University of Illinois at Urbana-Champaign

Fall 2021

Smart Gate IoT - Intrance

Intrance Smart Gate: Technical Paper

Edward Passagi, passagi2
Mohamed A. Belakhoua, mab21

12/12/2021

Project Submission

Important Links

YouTube Showcase:

https://youtu.be/hzaqfwvpsiY

GitHub Repo:

https://github.com/edwardpassagi/SmartGateIOT

Project

Our project is called ***Intrance*** (Internet - entrance), which is a simple, RFID-based, smart module that is used to manage entrances (doors, gates, etc.). Intrance's backend system is powered by MongoDB (for storing user access data) and ExpressJS (for hosting real-time authentication endpoint).

Motivation

We wanted to create a solution for an automated gating system, which can be used for gated environments (office, parking lot, home doors, etc.) and runs on a small and flexible control module (ESP8266 NodeMCU) with basic internet capability. *Intrance* will enable its owner to manage door access *in real-time*, while also allowing faster identity authentication by leveraging RFID instead of other traditional methods (passcodes, PIN, etc.).

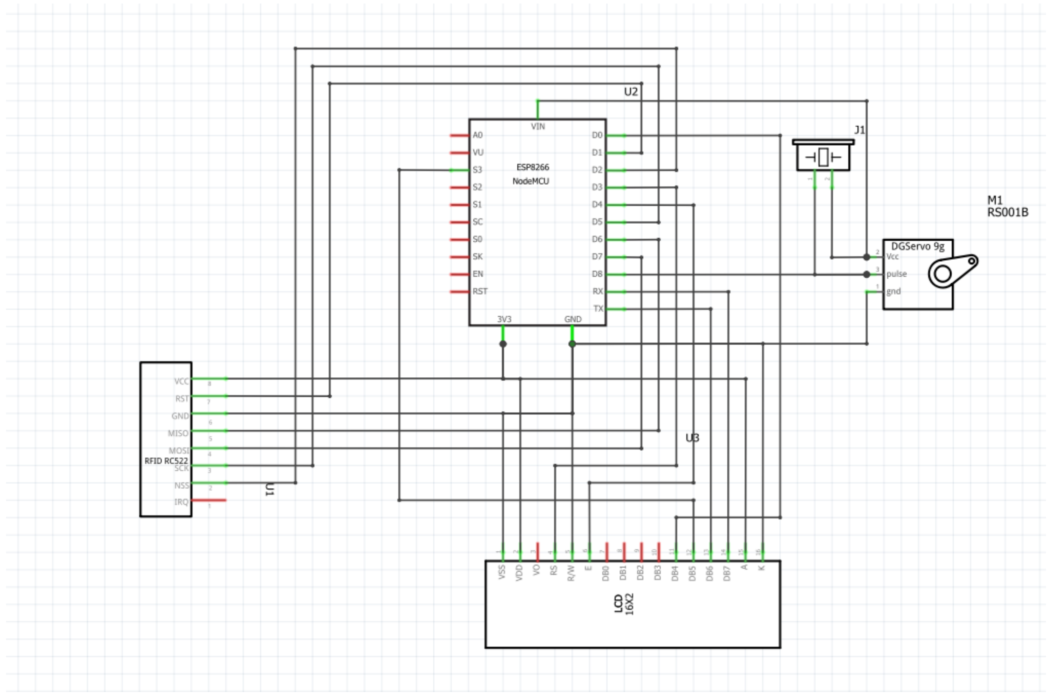
Intrance offers a simple approach for managing access in physical locations. Owners can easily expand their *Intrance* system without having to reconfigure user access for individual devices; the centralized backend system will allow easy configuration for managing new *Intrance* devices and user access. Additionally, we offer a huge logistical benefit -- there's no need to expand dedicated wirings between several different *Intrance* modules; as long as the area is covered with WiFi that connects to the internet. For security purposes, our RFID authentication method can also easily be exchanged for something more secure (like NFC), which will only require a small change on our backend system.

Technical Approach

Hardware

For our hardware, we used ESP8266 module as the “brain” for *Intrance*. Our ESP8266 module manages the RFID input (with MFRC-522 module) and output (16x2 LCD display, Servo motor). For our prototype, we simply used a servo motor to drive our miniature door. This, however, can easily be customized to move bigger motors/door systems (with proper power amplification). Our 16x2 LCD display shows useful instructions and the current state of the device (connecting to WiFi, ready to scan, access granted, access denied, WiFi signal lost, etc.).

Below is the circuit diagram of *Intrance*:



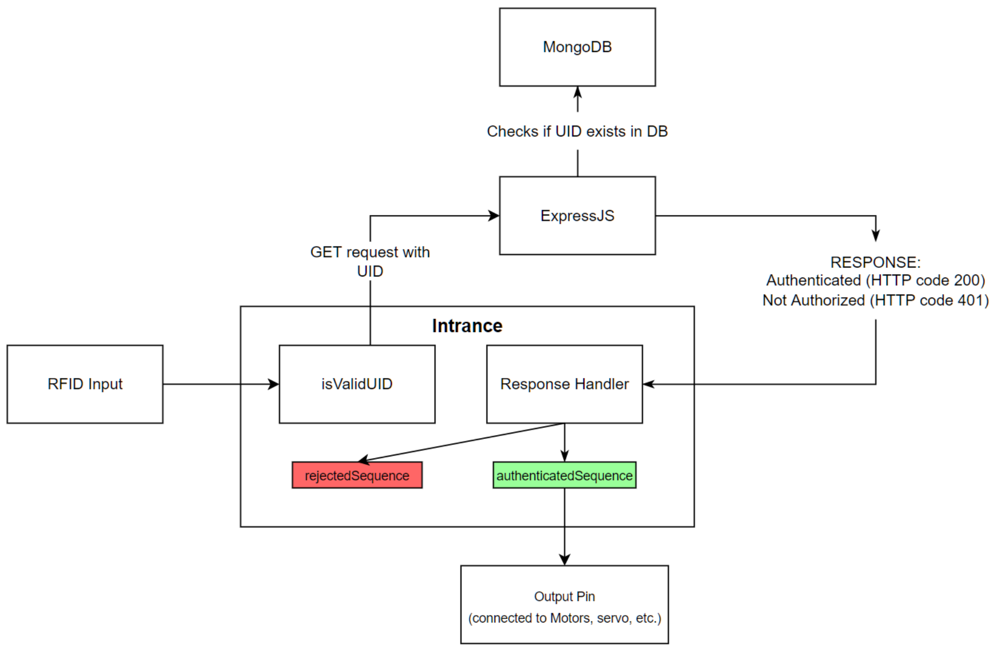
Software

For our software, we utilized MongoDB to store user data (name, linked RFID) and ExpressJS to provide API endpoints that our *Intrance* device can interact with. Our core ExpressJS logic (for authentication) is available here, and it is consumed by ESP8266 here.

Our MongoDB is hosted at MongoDB Atlas cloud service, while our ExpressJS is hosted at Heroku. In theory, we can use our prototype anywhere around the world as long as our device has access to the internet through WiFi.

Implementation Details

Here's a diagram of how *Intrance* processes an RFID input:



We first set up our MFRC522 module to extract the UID (Unified Information Devices) of our RFID. This UID is unique to each card, and it's how we map a user to any RFID card. We then create the MongoDB database to store our user data, alongside ExpressJS to allow *Intrance* to validate a given UID.

Our ExpressJS has 1 GET endpoint ("/validate") that takes in a UID as a parameter. Our current implementation simply queries the UID and sees if it exists within the database. If the UID exists in the database, we return a response with HTTP code 200 (success) and the name of the user embedded as JSON. If not, we respond with code 401 (unauthorized). Sending a request to an invalid endpoint will return a "400: Bad Request" error.

Intrance will then process the response accordingly; an authenticated response will trigger the *authenticatedSequence* (show the welcome message on LCD, trigger the servo) while an unauthorized response will trigger the *rejected sequence* (show access rejected message on LCD).

Our "output pin" is just a GPIO pin that we sent a signal to after successful authentication. It can be hooked to any device (beeper, servo, motor for gates) as long as they're appropriately powered.

We finally created a miniature setup to demonstrate how *Intrance* can open/close a door in a real-life setting.

Results

We are really excited to actually deliver what we envision *Intrance* to be. It has the full functionality for authentication and interacting with our miniature door. The centralized access control means that we can modify user access through MongoDB and *Intrance* will reflect it in real-time.

While it was very fulfilling to see how our project comes to fruition, it did not come without challenges along the development process.

First, we had to make a drastic switch from Arduino Uno to ESP8266 NodeMCU module, because we weren't aware that our ESP8266 comes as a NodeMCU (which has its own CPU). This makes it almost impossible for us to connect it with our Arduino Uno, and we didn't have enough time to reorder a standalone ESP8266 module.

Second, we initially wanted to also create a frontend for managing user access (as a website), but we ended up deciding to spend more time on our IoT development of *Intrance* because, well, it is far more exciting than developing a website. On a positive note, this decision enabled us to handle edge cases in *Intrance* (such as when the WiFi disconnects abruptly) and spend the extra time to deploy our ExpressJS code to Heroku instead of running it locally.

This project has enabled us (a CS and EE major) to learn a bit more about the technical fields of one another. *Intrance* gives a little taste of developing our own circuit, reading through datasheets, utilizing a breadboard, setting up a cloud database, creating API endpoints, and deploying it to the cloud. As a CS major, it's very refreshing to see how our code interacts with real life, while as an EE major, it's interesting to see how cloud services can enable *Intrance* to practically run anywhere around the world (with WiFi internet connection, of course.).

Group

Edward Passagi (passagi2), Mohamed Belakhoua (mab21).