Text, logo

Description automatically generated

Internet of Things Project

**DoorGuard**

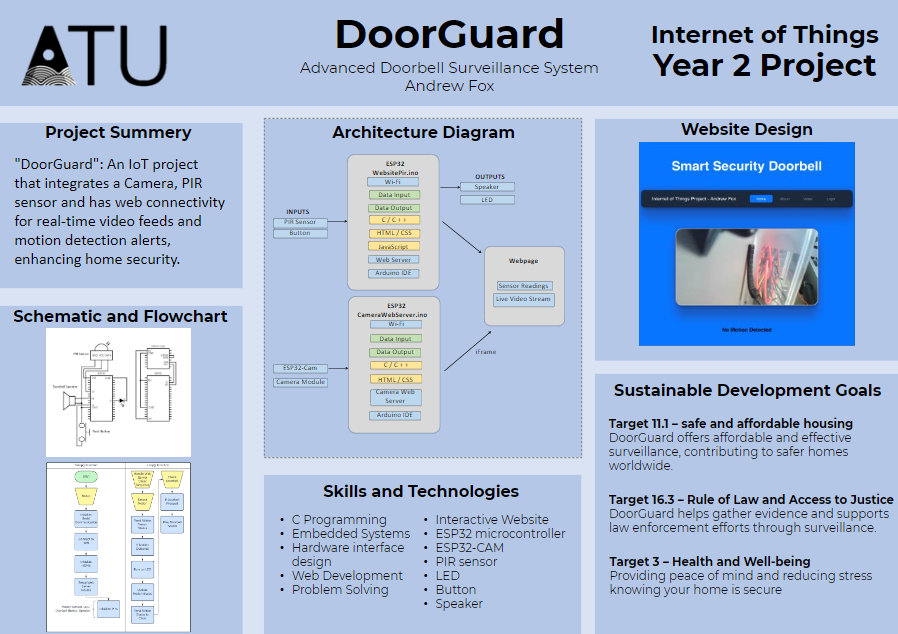
Advanced Doorbell Surveillance System

Andrew Fox

Bachelor of Software & Electronic Engineering

Atlantic Technical University

2023/2024



**Declaration**

This project is presented in partial fulfilment of the requirements for the degree of Bachelor of Engineering in Software & Electronic Engineering at the Atlantic Technical University, Galway campus.

This project is my own work, except where otherwise accredited. Where the work of others has been used or incorporated during this project, this is acknowledged and referenced.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Acknowledgements**

Use this section to acknowledge anyone, if you wish to, who might have helped during your project.

**Table of Contents**

[Summary 6](#_Toc558189338)

[Project Architecture 7](#_Toc1344213107)

[Development Platform and Tools 8](#_Toc511368891)

[Web Server 11](#_Toc1033223258)

[1.1 WiFi 11](#_Toc470622657)

[1.2 Web Server 11](#_Toc639434371)

[1.3 HTML & CSS Code 11](#_Toc1885438763)

[1.3.1 Responsive Design 11](#_Toc810724620)

[1.4 Web Design for Accessibility 11](#_Toc1502619991)

[Problem Solving 13](#_Toc60472733)

[During the project, there was two major challenges: 13](#_Toc915885469)

[ESP32-CAM Setup: Initially, configuring the ESP32-CAM module was challenging due to its complex setup process. 13](#_Toc1359836905)

[Impact of Project on Sustainability 14](#_Toc98869571)

[Conclusion 15](#_Toc610131003)

[References 16](#_Toc1067302663)

[Appendix 1: Code 17](#_Toc274743431)

[Appendix 2: Bill of Materials 18](#_Toc169413777)

[Appendix 3: Schematic 19](#_Toc1006240707)

[Appendix 4: Simulation 20](#_Toc1723861825)

# Summary

DoorGuard is an innovative IoT project aimed at enhancing home security through an advanced doorbell surveillance system. It integrates a camera, PIR sensor, button sensor, and temperature sensor, providing real-time video feeds and motion detection alerts accessible through a user-friendly web interface.

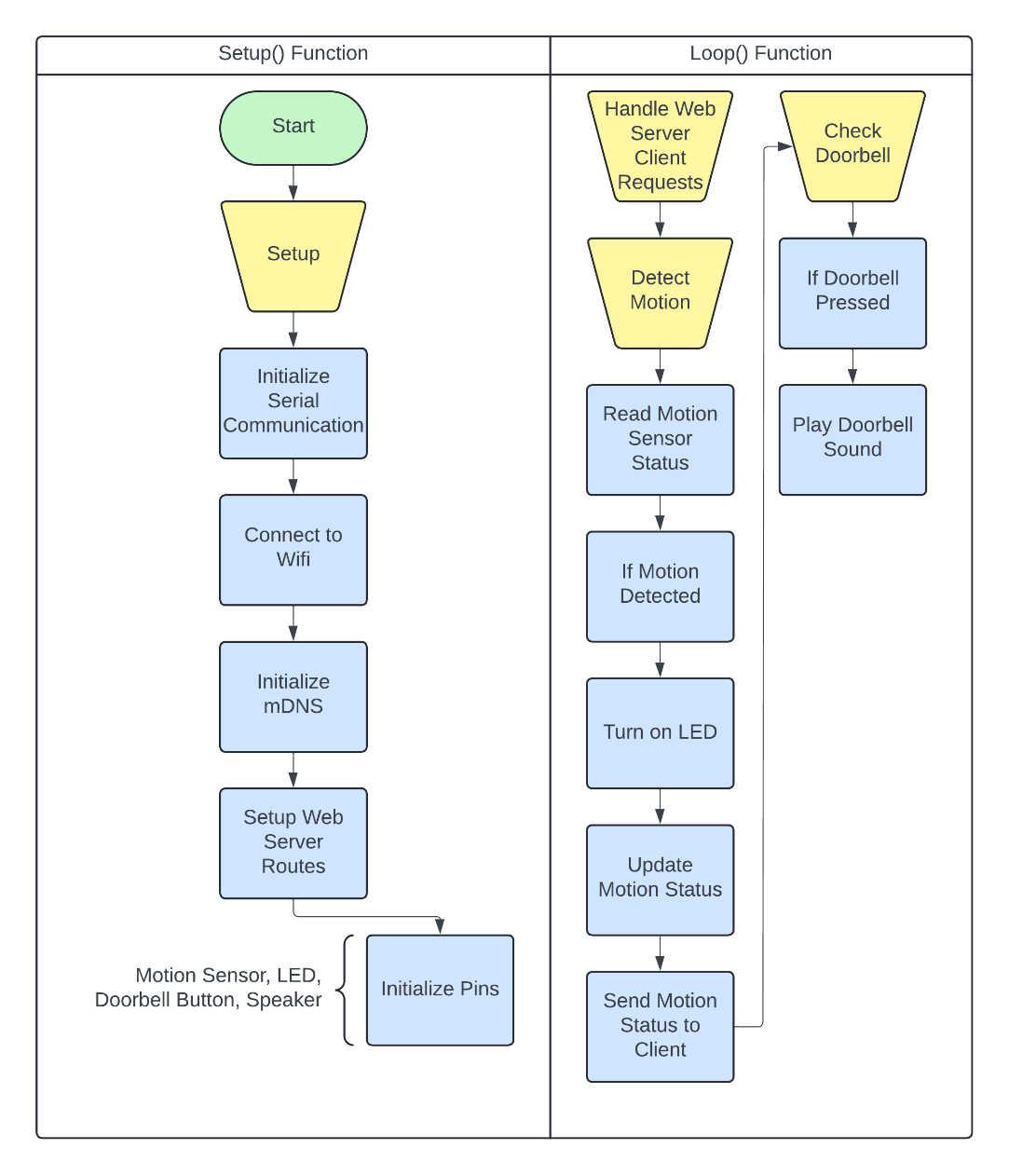
It uses an ESP32 microcontroller programming, ESP32-CAM for video streaming, and Arduino IDE for software development. Through this project, I have achieved a comprehensive solution for home security, aligning with the principles of sustainability and technological advancement.

This project contributes to United Nations Sustainable Development Goals by promoting safe and affordable housing, technological innovation, and access to information and communication technology.

# Project Architecture

The development platform I used is the ESP32 board, It's a powerful microcontroller with built-in Wi-Fi and Bluetooth capabilities. The ESP32 board is commonly used for IoT projects due to its versatility and low power consumption.

The Arduino IDE is used to write, compile, and upload code to the ESP32 board. It offers a user-friendly interface and extensive libraries, making it ideal for programming microcontrollers like the ESP32.

[1]‑

# Development Platform and Tools

Development Platform: ESP32

The ESP32 is used as the primary platform for DoorGuard development, offering Wi-Fi connectivity which I will be using to connect to the user’s phone.

IDE Used: Arduino IDE

Arduino IDE simplifies ESP32 programming with its user-friendly interface and extensive library support.

Web Development Platform: HTML/CSS/JS

HTML, CSS and JS are used to create the user interface of DoorGuard's web application, ensuring a user-friendly experience.

Hardware Components

Motion Sensor

DoorGuard utilizes a motion sensor PIR to detect movement. The sensor triggers an alert when motion is detected, enhancing the security features of the system. This alert is sent to the website.

Doorbell Button

A physical button is integrated into DoorGuard to function as a doorbell mechanism. When the button is pressed, it activates the doorbell functionality, alerting occupants of visitors.

Speaker

A speaker is used in the DoorGuard as the doorbell sound it is activated when the button is pressed.

Esp32-Cam

DoorGuard features a camera module that provides a live feed accessible through the website interface. This enables users to monitor their surroundings in real-time, enhancing the system's surveillance capabilities and providing peace of mind.

These hardware components work together to make DoorGuard effective for surveillance and improved security.

A diagram of a speaker system

Description automatically generated

# Web Server

Intro…

## WiFi

## Web Server

## HTML & CSS Code

You may want to include Flex design, responsive design, JavaScript and/or fetch API.

### Responsive Design

Include code and screenshots of website on different screen sizes if you used media queries.

## Web Design for Accessibility

Detail what you have included to ensure that your website is accessible.

You could break it down to the following headings:

* Perceivable - Web content is made available to the senses - sight, hearing, and/or touch
* Operable - Interface forms, controls, and navigation are operable
* Understandable - Information and the operation of user interface must be understandable.
* Robust - Content must be robust enough that it can be interpreted by a wide variety of user agents, including assistive technologies

Perceivable:  
e.g. For Screen readers to know what language to use:  
<html lang="en" >   
All links and alt tags were given meaningful names e.g.   
<p><a href="http:some big long address" >Make link text meaningful</a></p>

<p><img src="/image path.png" alt="Include meaningful description/make empty if unnecessary"></p>   
etc….

# Problem Solving

# During the project, there was two major challenges:

# ESP32-CAM Setup: Initially, configuring the ESP32-CAM module was challenging due to its complex setup process.

# Impact of Project on Sustainability

Project Application: DoorGuard's application in enhancing home security contributes to sustainable living by promoting safety and peace of mind for homeowners. By leveraging IoT technology, DoorGuard offers a cost-effective and efficient solution to safeguarding residential properties against intrusions and unauthorized access, aligning with the principles of sustainable urban development.

Impact on United Nations Sustainable Development Goals (UN SDGs): DoorGuard positively impacts several UN SDGs, including:

Target 11.1 – safe and affordable housing

* DoorGuard offers affordable and effective surveillance, contributing to safer homes worldwide.

Target 16.3 – Rule of Law and Access to Justice

* DoorGuard helps gather evidence and supports law enforcement efforts through surveillance.

Target 3 – Health and Well-being

* Providing peace of mind and reducing stress knowing your home is secure

Accessibility: Ensuring inclusivity for all users is a priority for the project's website. This involves using clear HTML styling making the website usable for individuals with different abilities.

Power Management: DoorGuard manages its power usage to extend device lifespan and reduce energy consumption. By employing efficient power management methods and using low-power hardware components, the project minimizes its environmental footprint and promotes sustainable resource utilization.

# Conclusion

DoorGuard stands out as a significant addition to home security technology. By integrating various sensors and hardware components with well-designed software, this project offers a reliable solution for enhancing residential security.

Using the ESP32 microcontroller platform and ESP32-CAM module, along with Arduino IDE for software development, DoorGuard provides an effective, affordable, and user-friendly system.

DoorGuard contributes to sustainability by promoting safe and affordable housing, aligning with UN Sustainable Development Goals. Additionally, it prioritizes accessibility, ensuring usability for people with diverse abilities.

Looking ahead, there are opportunities for further improvement and expansion of DoorGuard's features. Future iterations could explore remote access, facial recognition, or integration with smart home systems. Overall, DoorGuard signifies progress towards safer, more sustainable living environments, showcasing the potential of IoT technology in addressing real-world security challenges.

# References

|  |  |
| --- | --- |
| [1] | Espressif, "ESP32\_Datasheet," 2022. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32\_datasheet\_en.pdf. |
| [2] | Mathsworks, "IoT Analytics - Thingspeak Internet of Things," [Online]. Available: https://thingspeak.com/. |
| [3] | Texas Instruments, "www.ti.com," 1999. [Online]. Available: https://www.ti.com/lit/ds/symlink/lm35.pdf. |
| [4]  [5] | Arduino, "www.arduino.cc," Arduino, 6 April 2022. [Online]. Available: https://www.arduino.cc/reference/en/libraries/servo/.  https://randomnerdtutorials.com/esp32-cam-pir-motion-detector-photo-capture/#:~:text=The%20ESP32%2DCAM%20is%20in,PIR%20motion%20sensor%20is%20received.  https://www.youtube.com/watch?v=ApGwxX6VVzk |

Some example references are given above. For example, the section in your report on your servo motors should have a sentence that includes [3] in it, referring to reference [3] here.

Use the References facility in MSWord as described in the Lecture. Use the IEEE style.

Include all your Datasheets.

# Appendix 1: Code

#include <WiFi.h>

#include <WiFiClient.h>

#include <WebServer.h>

#include <ESPmDNS.h>

#include "homepage.h"

#include <Arduino.h>

// WiFi credentials

const char\* ssid = "Iphone";

const char\* password = "12345678";

// Web server

WebServer server(80);

// Motion sensor and LED pins

#define MOTION\_SENSOR\_PIN 32

#define LED\_PIN 17

// Doorbell button and speaker pins

const int doorbellButtonPin = 12;

const int speakerPin = 25;

// Variables for motion detection

int motionStateCurrent = LOW;

int motionStatePrevious = LOW;

String motionStatus = "No Motion Detected";

unsigned long lastMotionTime = 0;

void setup() {

Serial.begin(115200);

// Connect to WiFi

WiFi.mode(WIFI\_STA);

WiFi.begin(ssid, password);

Serial.println("");

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.println("");

Serial.print("Connected to ");

Serial.println(ssid);

Serial.print("IP address: ");

Serial.println(WiFi.localIP());

// Initialize mDNS

if (MDNS.begin("esp32")) {

Serial.println("MDNS responder started");

}

// Setup routes for web server

server.on("/", handleRoot);

server.on("/motion-status", HTTP\_GET, handleMotionStatus);

server.on("/inline", []() {

server.send(200, "text/plain", "this works as well");

});

server.onNotFound(handleNotFound);

server.begin();

// Initialize motion sensor and LED

pinMode(MOTION\_SENSOR\_PIN, INPUT);

pinMode(LED\_PIN, OUTPUT);

// Initialize doorbell button and speaker

pinMode(doorbellButtonPin, INPUT\_PULLUP);

pinMode(speakerPin, OUTPUT);

}

void detectMotion() {

motionStatePrevious = motionStateCurrent;

motionStateCurrent = digitalRead(MOTION\_SENSOR\_PIN);

unsigned long debounceTime = 2000;

if (motionStatePrevious == LOW && motionStateCurrent == HIGH) {

if (millis() - lastMotionTime > debounceTime) {

Serial.println("Motion detected");

digitalWrite(LED\_PIN, HIGH);

motionStatus = "Motion Detected";

lastMotionTime = millis();

}

} else if (motionStatePrevious == HIGH && motionStateCurrent == LOW) {

Serial.println("Motion stopped");

digitalWrite(LED\_PIN, LOW);

motionStatus = "No Motion Detected";

}

}

void handleRoot() {

String message = homePagePart1;

server.send(200, "text/html", message);

}

void handleMotionStatus() {

server.send(200, "text/plain", "updateMotionStatus('" + motionStatus + "');");

}

void handleNotFound() {

String message = "File Not Found\n\n";

message += "URI: ";

message += server.uri();

message += "\nMethod: ";

message += (server.method() == HTTP\_GET) ? "GET" : "POST";

message += "\nArguments: ";

message += server.args();

message += "\n";

for (uint8\_t i = 0; i < server.args(); i++) {

message += " " + server.argName(i) + ": " + server.arg(i) + "\n";

}

server.send(404, "text/plain", message);

}

void checkDoorbell() {

if (digitalRead(doorbellButtonPin) == LOW) {

Serial.println("Doorbell pressed!");

playDoorbellSound();

}

}

void playDoorbellSound() {

tone(speakerPin, 500, 250); // Ding

delay(300); // Pause between ding and dong

tone(speakerPin, 800, 500); // Dong

delay(500); // Pause before repeating

noTone(speakerPin); // Stop the tone

}

void loop() {

server.handleClient();

delay(2);

detectMotion();

checkDoorbell();

}

# Appendix 2: Bill of Materials

Include a Bill of Materials table. State whether you got your part from ATU stores by placing a ‘y’ or ‘n’ under the ATU Stores column.  
If you can include the price of all components to show the approximate cost of your project. It’s not mandatory to have all costs.  
Mandatory sections to complete are Item, Quantity and ATU Stores.  
Don’t forget to multiply the unit cost by the number of components you use in your project e.g. the HC-05 module costs 19.45 each so the total cost for 2 is 38.90.  
You don’t need to cost resistors, breadboards, stripboard and wiring.  
It’s useful to create the table in Excel and then copy and paste here.   
Some students sourced parts themselves nd you can indicate this by using the ‘\*’.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Item** | **Quantity** | **Manuf** | **Manuf No** | **ATU Stores** | **Sourced from** | **Order No** | **Cost Euros** |
| ESP32 | 2 |  |  | y |  |  |  |
| ESP32 Cam | 1 |  |  | y |  |  |  |
| PIR Sensor | 1 |  |  | y | \* |  |  |
| Button | 1 |  |  | y |  |  |  |
| LED | 2 |  |  | y |  |  |  |
| Speaker | 1 |  |  | y |  |  |  |
| PCB Board | 2 |  |  | y |  |  |  |
|  |  |  |  |  |  |  |  |

Components marked with an asterisk indicate that the student sourced this component themself.

# Appendix 3: Schematic

# Appendix 4: Simulation

Include simulation circuits and code if simulation was carried out as part of your project.