Singapore Polytechnic

REPORT ON Secured End-to-End IoT for Door Entry System

Prepared and submitted by:

Class:	DCPE/FT/3A/25
Team Members	Student Id
Joseph Luther Tabaluyan	2033204
Brandon Chong Jun Wei	2032779
Ling Yi Hao Craigus	1908828

In partial fulfilment of the requirements for the module
ET0731: Internet of Things Security

Lecturer: Chee Wai Chan
Submission Date: 5th February 2023

Overview

Based on the topic given, we are planning to implement a door entry system for a house owner. The features used would ensure high security in the physical layer as it would be very difficult for threat actors to break and enter into the household.

First of all, residents would have to log into a web server with their username and password and request for an otp based password. Subsequently, the user will then input the otp based password into the web server to unlock the door. Additionally, there will be a camera on the door which is activated by the microcontrollers to automatically capture a picture when the otp based password input is wrong. This can also act as a physical deterrence. The photo will be sent to the home owner's phone via telegram bot.

Lastly, when the user decides to leave, he would have to press a button inside the booth. That will give him 15 seconds to exit before the door will be locked.

Why we decide to implement this project

1. To provide home owners a secure and safe solution to control access into their household
2. Remove the use of keys

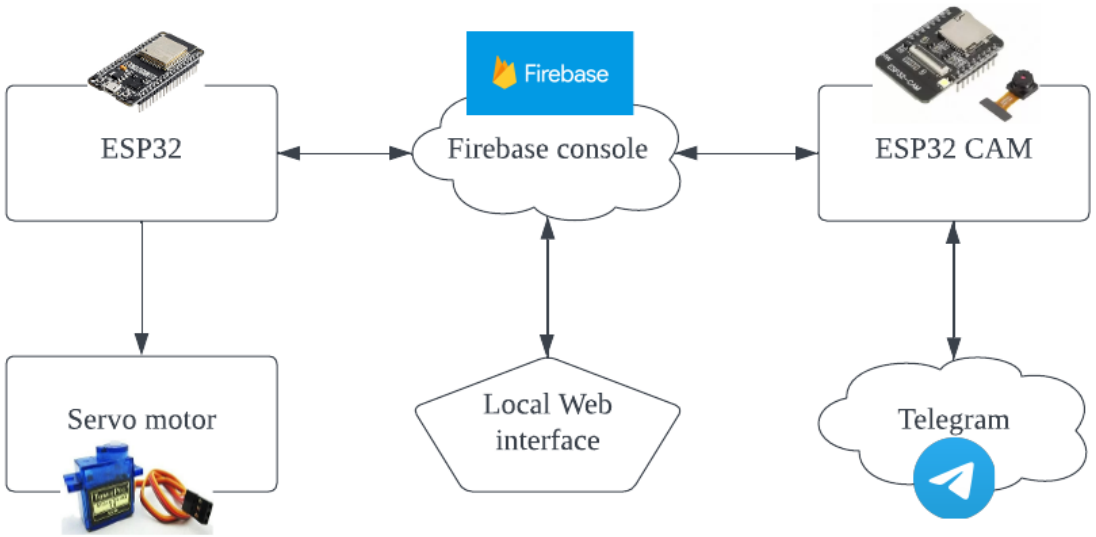
Goals

- Collect data using Firebase (database)
- OTP Based Password using random number generator]
- Complete the camera function using the ESP32 Cam
- Simulate a door lock using a servo motor
- Create a secured end-to-end IoT for door entry system

Hardware & Software required

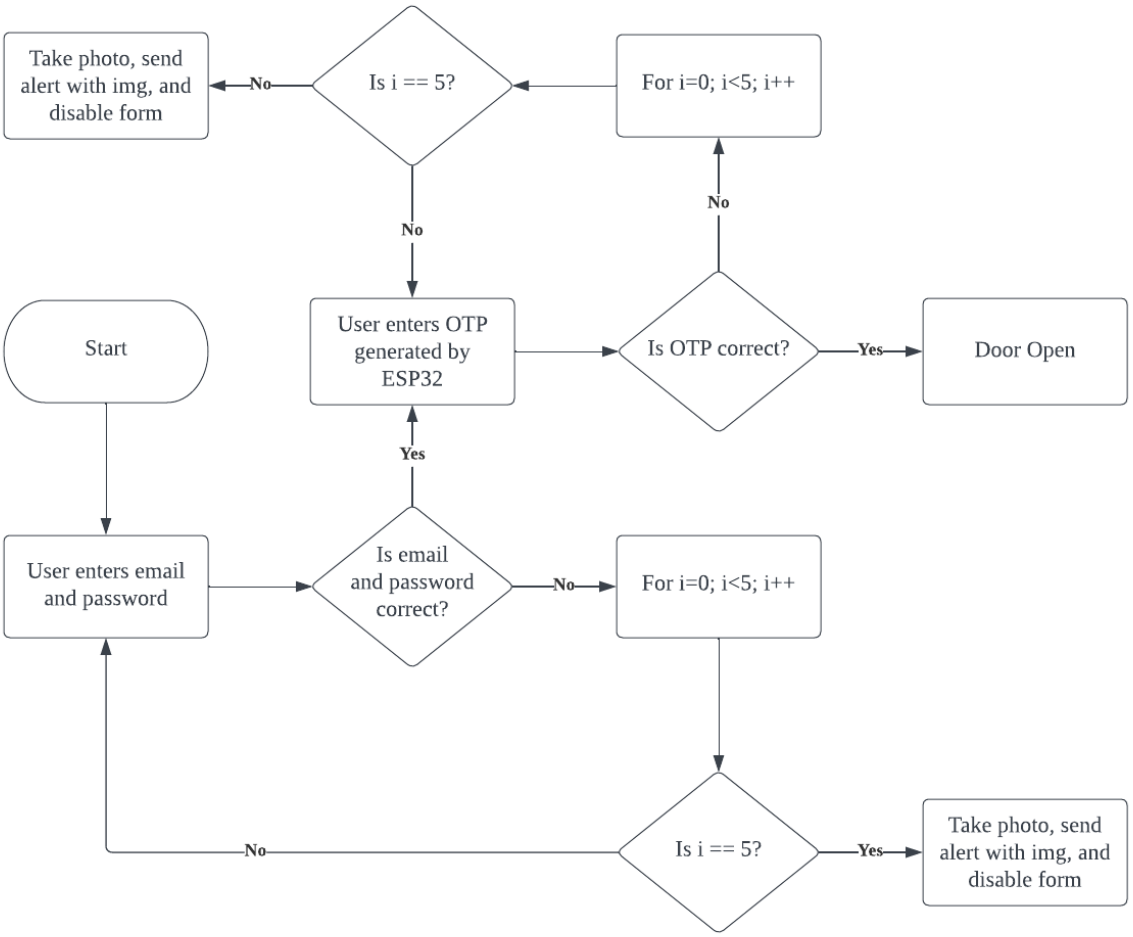
1. ESP32 Camera
2. ESP32
3. Micro Servo Motor SG90
4. Firebase
5. Telegram bot
6. Arduino Software
7. Website

Flow charts and component connections



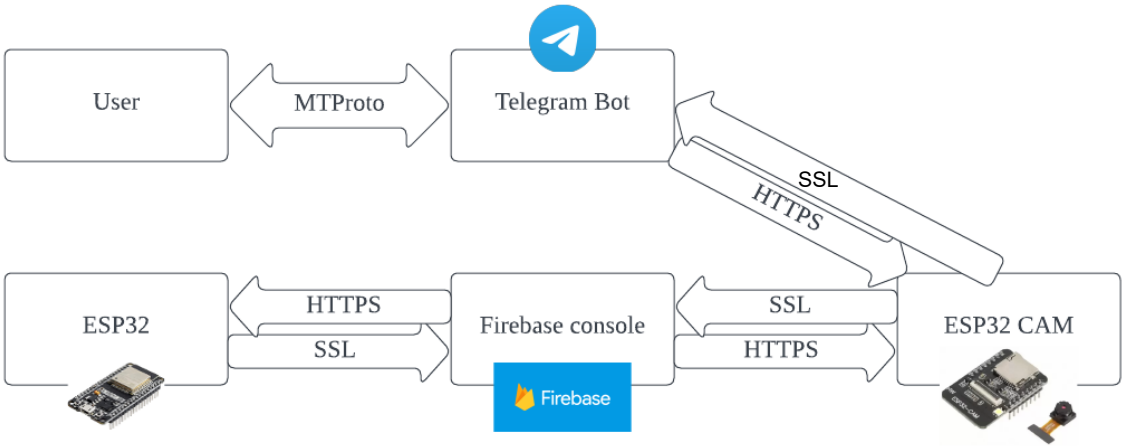
We will be using 2 microcontrollers in this system, ESP8266 acts as the middleman and will handle the motor, the camera, telegram integration, and communication with Firebase which also hosts the webserver. The data will be sent through and uploaded onto Firebase Console. Not only will we be storing our data in Firebase, but also using it as an OTP generator. Firebase has embedded encryption available making it a reliable and relatively safe option.

System Flow Chart



The user will have to log in with their username and password on a web browser. The user has 3 attempts before he is logged out and his picture will be taken by the ESP32 CAM. Upon a successful log in an OTP will be generated and sent to the user and he has to input it into the web browser. Once the right OTP is inputted the door will open.

Dataflow



Users access the Firebase console as an interface to unlock the door lock. Firebase will contain all the residents usernames and passwords along with their email address for verification. Besides acting as a database, Firebase will host a web server as well as communicate with the ESP8266 via HTTPS. ESP8266 is connected physically to the ESP32 CAM and will send an output to the chip for the camera to be activated whenever password is detected incorrect 3 times. The ESP32 CAM then sends the image to telegram via MQTT. Additionally, the homeowner can request for an image capture anytime he wants by slash commands in the bot's chat.

Justification

Why a Website?

For our final solution, we decided to choose a website over a mobile application as it is more accessible, easier for deployment, and provides various encryption methods to help protect user data. Additionally, we will be hosting our website locally to make it hard to access unless you are within the local WiFi range.

Application	Pros	Cons
Website (HTML,CSS,JS)	Websites can be secured by HTTPS and SSL encryption, helping to protect user data.	Websites are vulnerable to various attacks such as DDoS, SQL injection, and XSS
	Websites can benefit from firewalls and security plugins.	Websites will not be able to take full advantage of features on mobile devices.
	Web Applications can be easily updated to address security vulnerabilities	Depending on use, websites generally have slower load times which may impact user experience.
	In terms of distribution, a website would be easier for our context which is home use.	
	Websites can be accessed with many devices.	
Mobile Application	Mobile applications may make full use of features such as notifications, location services, and alerts	Mobile applications may be harder to develop, especially if they need to be available to multiple platforms.
	Since mobile apps are directly installed into the device, they can be faster and more efficient than websites.	They are typically more difficult to modify or update as compared to a website.
	From a business perspective, mobile applications would be more accessible and easier to deploy once published onto the App Store, or Google play store	Mobile applications are more prone to code injection as well as reverse engineering.
	Mobile applications can benefit from user authentication features already included inside the device	
	To make it more secure,, applications may be “sandboxed” and operate in a protected environment.	

Why Firebase Console?

For this project, we decided to use the Firebase console to control backend operations as well as a realtime database for our project. Below are some of the reasons we decided to use Firebase:

1. **Built-in authentication and user management.** Having access to such features would make it easier for us to secure the process of Access Control to our realtime database.
2. **Realtime Database.** Having access to a real-time database would allow us to use Firebase as a backbone for our project. This ensures there are no inconsistencies in the data provided and provides the latest data in real time which is necessary for our project.
3. **Access Control and Rules.** Using Firebase would allow us to grant access to only authorised users and is fully customizable so that we are able to give different levels of access to each respective user.
4. **Encryption.** Firebase offers both client-side and server-side encryption for its storage services. Firebase uses the AES-256 algorithm for its encryption. Additionally, Firebase supports HTTPS, which encrypts data in transit between itself and clients.
5. **Certificates of Compliance.** All of Firebase services have successfully completed the following evaluations:
    a. **ISO 27001**
    b. **SOC 1**
    c. **SOC 2**
    d. **SOC 3**

    And some services are in the process of completing the **ISO27017** and the **ISO27018** certification process which include Authentication.

Overall, Firebase provides many secure storage options that will contribute to the comply with TR64 and create a secure End-to-End Door system.

Attack Surface	TR64 ID	Description
Website	CS-01	One-time password has a cryptoperiod of 90 seconds
	AP-01	User will be locked out indefinitely after 5 failed login attempts
	AP-02	User will need to enter login and enter a OTP
	AP-05	After keying in OTP user will be brought back to login page
Firebase	CS-02	Uses AES-256 to encrypt data at rest
	CS-03	Data encrypted at rest using AES-256
	IA-01	Users password will be hashed
	IA-02	User will need to enter a OTP
	NP-01	Firebase will double check user credentials in order to read and write to and from database
	DP-01	Data in transit is encrypted using TLS 1.3, data at rest is encrypted using AES-256 and passwords are also hashed
	DP-04	Implement role-based access control
	RS-02	TLS 1.3 enables perfect forward secrecy by default
ESP32	AP-02	Requires API key and login
	AP-03	Device will be located in the home
	AP-04	Device will be secured in a box
Telegram	DP-01	Data in transit is encrypted using SSL
Network	MT-01	Used a strong password for Wi-Fi network
	LP-01	Conduct threat modelling to identify and analyse threats

API key + login esp32
Cam to telegram ssl
API key for telegram
Chat Id for telegram
Firebase SSL certificate

Threat Type	Mitigation
Spoofing	TLS is protected against replay attacks using the Message Authentication Code(MAC)
Tampering	Data at rest is encrypted using AES-256 and data in transit is encrypted using TLS 1.3

Repudiation	One-time password shows the email of the user who requested it
Information Disclosure	Role-based access control prevent unauthorised access to information
Denial of Service	Strong WiFi password, MAC address filtering, turn off ports that are not used, Firewall
Escalation of Privilege	Access to the physical system is needed