Data Center Security System.

June 2022

Kareen Ziadat

Contents

Standard architecture, frameworks, tools, hardware, and APIs

Architecture

No.	Layer Name	Functionality
1	Perception Layer (physical/device layer)	Its primary function is to collect data from physical devices (sensors) placed in an environment. Temperature, humidity, and motion sensors, cameras, and GPS receivers are examples of these. The purpose of this layer is to observe and record real-world events or circumstances.
2	Network Layer (communication layer)	The network layer handles the transferring of data from the perception layer to higher levels. It allows communication between devices, gateways, and the cloud, ensuring reliable and secure data transfer. This includes Wi-Fi, Bluetooth, cellular networks (2G/3G/4G/5G), and other IoT protocols.
3	Data processing Layer	refers to the software and hardware components that collect, analyze, and interpret data from IoT devices. This layer receives raw data from devices, processes it, and makes it available for further analysis or action. It includes a variety of technologies and techniques used to extract useful insights from data and make decisions based on that data.
4	Business Layer	This layer has direct contact with the end user. It delivers user-friendly interfaces and functionality for accessing and controlling IoT devices. It also comprises analytics and processing technologies such as machine learning techniques and data visualization tools. This covers a wide range of software and apps designed to interface with the underlying IoT infrastructure.

Frameworks and Tools

No.	Framework Name	Description
1	MQTT (Message Queuing Telemetry Transport)	It provides a scalable and adaptable architecture for connecting sensors, devices, and applications, enabling real-time data sharing. Designed for low-bandwidth, high-latency networks, and restricted devices.
2	Apache Kafka	Distributed streaming platform used to develop scalable and real-time data pipelines. Kafka enables real-time processing, data streaming, and analytics by allowing the seamless integration of numerous IoT devices and applications.
3	Arduino IDE	Open-source electronics platform that offers a simple environment for programming microcontrollers. The NodeMCU can be programmed using the Arduino IDE after installing the ESP8266 board support package. The Arduino IDE includes numerous libraries and examples that can be utilized with NodeMCU.
4	TensorFlow	Open-source machine learning platform for developing and deploying AI models in IoT applications. It enables applications such as image and speech recognition, and predictive analytics. TensorFlow is suited for IoT deployments because it supports a wide range of hardware platforms, including embedded devices.

Hardware

No.	IOT	Functionality
1	Raspberry Pi	It can connect to a variety of sensors, actuators, and other external devices. It is low-cost, compact, and has widespread community support; it is frequently used for home automation and monitoring systems, among other things.
2	ESP8266	It consumes little power, is inexpensive, small, and simple to use, making it a popular choice for IoT prototyping. It has Wi-Fi built in, allowing it to connect to wireless networks and communicate with other devices over the internet. It can communicate with a variety of sensors, actuators, and electronic components and with cloud platforms.

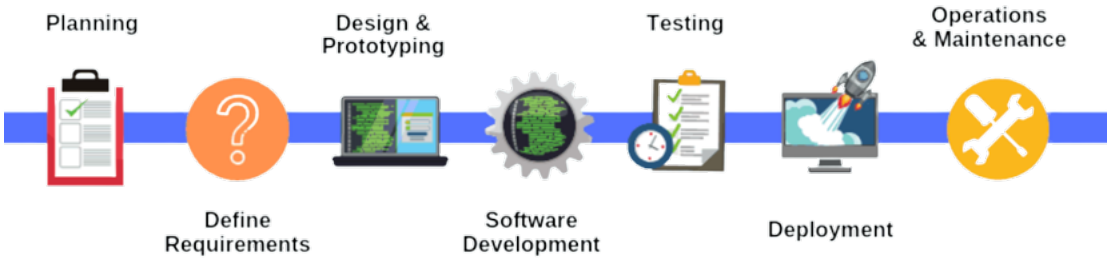
Problem description

Data centers are key components of the current business landscape, serving as the hub for data storage, management, and processing for businesses of all sizes. Massive amounts of sensitive and important data, including financial records, customer information, intellectual property, and other secret data, are stored in these facilities. As a result, a data center security breach can have serious consequences for businesses, including financial losses, reputational harm, and legal liability.

One of the major issues with data centers is that they serve as a single point of failure for intruders. An attacker who gains unauthorized access to a data center may be able to compromise the entire system.

The data center will be compromised if an unauthorized person (whether a previous employee or an intruder) acquires access to the key. As a result, the typical locking mechanism is insufficient because anyone can take the key and changing the key locks takes time. As a result, it is critical to employ additional effective security measures.

Impact of IoT components on the stages of SDLC



SDLC Stages

Planning: The project's **scope, aims, and objectives** are established. It also entails identifying the **resources** needed and calculating the project's **price and timetables**. Input from stakeholders is also provided.

Requirements Gathering: The project team identifies **the software's functional and non-functional requirements** at this stage. This is accomplished by collecting and analyzing information from stakeholders and compiling it into a complete requirements document.

Design: At this stage, the emphasis is on simulating how the software product will work. This includes specifying elements like the architecture, user interfaces, platforms, programming approaches, communications, security, and even prototyping.

Implementation (software development): This stage involves writing code, testing it, and debugging it to create the software.

Testing: Before releasing software to users, testing assures that it fulfills the necessary standards and requirements. Security testing (typically automated), case testing, performance testing, and other sorts of testing are carried out. Testing guarantees that the number of errors and malfunctions encountered by customers is reduced, resulting in greater customer satisfaction and utilization rates.

Deployment: The emphasis is on making the software application available to users, which includes the installation process in the target environments where it will be used. In addition to installation, the deployment step includes testing to ensure that the program works properly in the target environment.

Maintenance: The program is actively maintained and updated to ensure that it meets the changing needs of its users. Flaws and defects that were not detected during the testing phase may be revealed while the

software is used. These mistakes must be detected and fixed, which may necessitate further development cycles.

The impact of common IoT architecture, frameworks, tools, hardware, and APIs in the software development lifecycle (SDLC) can be significant.

Below is a breakdown.

Planning: A general IoT architecture provides a blueprint for designing software systems that incorporate IoT components. It helps identify the layers, applications and interactions required to achieve the desired IoT-enabled capabilities.

Furthermore, the availability of pre-defined frameworks, tools, and hardware for IoT development provides effective planning to provide insight into the capabilities, constraints, and integration possibilities of the many IoT fields.

Requirements Gathering: The architecture guides the identification of specific requirements for data collection, communication, processing, and deployment. It aids in the definition of functional and non-functional in an IoT environment.

Frameworks, devices, hardware, and APIs aid in comprehending the possibilities of IoT devices, sensors, communication protocols, and data processing methods (answers the question: what are the prerequisites for the successful deployment of these resources?)

Design: The architecture defines the communication patterns of the components and defines the data flow. Design standards, templates, and best practices for integrating IoT applications into software systems are provided via frameworks, tools, hardware, and APIs. (Easier to deploy IoT solutions)

Implementation: The architecture acts as a guide for implementing the software system, ensuring that all the necessary layers, protocols, and interfaces are well integrated. Moreover, Frameworks, tools, hardware, and APIs help to speed up the development process.

Testing: The IoT architecture specifies the testing requirements for ensuring system behavior and operation in an IoT context. This contributes to the software system's compatibility and reliability with IoT components. Hardware components and APIs enable actual devices to communicate with one another for comprehensive testing.

Deployment: The architecture guides the implementation by defining the infrastructure, connectivity, and security considerations required to deploy IoT-enabled software systems. Moreover, Frameworks, Tools, Hardware, and APIs support software implementation.

Maintenance: The architecture identifies associated components, interfaces, and dependencies that must be monitored and updated. To add, there are various tools for remotely monitoring, maintenance and updating IoT devices. They also offer alternatives for dealing with device failures, data consistency, and other security concerns.

My choice of IoT architecture, frameworks, tools, hardware, and APIs

Architecture

Perception layer: The keypad acts as a sensor in the perception layer, allowing users to enter their password. It is an appropriate solution for this application because it provides a simple and safe means of user authentication. Also the servo motor is used to sense if someone entered the data center.

Network layer: Wi-Fi technology allows for easy communication between the data center security administrator and the IoT device. Wi-Fi provides reliable and safe data transport, making it ideal for transmitting real-time warnings and notifications.

Data processing layer: The data processing layer receives the keypad password and compares it to the correct password. This comparison guarantees that the entered password is validated before proceeding with further activities, hence increasing system security. The servo motor detects if the distance is decreased indicating someone is present in the room.

Business layer: The servo motor is responsible for unlocking the door upon successful password entry. It provides a physical mechanism to control the door, ensuring secure access to the data center. Also, the buzzer acts as an audible alarm when an incorrect password is entered three consecutive times (deterrent and alerts nearby individuals to possible intrusion). An ultrasonic sensor was used to determine when a user entered the data center. And lastly, Using Blynk in the business layer allows for remote turning off the buzzer and provides real-time monitoring of potential intrusions. It offers a convenient platform for managing and interacting with the IoT system remotely.

Frameworks and tools

Arduino IDE: It is a simple programming language that will be appropriate for the project (if statements, etc.). It is fast, allowing for rapid prototyping and iterative development. It is also compatible with the hardware used.

Blynk: it will allow remote unlocking of the door, giving a convenient solution to manage data center access from a mobile device. It also allows for real-time monitoring of the system's condition, including displaying the time of a potential incursion and providing notifications. Blynk's cloud connectivity allows for real-time updates and notifications between the IoT device and the administrator's mobile device.

Hardware

Node MCU ESP8266: It offers a built-in Wi-Fi module and appropriate memory capacity. It also can be integrated with the Arduino IDE so easily programmed it is also less expensive.

Keypad: The password for unlocking the door is entered using a keypad. It provides a physical interface via which the user can interact with the system while also maintaining safe access management. They are simple to connect to the NodeMCU and integrate into the overall system design.

Servo motor: Once the right password is entered, a servo motor is responsible for physically unlocking the door. Servo motors are perfect for this project since they can be readily controlled and operated by Node MCU boards and provide precise movement control.

Buzzer: When unauthorized access is attempted, the buzzer provides an audible alert as an extra security measure. It raises immediate attention of a security violation. Also works with Node MCU boards.

ultrasonic: it detects whenever someone enters the room that way when the security administrator gets a notification about someone entering the data center after working hours, they he/she will be alerted and will take actions accordingly.

IoT application development plan

Objective

The project's main objective is to improve data center security by implementing an IoT-based access management system. By implementing extra security measures, the project hopes to go beyond traditional key-based access. The approach proposes that once a person enters the data center with a key, he or she won't be able to leave the room unless they enter the correct password. A buzzer will sound if an incorrect password is entered three times, and the security administrator will be notified of the incident which will prompt him to take action (Check surveillance / Go to the data center). Furthermore, the application will detect anomalous data center access, signaling a possible breach (entry at unusual time).

Scope

The scope includes the design and development of an IoT application that enables secure access control to data centers. The key components of the system include a keypad for password entry, a servo motor for physical door unlocking, an ultrasonic sensor that detects entry to the room and a buzzer for audible alerts. The application will be built using the NodeMCU ESP8266 board programmed with the Arduino IDE.

Functionalities:

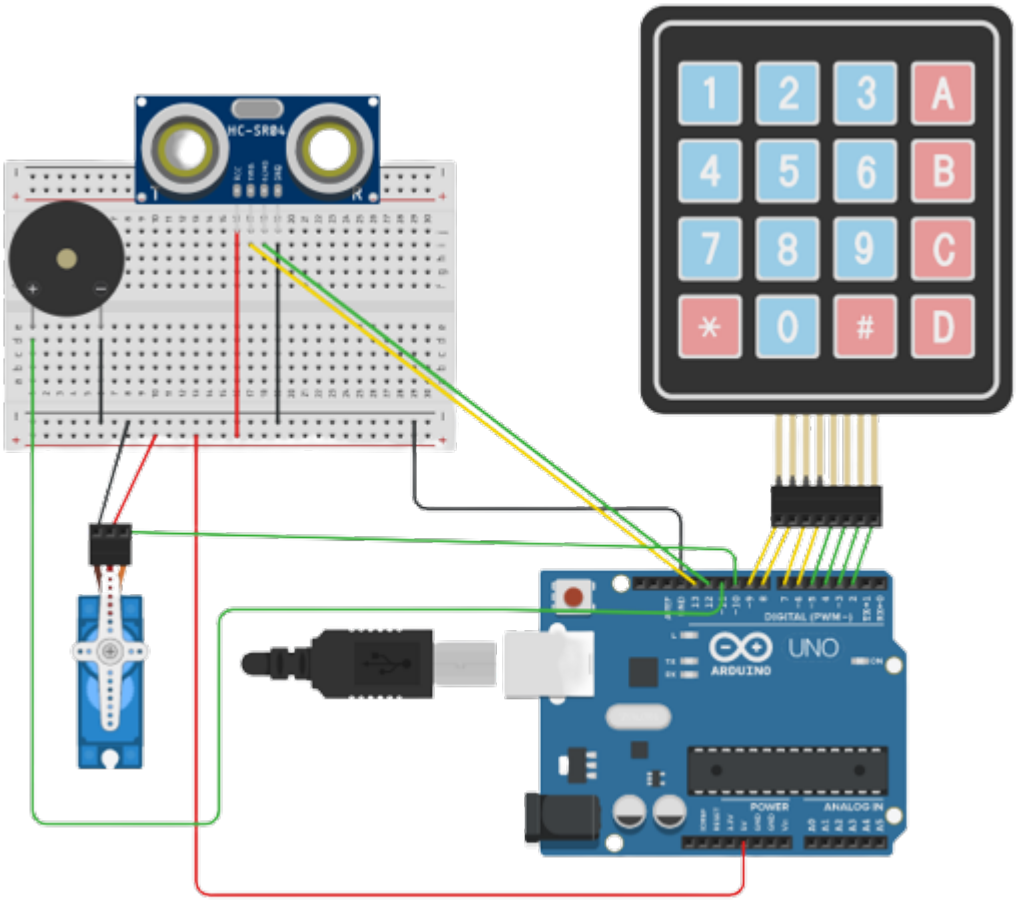
1. When trying to leave the data center, users will be required to enter a password on the keypad. The entered password will be validated against the correct password. Users will be able to get out of the room by entering the correct password.
2. Upon successful password entry, the servo motor will be triggered to unlock the door, allowing the user to exit the data center. It will automatically lock after 7 seconds.
3. Incorrect password detection: If an incorrect password is entered three consecutive times, the buzzer will sound an audible alert.
4. In case of an incorrect password entry, an alert will be sent to Blynk, to notify the data center security administrator of a possible security breach.
5. In case an entry to the room is detected by the ultrasonic a notification will be sent to blynk.

Out of scope:

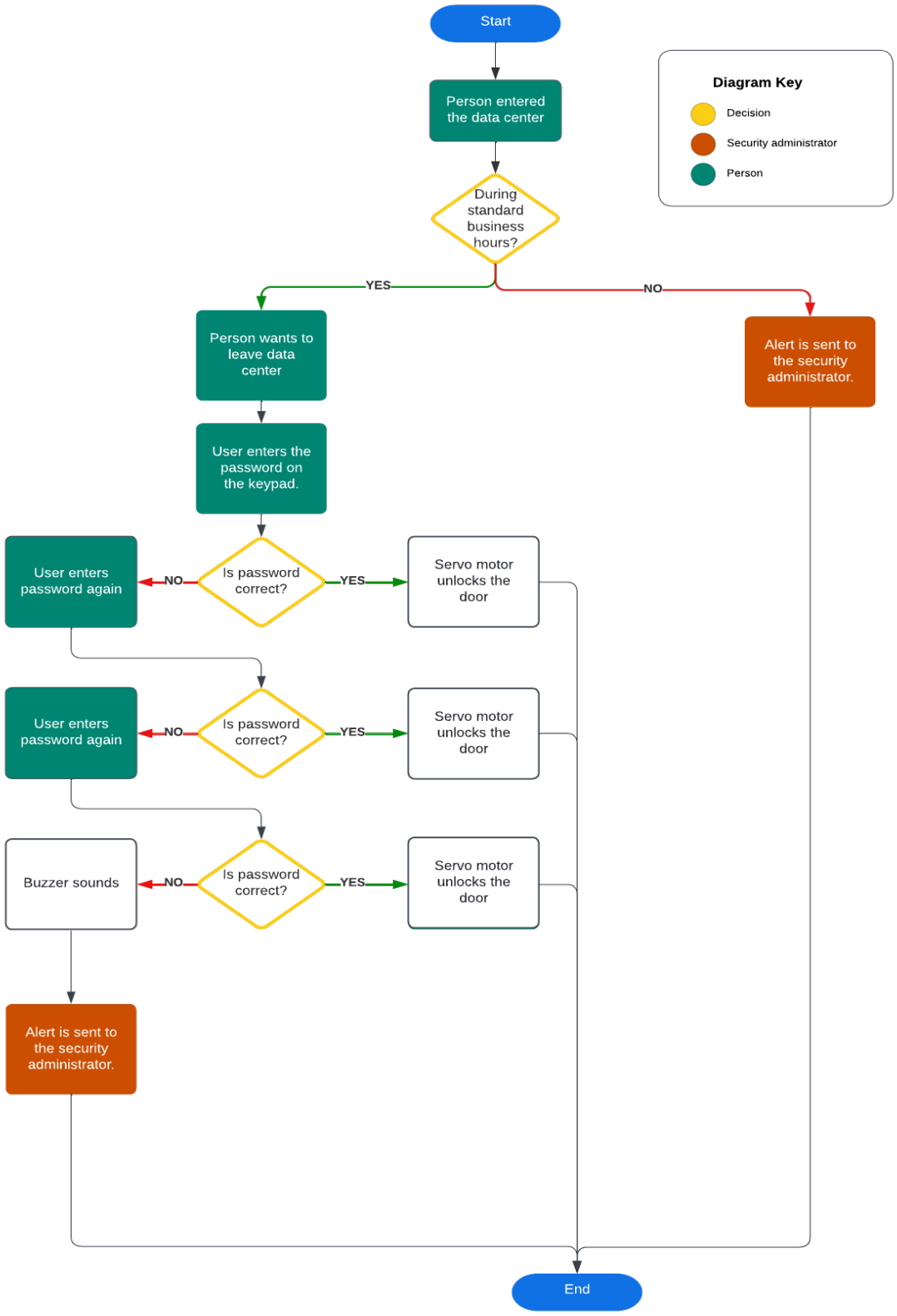
1. Physical installation of the access control system.
2. Integration with additional security measures (surveillance cameras).

IOT architecture

Tinkercad blueprint



Flowchart


Start
Person entered the data center
During standard business hours?
YES
NO
Person wants to leave data center
Alert is sent to the security administrator.
User enters the password on the keypad.
User enters password again
NO
Is password correct?
YES
Servo motor unlocks the door
User enters password again
NO
Is password correct?
YES
Servo motor unlocks the door
Buzzer sounds
NO
Is password correct?
YES
Servo motor unlocks the door
Alert is sent to the security administrator.
End
Diagram Key
Decision
Security administrator
Person

Layers

Perception Layer: We will use a keypad as a sensor for password input, allowing users to unlock the door. Use an ultrasonic sensor that will detect entry to the data centers.

Network Layer: We will employ Wi-Fi technology for reliable and secure communication between the IoT device and the data center security administrator.

Data Processing Layer: We will develop algorithms to compare the entered password with the correct password and validate the authentication process. The code will also be able to control actions like send notification, move servo motor, and sound buzzer. The code will also detect entry by using the ultrasonic and comparing the distance to a set threshold.

Business Layer: Utilize a servo motor to physically unlock the door, a buzzer for audible alerts, and Blynk for remote turning off the buzzer and real-time monitoring.

Frameworks and platforms

Arduino IDE: Utilize Arduino IDE for programming the NodeMCU ESP8266 board and integrating the hardware.

Blynk: Use it to enable real-time monitoring, and notification management.

Tinkercad: It will be used to prototype the projects without the need of actual components, to design our circuit design and test our code.

Hardware

NodeMCU ESP8266: It is the central hardware component with built-in Wi-Fi module that is compatible with Arduino IDE. (Around 7 jds)

Keypad: It will be connected to the NodeMCU board to allow users to enter the password for unlocking the door securely. (0.70 jds)

Servo Motor: Connect the servo motor to the NodeMCU board to physically unlock the door upon successful authentication. (3 jds)

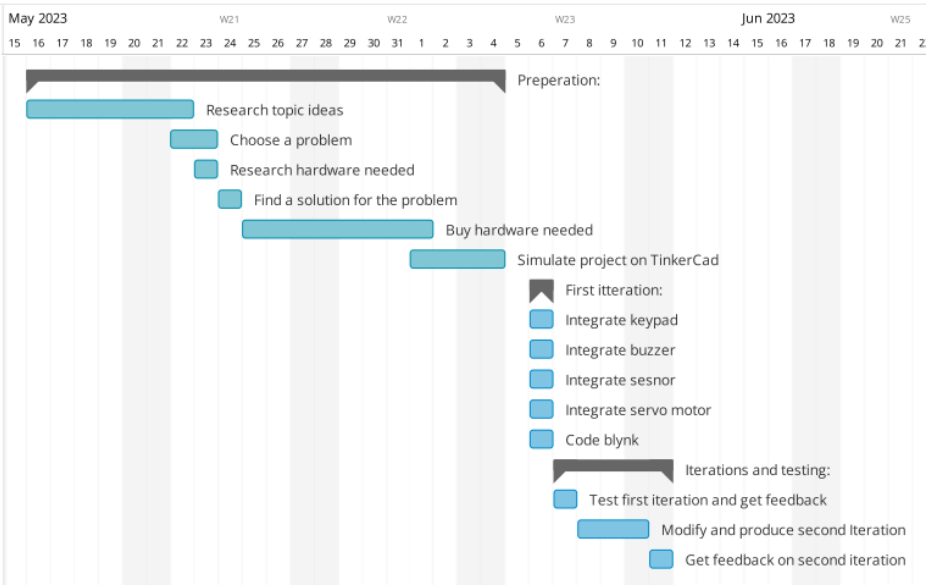
Buzzer: The buzzer is also connected to the NodeMCU to generate an audible alert when an incorrect password is entered three consecutive times. (0.5 jds)

Ultrasonic sensor: it will be connected to detect if anyone entered the data center. (2.5 jds)

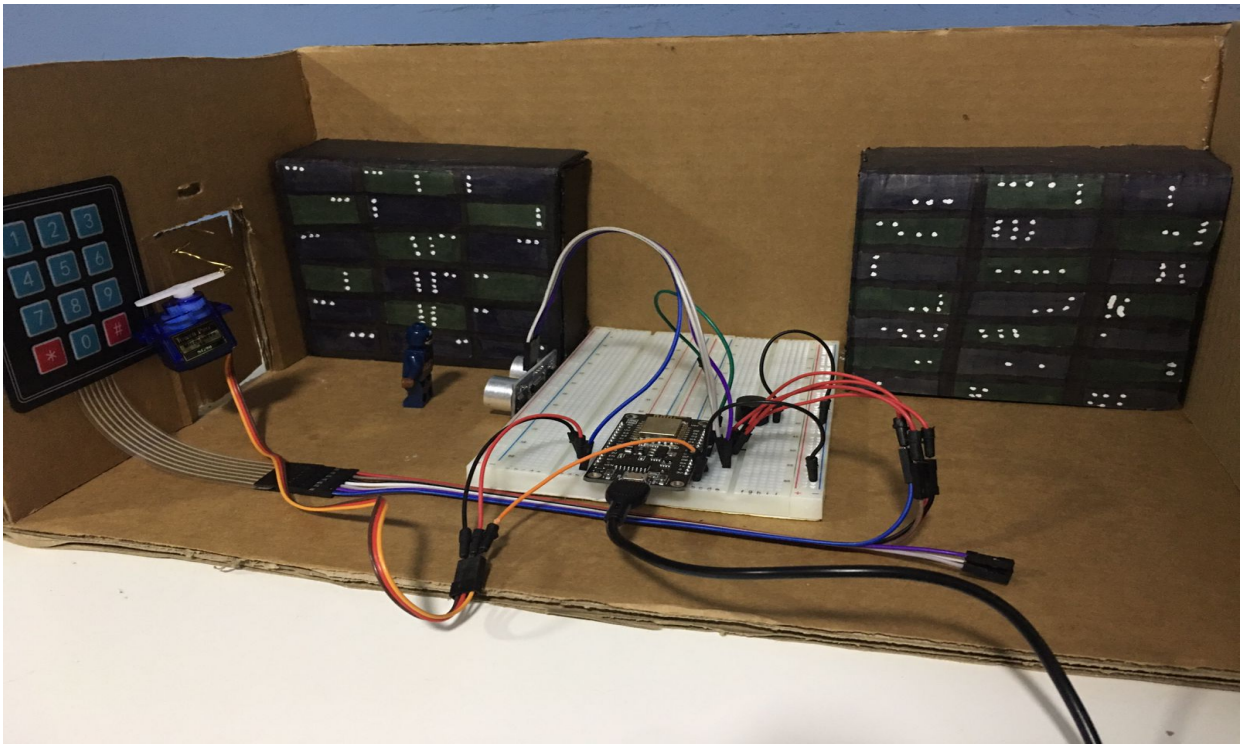
Development steps:

Gantt chart

	ACTIVITIES	START	DUE
	Preperation:	16/May	04/Jun
1	Research topic ideas	16/May	22/May
2	Choose a problem	22/May	23/May
3	Research hardware needed	23/May	23/May
4	Find a solution for the problem	24/May	24/May
5	Buy hardware needed	25/May	01/Jun
6	Simulate project on TinkerCad	01/Jun	04/Jun
	First itteration:	06/Jun	06/Jun
8	Integrate keypad	06/Jun	06/Jun
9	Integrate buzzer	06/Jun	06/Jun
10	Integrate sesnor	06/Jun	06/Jun
11	Integrate servo motor	06/Jun	06/Jun
12	Code blynk	06/Jun	06/Jun
	Iterations and testing:	07/Jun	11/Jun
14	Test first iteration and get feedback	07/Jun	07/Jun
15	Modify and produce second Iteration	08/Jun	10/Jun
16	Get feedback on second iteration	11/Jun	11/Jun



Our prototype



End-user feedback:

Advantages:

1) The user-centric approach ensures a better experience by integrating end users in the testing and feedback process.

2) Based on customer feedback, the iterative development approach enables ongoing improvement. Each iteration builds on the one before it, refining the system and addressing user issues to produce an optimum solution.

3) Additional features were added because of customer input, including ultrasonic sensor distance adjustment, and notification alerts. These upgrades improve the IoT system's functionality and usability.

4) The selected IoT techniques illustrate the capacity to adjust to user needs and preferences. Based on user feedback, changes were made to the alarm sound, distance thresholds, and notification settings, allowing for a more personalized experience.

Overall, incorporating user feedback allowed us to improve the security of the data center reducing resources wasted by increasing the flexibility and adding functionalities like the notification.

Disadvantages:

1) The iterative process of gathering input, changing the prototype, and retesting with consumers can be time demanding. It takes several revisions and rounds of feedback to arrive at a suitable solution.

2) The feedback obtained from the same three users across the iterations may not represent the opinions and preferences of a larger user base due to the small sample size.

The low number of password iterations will decrease the security. To add, our solution may need training of staff.

Overall, users were impressed by the trap mechanism and found it extremely useful, but they wanted o to add more flexibility to make sure they can control false alarms and get more notifications.

Testing:

We will produce the first iteration, show it to three users to get feedback and based on the feedback gathered, the prototype will be modified, and a newer iteration will be developed. Another round of feedback with the same users will be held and the feedback gathered will be integrated. We will stop iterating once the user is okay with everything.

Iteration 1 feedback

When testing the buzzer its continuous sound was one long sound which didn't feel like an alarm and was irritating, later we made it sound like beeps.

Iteration 2 feedback

During the second phase, the focus revolved around devising a method to incorporate the keypad into the system while seamlessly integrating it with both the buzzer and servo. We experimented with the angles of the servo motor to simulate the door opening and closing until the user was satisfied with the simulation.

Iteration 3 feedback

We introduced the ultrasonic sensor in the third itteration. We initially coded it to display the message "door opened" on the serial monitor anytime the measured distance was less than 5cm. This distance was changed multiple times until it made sense and the last distance set was <10 CMs.

Iteration 4 feedback

After connecting the circuit to blynk and simulating how that will work to the user. We received feedback that they would also like to get a notification once a person is detected and not only see the value change as they are not going to be constantly watching the status on the application.

Iteration 5 feedback

When incorporating this feedback and showing it to the user again. They got so annoyed by the alarm noise and expressed that they would want to turn it off from their phone. We coded this and added the feature to accommodate the feedback.

Deployment and Monitoring:

The IoT system will be installed in the data center, ensuring proper physical connections. We will continuously monitor the system's performance, including password authentication, door unlocking, and alert notifications. It is important to make sure the application is regularly updated and maintained to address any potential security vulnerabilities and improving system functionality based on user feedback.

Finalizing

Review of the IoT application, problems it solves.

After accessing the data center room with the traditional key mechanism, a person will need to input a password to unlock the door. An authorized user will know the password and will be allowed to leave; however, an unauthorized individual will be unable to leave since the door will not unlock, producing a trap. The security administrator will have adequate time to deal with the intruder (who will be unable to exit the room) after receiving notification of the probable intrusion or hearing the buzzer sound.

It addresses the topic of data center physical security. Since prior employees may have access to the key or someone may steal it, the locking key mechanism is insufficient.

Potential problems when integrating into the wider system.

1. There is a possibility of false traps, in which authorized personnel may become stuck within the data center if they forget their password or experience technical difficulties with the door unlocking system.

2. While the approach gives the security administrator time to react with an intruder, it also relies on the administrator being available and responding to notifications or the buzzer sound.

3. The password should be managed well by frequently updating the password and securely storing it.

4. As the data center expands or more people want access, managing and upgrading passwords might become difficult.

A comparison of the final application with the original plan.

Initially, we planned to use every number on the keypad. However, due to the NodeMCU's restricted digital pin availability, we had to prioritize the use of the servo motor, ultrasonic sensor, and buzzer. As a result, we were forced to use only one row and three columns on the keypad, reducing the amount of password combinations and jeopardizing overall security. However, this limitation would not be an issue in a real-world solution because additional digital pins would be available.

Despite this constraint, the final product successfully incorporates all the desired functionalities, with the added convenience of remotely disabling the buzzer. The servo motor effectively opens the door upon entering the correct password, while the buzzer alerts the user and triggers a notification if an incorrect password is entered three times. Additionally, the servo motor can detect any person entering the data center and promptly sending a notification. These implemented functionalities demonstrate a well-executed planning process and the achievement of the primary objective: enhancing the security of the data center.

An evaluation of the overall success of the application.

First, the IoT application improved data center security by introducing an additional layer of authentication, which helps protect sensitive data and valuable assets from unauthorized access and potential breaches, providing businesses and individuals with peace of mind that their data is better protected. It also addressed the issue of insider threats, reducing the risk of previous employees carrying out resentful activities in the data center and so improving the business's overall security posture. Furthermore, deploying the application for the physical protection of the data center would increase business productivity and lower the cost of data recovery. From a societal standpoint, the IoT solution improves the broader cybersecurity ecosystem by providing an additional layer of protection for sensitive data. It is consistent with the growing importance of data privacy and security in a digital age.

While the application offers benefits, there are potential challenges that need to be considered. These include the risks of false positives, safety hazards (in case someone was trapped inside during a natural disaster), and human error which affect the overall success and impact of the application if not properly addressed. To add, the cost (infrastructure, hardware, software) and complexity of implementing and maintaining the system, as well as potential resistance to change from employees or stakeholders, may have an impact on the application's overall success and adoption rate.

References

Jena, S. (2020). *Architecture of Internet of Things (IoT)*. [online] GeeksforGeeks. Available at: https://www.geeksforgeeks.org/architecture-of-internet-of-things-iot/.

www.linkedin.com. (n.d.). *Software Development Life Cycle (SDLC) Quick Summary*. [online] Available at: https://www.linkedin.com/pulse/software-development-life-cycle-sdlc-quick-summary-laouali-abdou/.