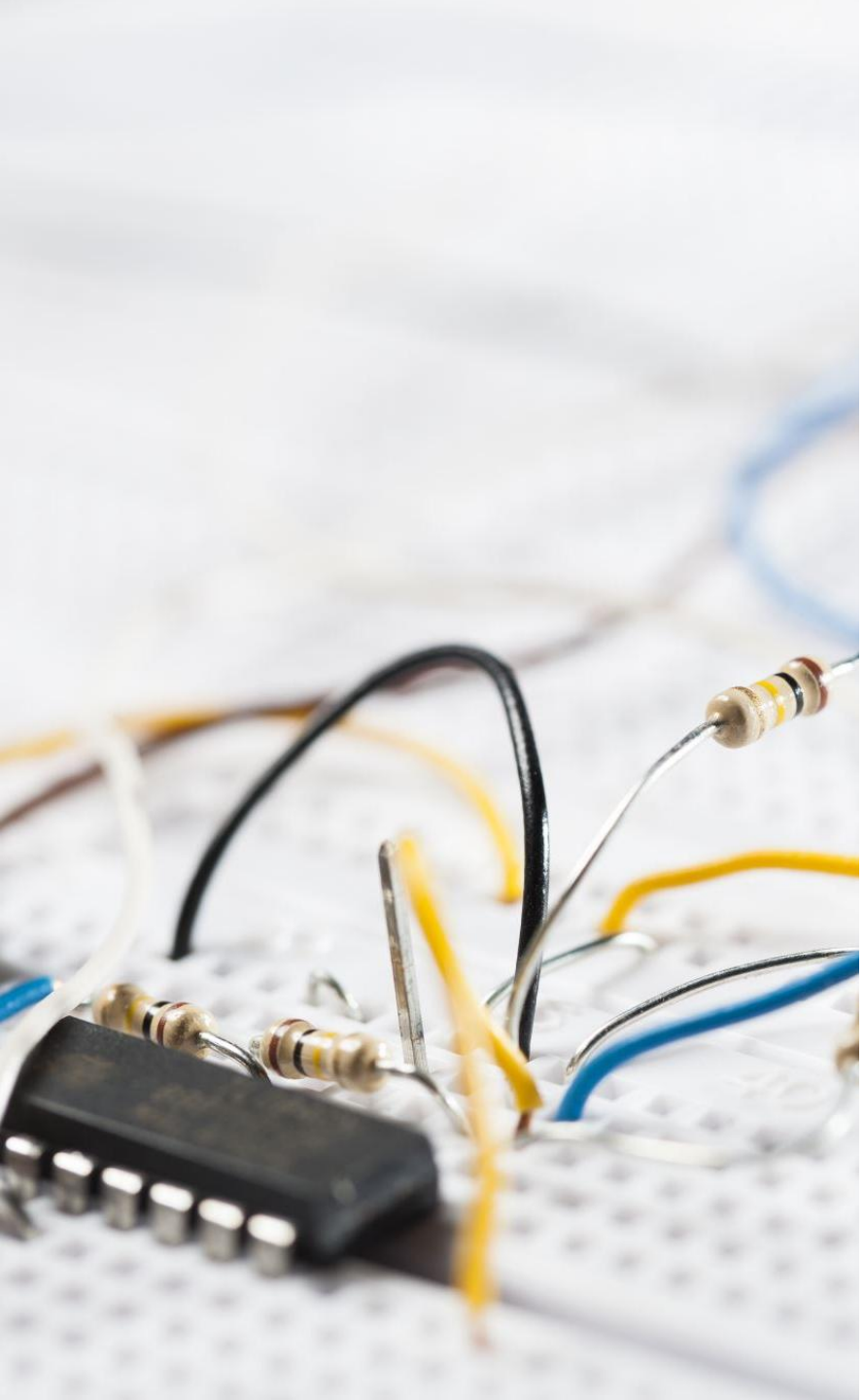
IMPROVES PERSONAL AND INSTITUTIONAL SECURITY.



RELAY-DRIVEN ACTUATION ENSURES ELECTRICAL SAFETY.



ECONOMICAL ALTERNATIVE TO COMMERCIAL IOT/BIOMETRIC SYSTEMS.



ETHICS IN ENGINEERING

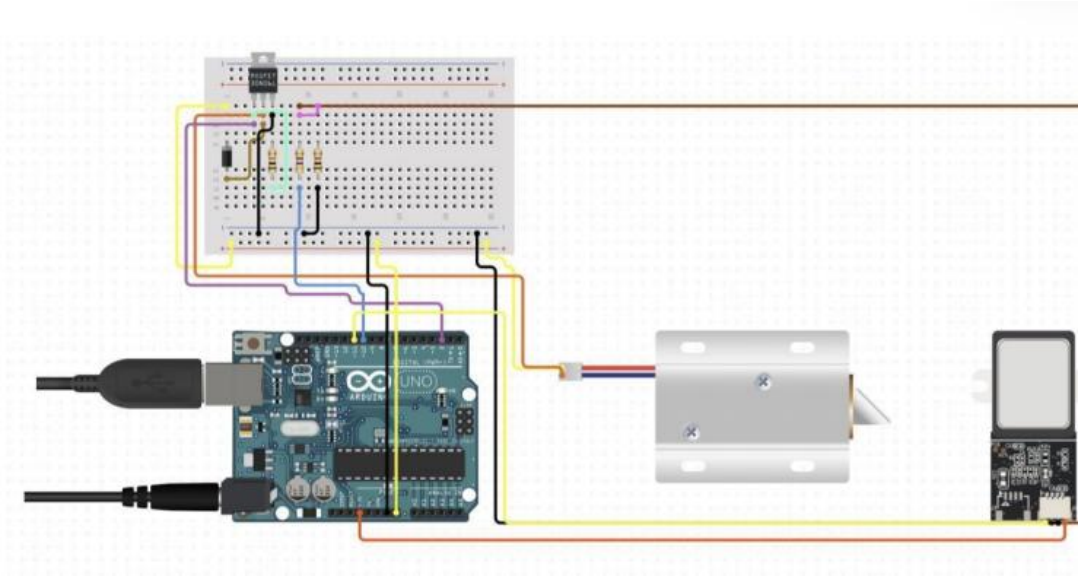
- OTPs are not stored long-term → ensures privacy.
- Secure relay isolation prevents electrical hazards.
- Failure handling using retry limits and lockout.
- Transparency through LCD prompts.
- No misuse of sensitive user data.

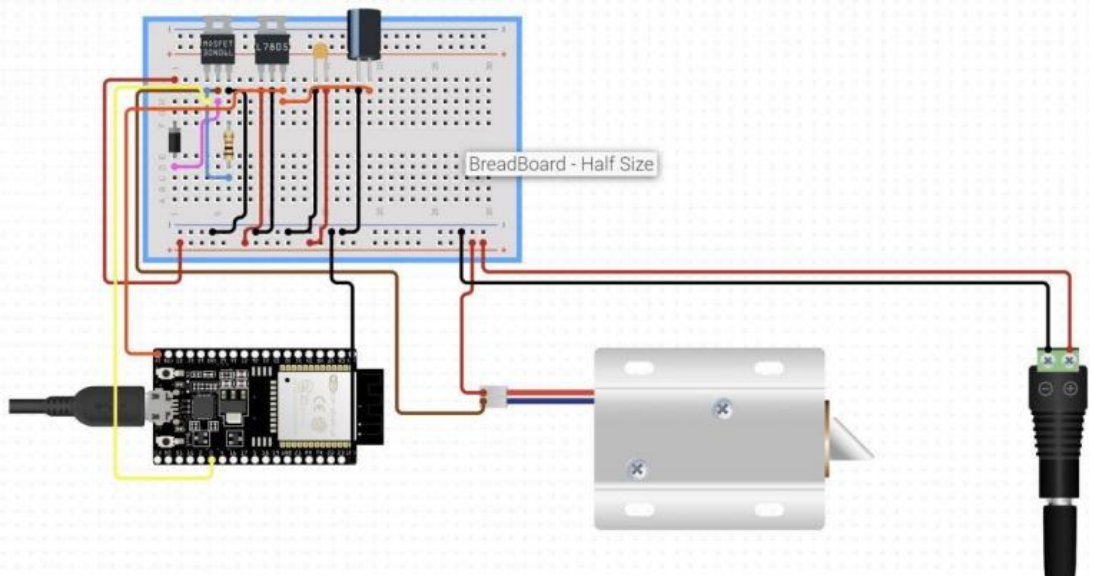
LITERATURE REVIEW: STATE OF THE ART

- IoT-enabled locks using Wi-Fi/Bluetooth: require apps and frequent connectivity.
- RFID & biometrics: risk of cloning/spoofing, higher cost.
- GSM-based OTP systems: SMS delay + message charges.
- Few low-cost OTP-based **physical lockers** available.
- Most academic works lack real-time constraints and OTP expiration logic.

LITERATURE GAPS IDENTIFIED

- Lack of affordable OTP-based hardware lockers.
- Over-dependence on smartphone apps.
- Limited focus on relay–solenoid actuation design.
- Few systems use finite state machines for safe control.
- Power optimization and mechanical reliability often overlooked.

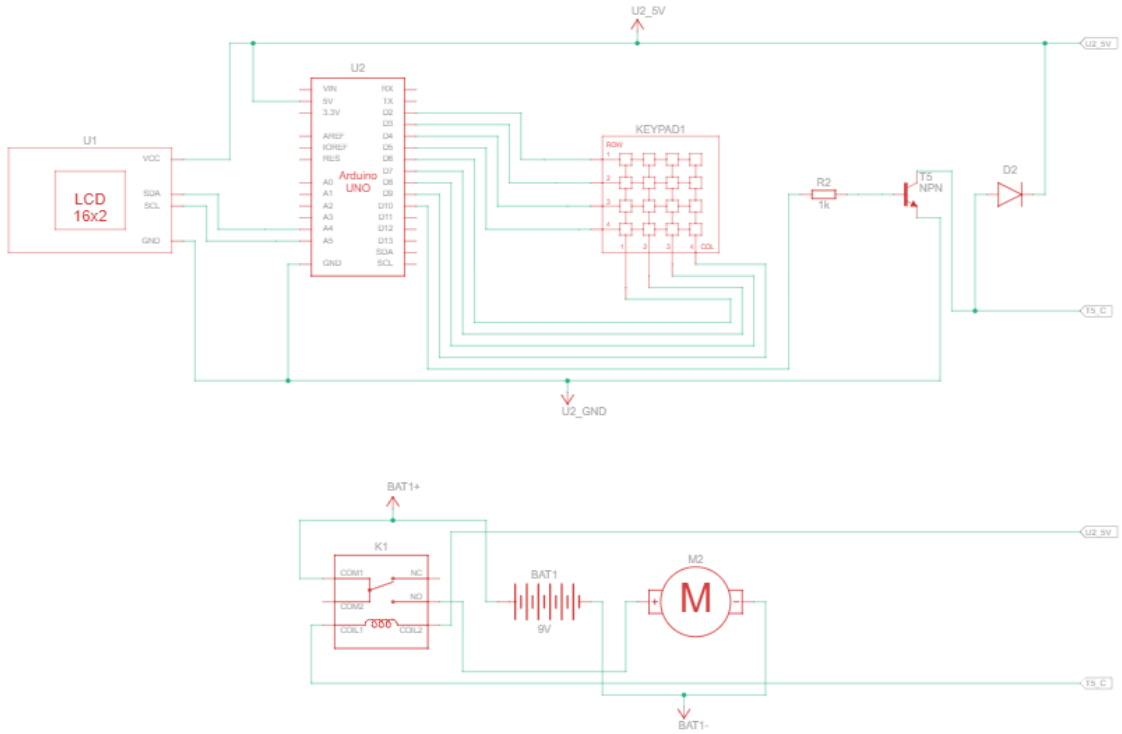




OBJECTIVES

- **Primary Objectives:**
- Develop secure OTP-based locker with Arduino.
- Use Gmail/SMTP for OTP delivery.
- Validate using Tinkercad + hardware prototype.
- **Secondary Objectives:**
- Implement FSM architecture.
- Introduce retries, lockout, and OTP invalidation.
- **Extended Objectives:**
- Scalability and low-cost implementation.
- Potential integration of biometrics/face recognition.

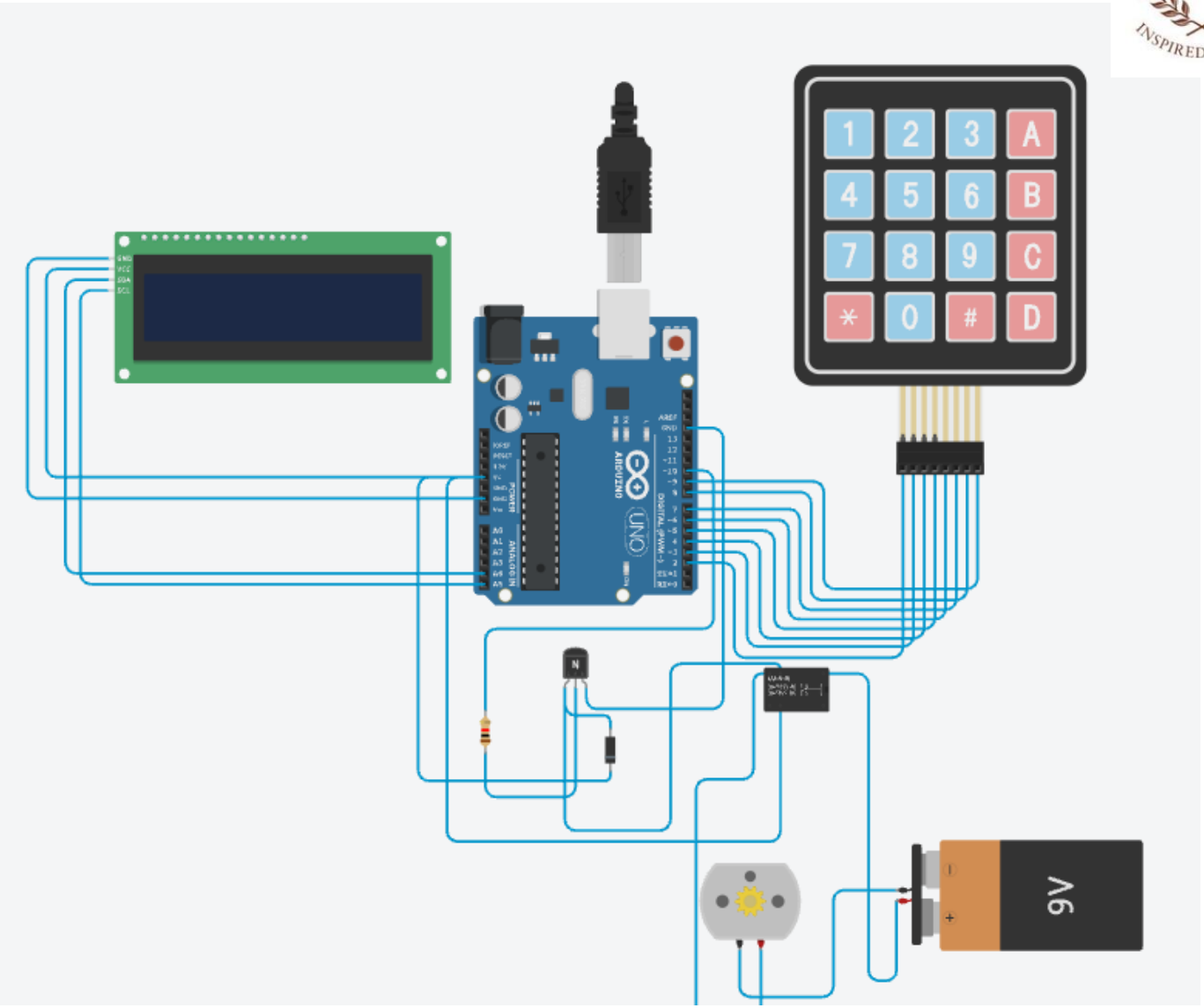
SYSTEM OVERVIEW



- User requests OTP → System generates OTP → Sent via email → User enters OTP → System verifies → Unlocks.
- Two models developed:
- **Tinkercad Simulation** – circuit logic validation
- **Hardware Prototype** – real solenoid + IoT OTP
- Clearly divided into **low-voltage logic** and **high-current actuator** sections.

COMPONENTS (SIMULATION MODEL)

- Arduino Uno
- 4×4 Keypad
- I²C 16×2 LCD
- LU5R bare relay with 2N2222 transistor
- 1N4007 flyback diode
- DC motor (acts as solenoid simulator)
- 9V battery
- Breadboard + wiring

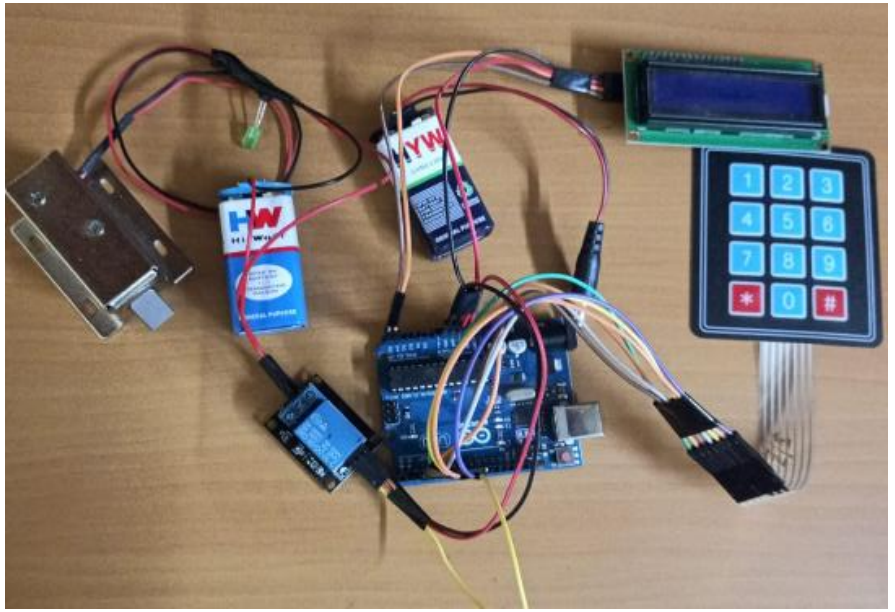


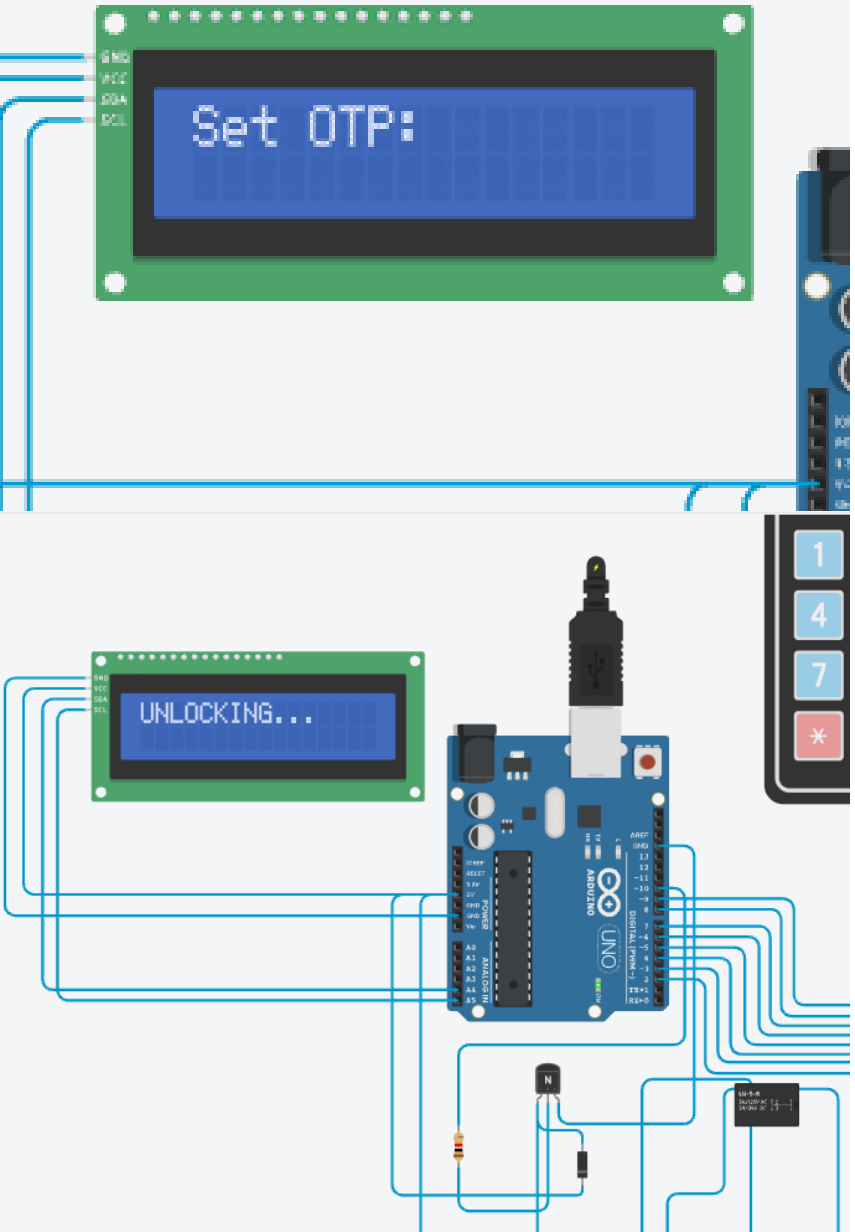
COMPONENTS (HARDWARE PROTOTYPE)

- Arduino Uno
- 4×3 Membrane keypad
- I²C 16×2 LCD
- 5V relay module with optocoupler
- 12V solenoid lock
- 9V supply for Arduino & 12V adapter for solenoid
- Jumper wires, enclosure

CIRCUIT DESIGN

- Low-voltage side: Arduino, keypad, LCD, relay signal pin.
- High-voltage side: solenoid lock + 12V supply.
- Relay provides electrical isolation.
- Flyback diode prevents back-EMF damage.
- Ensures safe solenoid actuation.





WORKING PRINCIPLE

- System initializes → displays prompt
- OTP generated → stored temporarily
- OTP transmitted to registered email
- User enters OTP
- OTP verified digit-by-digit
- Correct OTP → Relay energizes → Solenoid unlocks
- Incorrect OTP → retries decrement
- Excess failures → Lockout mode

OTP FLOW MECHANISM

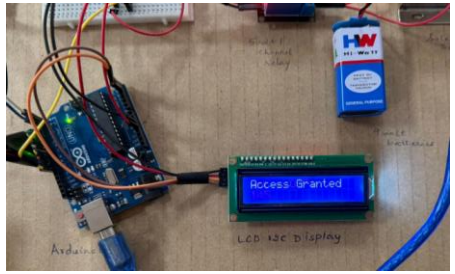
RANDOM 4–6 DIGIT OTP GENERATED.

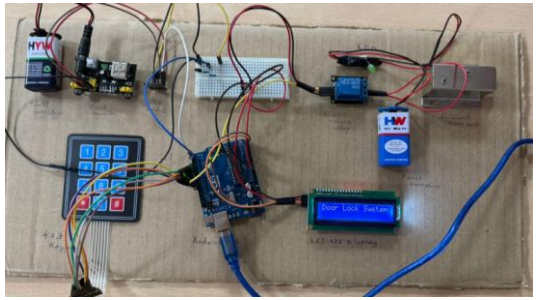
OTP EMAILED VIA GMAIL SMTP.

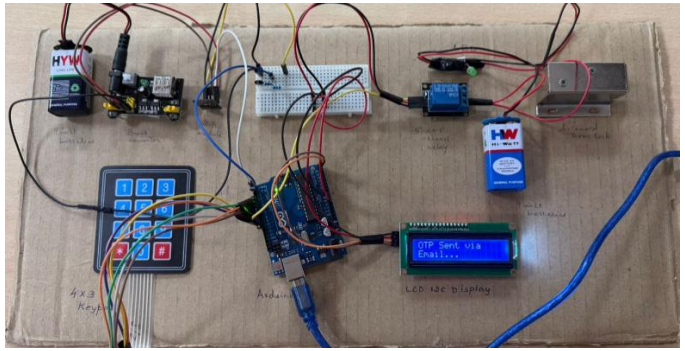
OTP VALID FOR SINGLE USE ONLY.

CLEARED FROM MEMORY AFTER USE.

RETRIES LIMITED TO PREVENT BRUTE-FORCE.

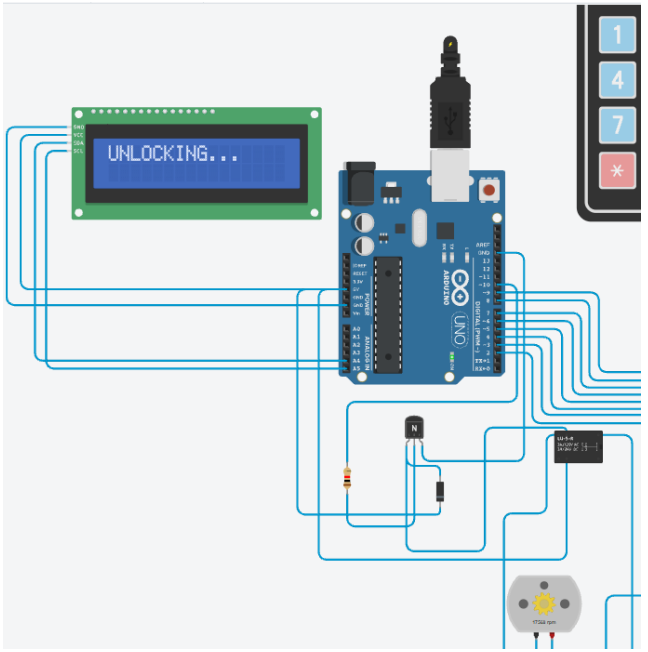


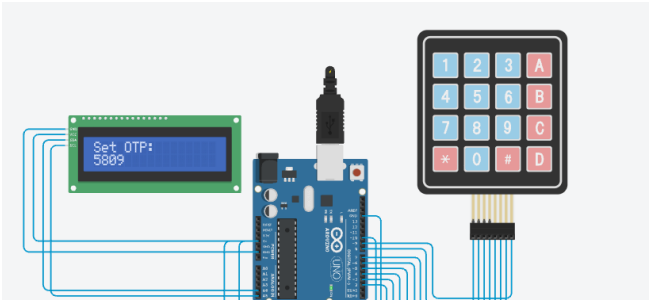




SOFTWARE DESIGN (FSM)

- Idle
- Generate OTP
- Send OTP
- Await Entry
- Verify OTP
- Access Granted
- Access Denied
- Lockout





KEY ALGORITHMS

01

Matrix keypad scanning + debouncing

02

I²C LCD communication

03

OTP randomization algorithm

04

Relay timing logic

05

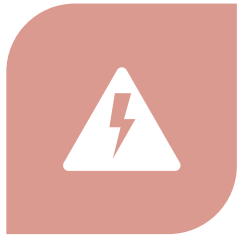
SMTP authentication for email delivery

TINKERCAD SIMULATION RESULTS

STABLE KEYPAD INPUT RECOGNITION

LCD MESSAGING ACCURATE

MOTOR ACTIVATES ONLY ON CORRECT OTP



DEBUGGING EASIER WITH NO ELECTRICAL RISK

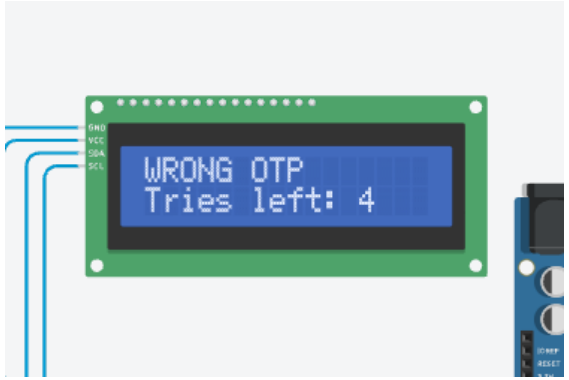
VERIFIED FULL LOGIC BEFORE HARDWARE ASSEMBLY

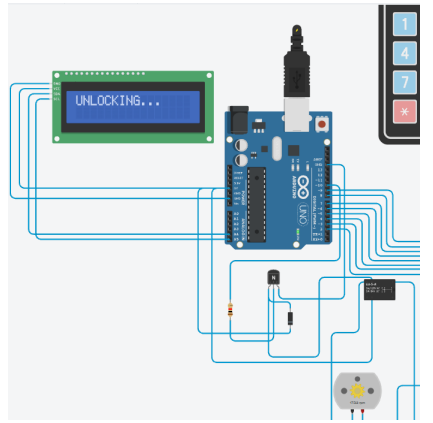
HARDWARE RESULTS

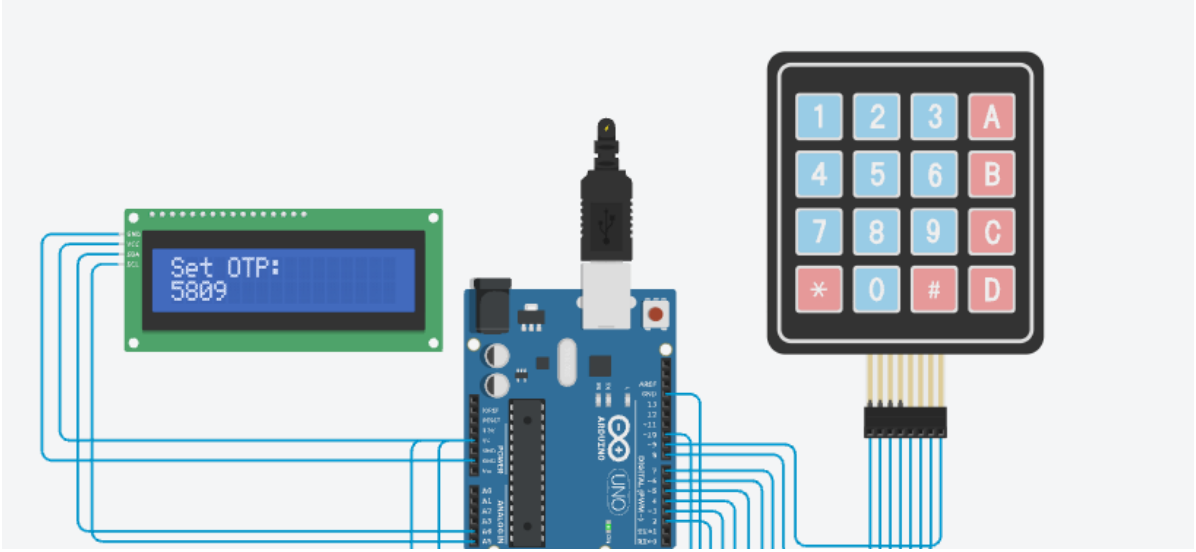
- Fast OTP delivery: **1–3 seconds**
- Solenoid lock shows strong mechanical pull
- Relay switching is stable and isolated
- LCD readable throughout
- Lockout after 5 wrong attempts

FUNCTIONAL TESTING

- OTP generation ✓
- OTP validation ✓
- Relay + solenoid actuation ✓
- LCD user feedback ✓
- Wrong OTP handling ✓
- System reset and retry logic ✓







SECURITY TESTING

- Replay attack prevention
- Single-use OTP only
- Brute-force protected via lockout
- Power reset forces new OTP
- Solenoid resists mechanical bypass

PERFORMANCE ANALYSIS

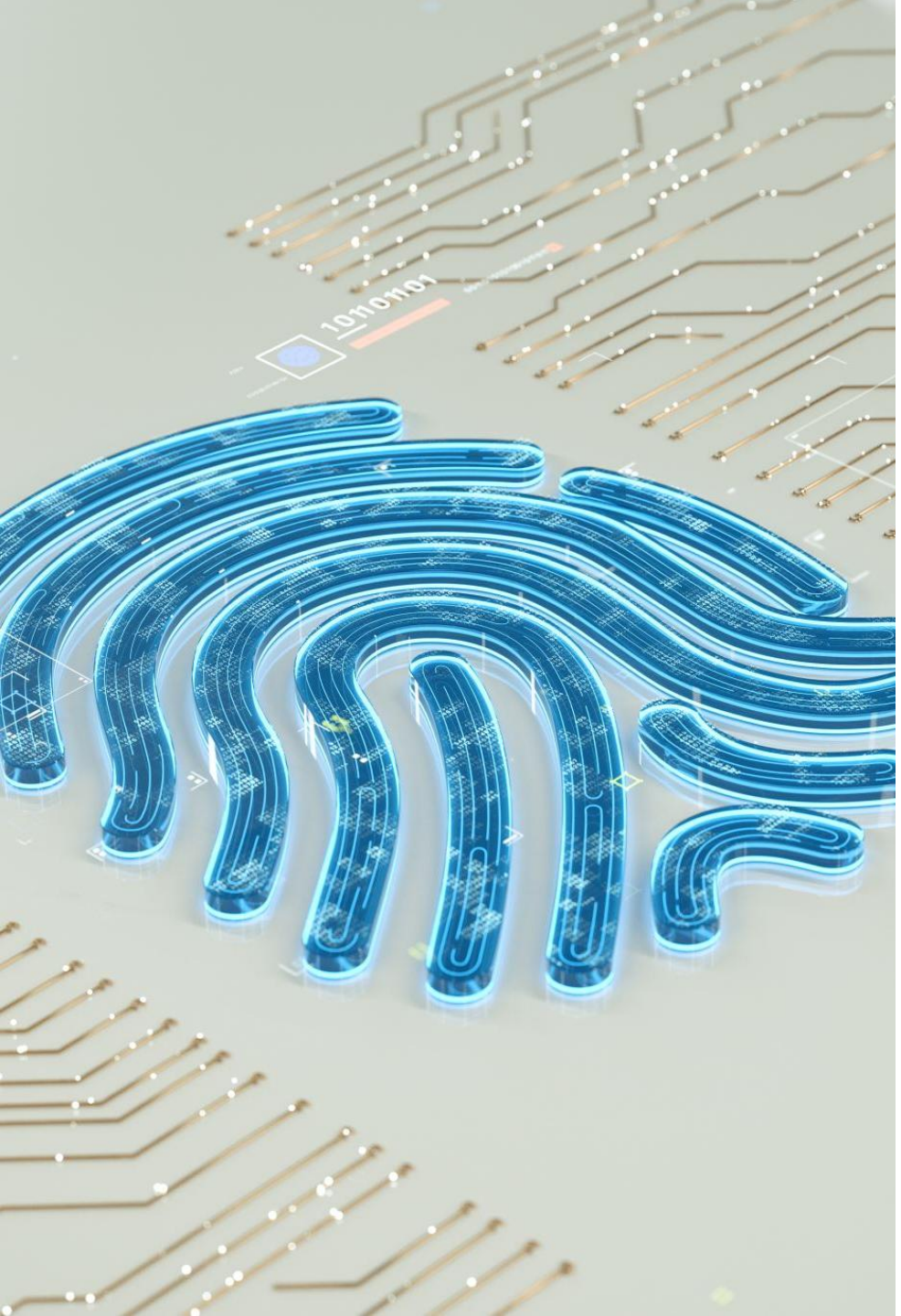
- OTP email latency: **~2 seconds**
- Relay response: **80 ms**
- Solenoid response: **300 ms**
- Keypad latency: **<50 ms**
- Current draw: **0.8–1.1 A** during solenoid actuation

SIMULATION VS HARDWARE

Simulation:	Motor-based actuation	No network latency	Ideal for logic testing	
Hardware:	Real solenoid lock	Actual email delivery delays	Electrical/EMF considerations	Better mechanical realism

CONCLUSION

- Successfully designed and implemented OTP-based smart locker.
- Combines embedded hardware with IoT-based authentication.
- Dual prototype approach ensured correctness and robustness.
- Secure, scalable, and affordable system.



FUTURE SCOPE

- Biometric authentication (Fingerprint, ESP32-CAM face recognition)
- Android/iOS app-based control
- Cloud dashboard with access logs
- Multi-user database
- Encryption + secure storage
- Tamper detection sensors

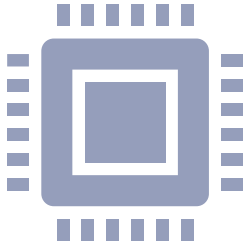
REAL-WORLD APPLICATIONS

- Residential lockers
- Hostel/PG cabinets
- College lab storage
- Office desk lockers
- Gym lockers
- Retail cash drawers
- Parcel pickup lockers
- Hotel room safes

LIMITATIONS

- Dependent on internet for OTP delivery
- Arduino UNO memory constraints
- No real-time clock module
- Single registered user
- High-current solenoid requires external power

CONTRIBUTIONS



Ishani Dey:
Report writing, hardware assembly, OTP workflow integration.

Harikishantini K:
Presentation, hardware development, relay/LCD/keypad integration.

Naveed Ismail:
Presentation, wiring, testing, power management, debugging.

THANK YOU