**🔐 Smart Locker System with Firebase Integration**

**📌 Project Overview**

This project is a **Smart Locker System** built using an **ESP32 microcontroller**. It uses **RFID technology** to allow or deny access, an **IR sensor** to detect locker usage, **LEDs** and a **buzzer** for feedback, and **Firebase Realtime Database** to **log access attempts and monitor locker usage** remotely.

**🧠 Key Features**

* 🔑 RFID-based access control (Card & Key Fob)
* 🚦 LED indication (Green for access granted, Red for access denied)
* 📢 Buzzer feedback for different events
* 🧍 IR sensor to check if the locker was physically opened
* ☁️ Firebase Realtime Database integration to store access logs
* 📲 Remote monitoring via Firebase Console

**🛠️ Components Used**

| **Component** | **Quantity** | **Description** |
| --- | --- | --- |
| ESP32 Dev Board | 1 | Main controller with Wi-Fi support |
| RFID Module (RC522) | 1 | Reads card/key UID for authentication |
| RFID Tags | 2 | One card, one key fob (pre-authorized) |
| IR Sensor | 1 | Detects if locker door was accessed |
| Green LED | 1 | Indicates access granted |
| Red LED | 1 | Indicates access denied or errors |
| Buzzer | 1 | Audible feedback for events |
| Jumper Wires | As needed | For connections |
| Breadboard | 1 | For prototyping |

**🧾 How It Works**

1. **Startup**: The ESP32 connects to Wi-Fi and Firebase.
2. **Scan RFID**: User scans a card/key fob on the RFID module.
3. **Authentication**:
   * If UID is recognized (authorized), access is granted:
     + Green LED glows
     + Buzzer sounds once
     + Access event is logged to Firebase
     + IR sensor waits for locker activity
     + If activity is detected, a success tone is played
     + If not, warning tones are played
   * If UID is unrecognized:
     + Red LED glows
     + Buzzer plays a denial tone
     + Access denied event is logged to Firebase
4. **Firebase Logging**: Each event (granted or denied) is stored with timestamp and UID.

**🔧 Firebase Setup**

1. Go to Firebase Console
2. Create a new project
3. Set up **Realtime Database**
   * Click **Realtime Database** > **Create Database**
   * Choose **Start in Test Mode**
4. In **Project Settings**, get:
   * **Database URL** (e.g., https://smart-lock-c33fb.firebaseio.com/)
   * **Database Secret** from **Service Accounts > Database Secrets**
5. Install Firebase Arduino library:
   * Open Arduino IDE → **Sketch > Include Library > Manage Libraries…**
   * Search and install Firebase ESP32

**⚙️ Code Highlights**

* **Wi-Fi & Firebase Setup**:

cpp

CopyEdit

WiFi.begin(ssid, password);

Firebase.begin(FIREBASE\_HOST, FIREBASE\_AUTH);

* **RFID UID Reading**:

cpp

CopyEdit

for (byte i = 0; i < rfid.uid.size; i++) {

cardUID += String(rfid.uid.uidByte[i], HEX);

}

* **Firebase Logging**:

cpp

CopyEdit

Firebase.setString(firebaseData, "/access\_log/" + cardUID, "Access Granted");

**📂 Firebase Data Structure**

json

CopyEdit

"access\_log": {

"93B4A929": "Access Granted - Card",

"F3365514": "Access Granted - Key",

"11223344": "Access Denied"

}

**🌐 Application Ideas**

1. **School Lockers** – Only students with registered RFID cards can open lockers.
2. **Parcel Lockers** – Automatically logs when a parcel is retrieved.
3. **Home/Office Security** – RFID access for drawers, cabinets, or doors.
4. **Gym Lockers** – Monitored and logged access for accountability.
5. **Tool Rooms** – Logs who accessed which compartment.

**🧠 Future Improvements**

* Add a mobile app using Flutter to read Firebase logs in real-time.
* Add a keypad or fingerprint sensor for multi-factor authentication.
* Trigger email/SMS notifications on denied access attempts.
* Use Firebase Firestore for advanced logging features.
* Add time-based restrictions using timestamps.

**📸 Optional UI with Firebase Hosting**

You can also build a simple HTML/CSS/JS dashboard and deploy it using Firebase Hosting to see who accessed the locker and when!