



**MedTech**  
Mediterranean  
Institute of Technology

**ENGINEERING PROGRAM**

**ECE 355 FINAL REPORT**

**Alarm System Project using STM32F0**

**Professor: Noura Baccar**  
**Lab Instructor: Nourhen Akroufi**

**BY**

**Ahmad Hamouda**  
**Yassine Mtibaa**  
**Youssef Ben Moussa**

**2024-2025**

## 1. Abstract

This project focuses on the development of an embedded alarm system designed to monitor the status of a door and provide timely alerts when the door remains open for a prolonged period. The system employs a reed switch and magnet for door state detection, an STM32F0 microcontroller for control and decision-making, an LED for visual alerts, and a buzzer for audible alarms. Timing logic is implemented to manage delays and ensure a structured response to specific conditions.

The system operates as follows: upon activation by the user, it continuously monitors the door's status. When the door is detected as open, a 10-second countdown is initiated. If the door is closed before the countdown completes, the system resets and resumes monitoring. However, if the door remains open for the full 10 seconds, the system triggers both a visual alarm (LED) and an audible alarm (buzzer), each lasting 5 seconds. These alarms repeat every 10 seconds until the door is closed, ensuring persistent and effective notification. Once the door is closed, all alarms are deactivated, and the system resets to its initial monitoring state. The system can also be stopped manually by the user via a key.

The design process incorporates comprehensive system modeling, including sequence diagrams, use case diagrams, and block definitions, to ensure a clear and structured approach to development. The STM32F0 microcontroller serves as the core processing unit, chosen for its efficiency and compatibility with embedded system applications. Timing logic ensures accurate execution of delays and alarm cycles, enhancing system reliability.

This alarm system is cost-effective, user-friendly, and versatile, making it suitable for a variety of security and monitoring applications in residential, commercial, and industrial settings. Its modular design and clear feedback mechanisms (LED and buzzer) improve usability while ensuring the system operates reliably under diverse conditions. The project demonstrates how embedded systems can be effectively utilized for real-time monitoring and safety solutions, contributing to enhanced security and peace of mind.

<b>1. Abstract</b>	<b>2</b>
<b>2. Introduction</b>	<b>5</b>
<b>3. System Design</b>	<b>6</b>
<b>3.1. components of the System</b>	<b>6</b>
<b>3.2. Design</b>	<b>7</b>
<b>3.3. Description</b>	<b>7</b>
<b>3.3.1. Circuit design</b>	<b>7</b>
<b>3.3.2.Housing for components</b>	<b>7</b>
<b>3.3.3.Housing for reed switch (sensor)</b>	<b>8</b>
<b>4.System Diagrams and Requirements</b>	<b>9</b>
<b>4.1.System Requirements</b>	<b>9</b>
<b>4.1.1.Functional and non-functional requirements.</b>	<b>9</b>
<b>4.1.2.Explanation of how each requirement was achieved.</b>	<b>9</b>
<b>4.2.SysML Diagrams</b>	<b>10</b>
<b>4.2.1.Structure Diagrams</b>	<b>11</b>
<b>4.1.2.1.Block Definition Diagram</b>	<b>11</b>
<b>4.1.2.2.Internal Block Diagram</b>	<b>13</b>
<b>4.1.2.3.Package Diagram</b>	<b>14</b>
<b>4.2.2Behavior Diagrams</b>	<b>15</b>
<b>4.2.2.1Use case diagram</b>	<b>15</b>
<b>4.2.2.2State Machine Diagram</b>	<b>16</b>
<b>4.1.2.4.Activity diagram</b>	<b>17</b>
<b>4.2.2.4.Sequence Diagram</b>	<b>19</b>
<b>4.3.Link between diagrams and requirements.</b>	<b>20</b>
<b>5.Code Implementation</b>	<b>21</b>
<b>5.1.Setup</b>	<b>21</b>
<b>5.1.1.Configuration</b>	<b>21</b>
<b>5.1.2.Library Integration</b>	<b>21</b>
<b>5.1.2.1.I2C-LCD.h</b>	<b>21</b>
<b>5.1.2.2.Timer</b>	<b>21</b>
<b>5.1.3.Equations</b>	<b>22</b>

5.2.Code Explanation	22
6.Results	24
7.Summary	25
7.1. Recap of Research Goals and Achievements:	25
7.1.Challenges Encountered and Their Solutions:	25
7.2.Potential Future Enhancements:	25
8.References	26

## **2. Introduction**

In modern security systems, real-time monitoring and responsive alert mechanisms play a vital role in ensuring the safety of residential, commercial, and industrial environments. Doors and entry points are particularly critical, as they represent key access points for unauthorized entry or potential security breaches. Addressing this need, this project focuses on developing an embedded alarm system to monitor door status and provide timely alerts when the door remains open for an extended duration.

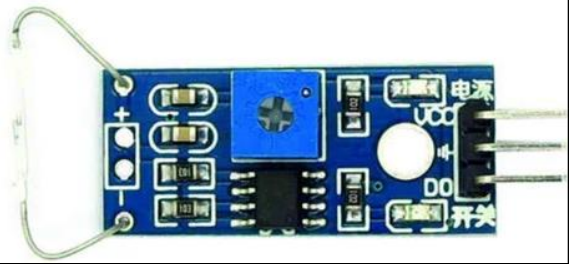



The proposed system is built around an STM32F0 microcontroller, which manages all system functionalities, including monitoring the door's state, initiating alarms, and controlling timing operations. A reed switch and magnet are employed as the primary sensors to detect whether the door is open or closed. When the door is left open for more than 10 seconds, the system activates both visual (LED) and audible (buzzer) alarms to alert users. These alarms persist in cycles until the door is closed, ensuring consistent and effective notification.

The project integrates several components to deliver a reliable and efficient solution. The system includes timing logic to manage countdowns and alarm durations, allowing it to respond appropriately to changes in the door's state. The user interface is simple yet functional, with a key used to activate and deactivate the system. Additionally, the system is designed to reset automatically once the door is closed, resuming its monitoring state without requiring manual intervention.

This project highlights the practical application of embedded systems in real-world scenarios, showcasing their ability to address everyday security challenges. By combining cost-effective hardware with efficient software logic, the system provides a reliable, user-friendly, and scalable solution suitable for diverse environments. The following sections of this report outline the design, implementation, and testing of the system, supported by diagrams and technical explanations.

### 3. System Design

#### 3.1. components of the System

Components discription	Component picture
Reed switch module: an electronic component used to detect magnetic fields. It typically consists of a reed switch (a small glass tube containing two ferromagnetic metal contacts) mounted on a circuit board, often with additional circuitry for ease of integration. Here's a breakdown of its features and operation	
Buzzer: an electronic device that produces sound when an electrical signal is applied. It is commonly used in devices and systems for alerts, notifications, and alarms	
RGB LED module: an electronic component designed to produce different colors by combining the primary colors red, green, and blue (RGB)	
Power bank: portable device that stores electrical energy in rechargeable batteries, allowing you to charge other devices like smartphones, tablets, and small electronics on the go	

Switch (ON/OFF): a simple, manually operated device that controls the flow of electrical current in a circuit. It is commonly used to turn devices or systems on and off by either completing or breaking the electrical circuit.



### 3.2. Design

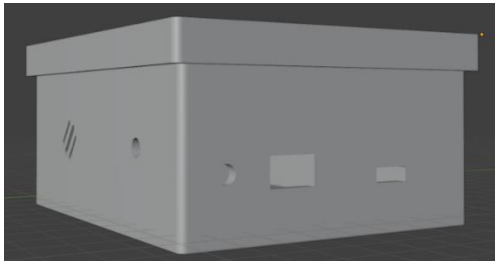


Figure 1box



Figure 2minbox

### 3.3. Description

#### 3.3.1. Circuit design

The circuit is composed of the reed switch module, RGB led, buzzer, LCD, on and off switch, breadboard, and microcontroller. Each component is connected to its designated GPIO and to a VCC and ground the connection goes like this:

- **Reed switch module:** the Digital output (DA) pin is connected to GPIO PB5. When there's a magnetic force applied to the reed switch that means the system is on when there's no magnetic signal applied to the reed switch then the system is off
- **Buzzer:** The input/output (I/O) pin is connected to GPIO PB15. If the system is on the buzzer is off if the system is off the buzzer is on.
- **RGB LED:** in this project only two led colors are used green to show the system is on and red to show the system is off the Red LED is connected to GPIO PB6, the Green LED is connected to GPIO PB7.
- **ON/OFF switch:** the switch is the bridge between VCC from the STM32 and the breadboard when the switch is on the components are supplied with power which means the system is working when the switch is off the components are not supplied with power which means the system is not working;

#### 3.3.2. Housing for components

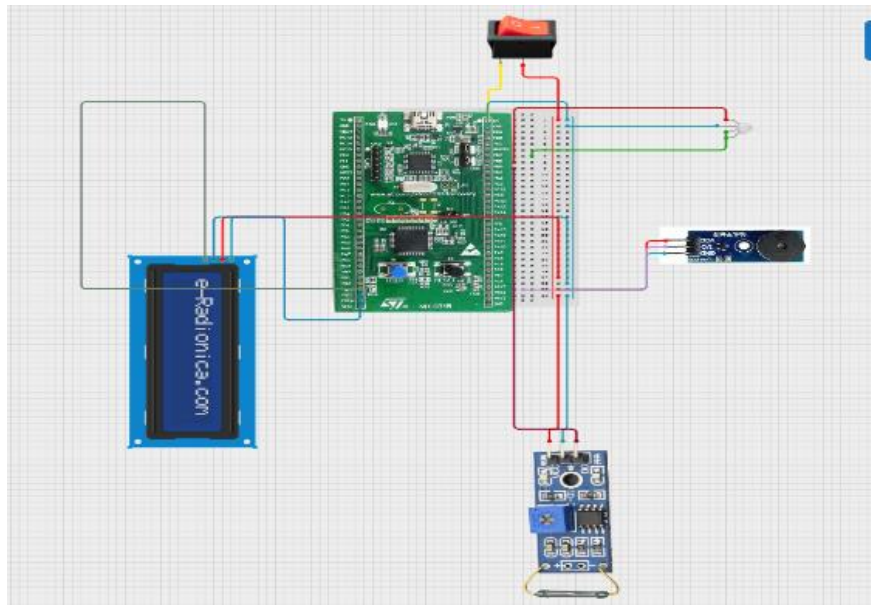
The housing is a Box with the dimensions 105x90x40mm the housing is to house the STM32, buzzer, RGB LED and the LCD. On one side of the box there's 2 square holes one for the cable connected to the STM32 and the other for the ON/OFF switch next to them there's a circular hole to the cable coming from the sensor connect to the microcontroller on the left side a circular hole for the RGB LED and a hole with grooves to hold the buzzer, the top compartment is where the LCD is attached.

### 3.3.3. Housing for reed switch (sensor)

The housing for the reed switch is a housing with dimensions 45x23x10mm it is to house the reed switch away from the magnetic field produced by any other component in the bigger housing

### 3.3.4. Component wiring

- **Configure GPIO pins:**
- **PB5** (Reed Switch) as input with pull-up resistor.
- **PB6** (green LED) as output (push-pull).
- **PB7** (red LED) as output (push-pull).
- **PB15** (Buzzer) as output (push-pull).
- **PB10** I2C2-SCL (LCD).
- **PB11** I2C2-SDA (LCD).





## 4. System Diagrams and Requirements

### 4.1. System Requirements

#### 4.1.1. Functional and non-functional requirements.

Requirement Type	Requirement
Functional	Detect door status
Functional	Set timer for delay
Functional	Activate buzzer if door remains open
Functional	Deactivate alarm if the door is closed
Functional	Repeat alarm if the door is still open
Non-Functional	Reliability
Non-Functional	Real-time performance
Non-Functional	User safety
Non-Functional	Maintainability

#### 4.1.2. Explanation of how each requirement was achieved.

Requirement	How It Was Achieved
Detect door status	Implemented using a reed switch and magnet to monitor whether the door is open or closed.
Set timer for delay	Achieved using the STM32F0 microcontroller with embedded timing
Activate buzzer if door remains open	Controlled by the microcontroller, which triggers the buzzer for 5 seconds when the countdown completes.
Deactivate alarm if the door is closed	The system continuously monitors the door state and deactivates both the LED and buzzer when the door is closed.
Repeat alarm if the door is still open	After a 10-second interval, the system reactivates the alarm (buzzer and LED) if the door remains open.
Reliability	Ensured by designing a robust feedback loop in the microcontroller logic to handle real-time changes in door state.
Real-time performance	Achieved by using the STM32F0 microcontroller, which supports precise timing and quick response.
User safety	The system provides timely alerts to ensure users are notified if the door remains open, enhancing security.
Maintainability	The modular design allows easy replacement of components (reed switch, buzzer, LED) and updates to software logic.

## 4.2 SysML Diagrams

**SysML (Systems Modeling Language)** is a graphical language for modeling complex systems, ideal for representing the architecture, behavior, and interactions within the alarm system that we are developing. By using SysML, we can model the system's components, how they interact, and the system's requirements. This approach will provide a clear and structured way to design our embedded alarm system, ensuring that the system operates as intended and meets its requirements.

### 1. Structure Diagrams

- **Block Definition Diagram (BDD):** this would define the system's key components (such as the STM32F0 microcontroller, reed switch, magnet, buzzer, and LED). It will specify the components and how they are related to each other (e.g., how the reed switch triggers the microcontroller).
- **Internal Block Diagram (IBD):** This diagram can be used to show how the internal parts of our system, such as the microcontroller, reed switch, and buzzer, are connected. For example, it can depict how the reed switch input triggers the microcontroller, which then controls the buzzer and LED.
- **Package Diagram:** This could be used to organize different parts of our system (hardware, software, etc.) into manageable packages and show the dependencies between those parts.

### 2. Behavior Diagrams

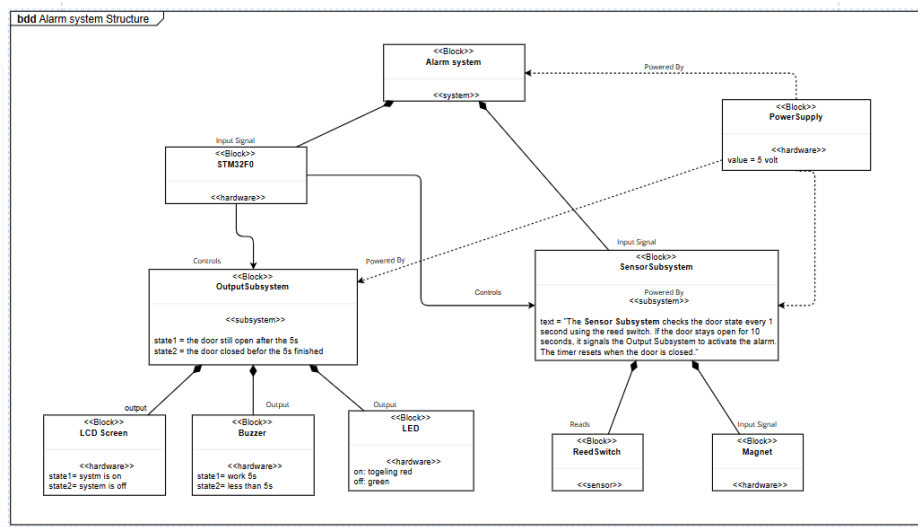
- **Use Case Diagram:** A Use case diagram for our alarm system can show the key scenarios or functions, such as "Detect Door Open," "Sound Alarm," and "Reset Alarm." This would capture interactions between the system components (e.g., the reed switch acting as a user input for detecting door status).
- **Activity Diagram:** This would represent the workflow of our alarm system, for example, the sequence of events that occur when the door is opened, followed by the 10-second wait, and then the periodic sounding of the buzzer. It can model the control flow and timing of these activities.
- **State Machine Diagram:** This diagram would be helpful to model the different states of the system, such as "Door Closed," "Door Open," "Alarm Active," and "Alarm Inactive." It would show how the system transitions between these states based on sensor inputs and time delays.
- **Sequence Diagram:** A sequence diagram can model the interactions over time, such as how the reed switch input triggers the microcontroller, which then activates the buzzer, and how these steps repeat based on the timing conditions.

### 3. Requirement Diagrams

- **Requirement Diagram:** For the system, this could show the requirements, such as the timing constraints (e.g., the buzzer sounds for 5 seconds after 10 seconds of door open), the maximum allowed delay between actions, or performance requirements (e.g., the buzzer should sound with a specific loudness). The requirement diagram would also link these requirements to your design elements.

## 4.2.1. Structure Diagrams

### 4.1.2.1. Block Definition Diagram



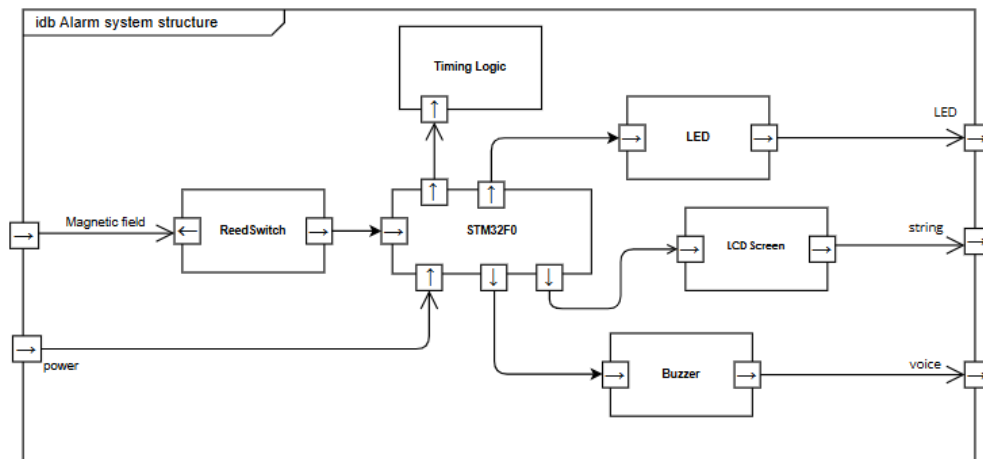
### Block Summary

1. **Alarm System** <<system>>  
Central system integrating all subsystems and components to monitor and respond to door state changes.
2. **Power Supply** <<hardware>>  
Supplies power to the system and all subsystems.
3. **STM32F0** <<hardware>>  
Microcontroller responsible for processing inputs and managing system outputs.
4. **Sensor Subsystem** <<subsystem>>  
Detects door state changes and sends control signals to the Output Subsystem if conditions are met.
5. **Output Subsystem** <<subsystem>>  
Manages the alarm outputs, activating the buzzer and LED based on input from the Sensor Subsystem.
6. **Reed Switch** <<sensor>>  
Senses the proximity of the magnet to determine the door state.
7. **Magnet** <<hardware>>  
Works with the reed switch to indicate the open/closed state of the door.
8. **Buzzer** << hardware >>  
Produces sound as an alarm when activated by the Output Subsystem.
9. **LED** << hardware >>  
Provides a visual alarm indicator (red when active, green when inactive).
10. **LCD Screen** << hardware >>  
display the current operational state of the system.

### Relationship Summary

From	To	Relation Type	Description
<b>Alarm System</b>	<b>Sensor Subsystem</b>	Composition	The Sensor Subsystem is part of the Alarm System.
<b>Alarm System</b>	<b>Output Subsystem</b>	Composition	The Output Subsystem is part of the Alarm System.
<b>Alarm System</b>	<b>Power Supply</b>	Dependency	The Alarm System depends on the Power Supply to function.
<b>Power Supply</b>	<b>Sensor Subsystem</b>	Association	The Power Supply provides power to the Sensor Subsystem.
<b>Power Supply</b>	<b>Output Subsystem</b>	Association	The Power Supply provides power to the Output Subsystem.
<b>Sensor Subsystem</b>	<b>Reed Switch</b>	Association	The Sensor Subsystem receives input signals from Reed Switch.
<b>Sensor Subsystem</b>	<b>Magnet</b>	Association	The Sensor Subsystem indirectly depends on the Magnet's proximity to the Reed Switch.
<b>Sensor Subsystem</b>	<b>Output Subsystem</b>	Dependency	The Sensor Subsystem sends control signals to the Output Subsystem.
<b>Output Subsystem</b>	<b>Buzzer</b>	Composition	The Buzzer is part of the Output Subsystem.
<b>Output Subsystem</b>	<b>LED</b>	Composition	The LED is part of the Output Subsystem.
<b>Output Subsystem</b>	<b>LCD Screen</b>	Composition	The LCD Screen is part of the Output Subsystem.
<b>STM32F0</b>	<b>Sensor Subsystem</b>	Association	The STM32F0 processes inputs from the Sensor Subsystem.
<b>STM32F0</b>	<b>Output Subsystem</b>	Association	The STM32F0 controls the Output Subsystem based on processed data.

#### 4.1.2.2. Internal Block Diagram



#### Components and Their Roles:

- **Reed Switch:**  
**Function:** Detects the magnetic field from the magnet, indicating whether the door is open or closed.  
**Input:** Magnetic field.  
**Output:** Sends a signal to the **STM32F0** based on the presence or absence of the magnetic field.
- **STM32F0 (Microcontroller):**  
**Function:** Acts as the system's brain, processing input signals from the Reed Switch and controlling the outputs.  
**Inputs:**  
Signal from the **Reed Switch** (indicating the door state).  
Power supply (to operate the microcontroller).  
**Outputs:**  
Control signal to the **Timing Logic**.  
Activates the **LED**, **LCD Screen** and **Buzzer** based on processed signals.
- **Timing Logic:**  
**Function:** Implements the delay and timing operations of the system.  
Monitors whether the door remains open for 10 seconds.  
Ensure the **Buzzer** runs for 5 seconds and repeats if necessary.
- **LED:**  
**Function:** Provides a visual alarm indicator.  
**Input:** Signal from the **Timing Logic**.  
**Output:** Toggles red (active alarm) or green (inactive) to indicate the door state.
- **Buzzer:**  
**Function:** Produces an audible alarm.  
**Input:** Signal from the **Timing Logic**.  
**Output:** Emits a sound (voice output) when activated.
- **Power Supply:**

**Function:** Supplies the necessary power to all components in the system.

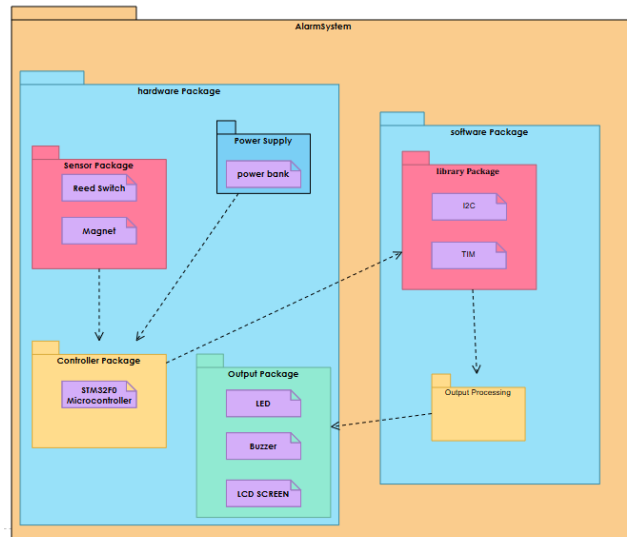
**Outputs:** Powers the **Reed Switch**, **STM32F0**, **LED**, and **Buzzer**.

- **LCD Screen:**

**Function:** display the current state of the system.

**Outputs:** print a string on the screen.

#### 4.1.2.3. Package Diagram



#### Hardware Package

- **Sensor Package:** This package contains the sensors responsible for detecting the alarm event.
  - **Reed Switch:** A sensor used to detect the opening of a door or window.
  - **Magnet:** A sensor used in conjunction with the Reed Switch to complete the circuit and trigger the alarm.
- **Power Supply:** Provides power to the system.
  - **Power Bank:** A battery used to store power for the alarm system.
- **Controller Package:** This package houses the microcontroller that manages the alarm system.
  - **STM32F0 Microcontroller:** The core component that controls the system's functionality.

#### Software Package

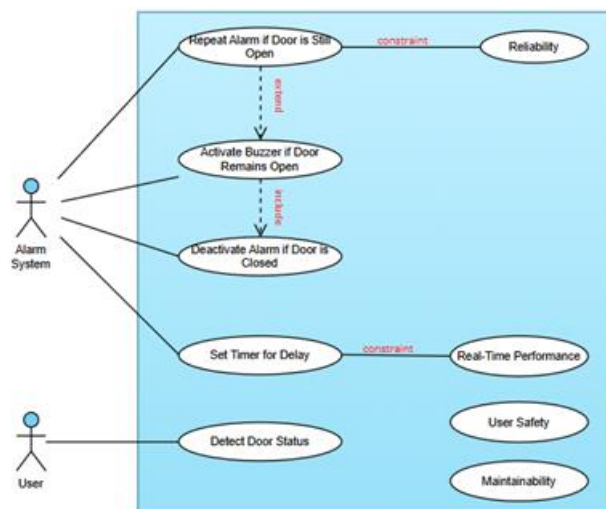
- **Library Package:** This package contains the libraries required for the system to function.
  - **I2C:** A communication protocol used for interacting with the sensors and outputs.
  - **TIM:** A timer module used for timing the alarm trigger and other functions.
- **Output Processing:** This package handles the responses triggered by the alarm system.
  - **LED:** A light-emitting diode that illuminates when the alarm is triggered.
  - **Buzzer:** A sound-producing component that emits an alarm signal when the alarm is activated.
  - **LCD SCREEN:** A display screen that provides information about the alarm status and any other relevant data.

## Interactions:

- The arrows in the diagram indicate the interaction between different packages.
  - The **Sensor Package** sends information to the **Controller Package**, which processes it and activates the necessary outputs.
  - The **Power Supply** provides energy for the **Controller Package** to operate.
  - The **Library Package** is used by the **Controller Package** to manage communication with the sensors and outputs.
  - The **Controller Package** activates the **Output Processing** package, resulting in actions like the LED turning on, the buzzer sounding, and information displayed on the LCD screen.

## 4.2.2 Behavior Diagrams

### 4.2.2.1 Use case diagram



## Actors:

- **User:** Interacts with the door and indirectly with the alarm system by opening or closing the door.
- **Alarm System:** The main system responsible for detecting the door's status and triggering the necessary alarm functions.

## Use Cases:

- **Detect Door Status:** Performed by the system to monitor whether the door is open or closed. This ensures the system knows when to trigger alarms or deactivate them.
- **Set Timer for Delay:** Adds a delay of 10 seconds after detecting the door's status before taking further action. This helps prevent unnecessary immediate triggering.
- **Activate Buzzer if Door Remains Open:** Triggers the alarm buzzer if the door stays open after the delay period.
- **Repeat Alarm if Door is Still Open:** Ensures the buzzer continues sounding at intervals (e.g., every 10 seconds) if the door remains open.
- **Deactivate Alarm if Door is Closed:** Turns off the alarm once the door is closed, ensuring the system resets.

## Relationships:

- Include:

Indicates a mandatory dependency. For example, the Activate Buzzer if Door Remains Open use case includes Set Timer for Delay because the delay is a necessary prerequisite for activating the buzzer.

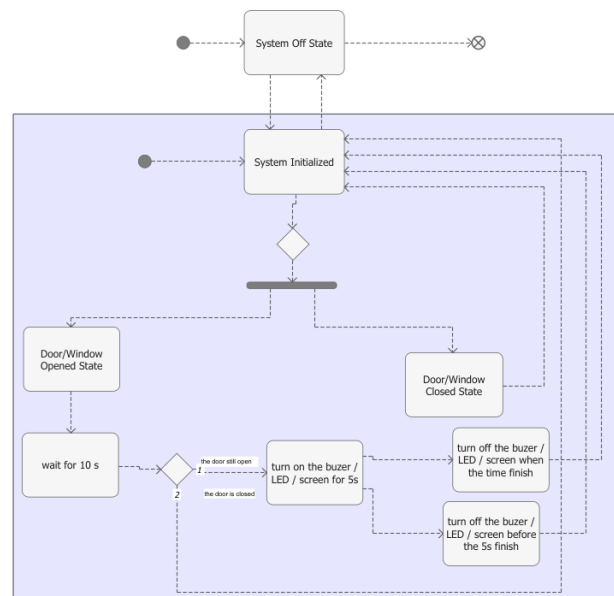
**- Extend:**

Indicates optional or conditional behavior. For example, Repeat Alarm if Door is Still Open extends Activate Buzzer if Door Remains Open, meaning it occurs only if the door remains open after the buzzer starts.

**Constraints (Constraints Diagram Notes):**

- **Reliability:** The system must reliably repeat the alarm if the door remains open, ensuring consistent behavior.
- **Real-Time Performance:** The system must operate in real-time to detect the door's status and trigger the alarm within the expected delay and intervals.
- **User Safety:** Ensures the system operates safely without causing harm, such as overly loud or harmful alarm sounds.
- **Maintainability:** The system should be designed in a way that it is easy to maintain or troubleshoot.

#### 4.2.2.2 State Machine Diagram



#### 1. States:

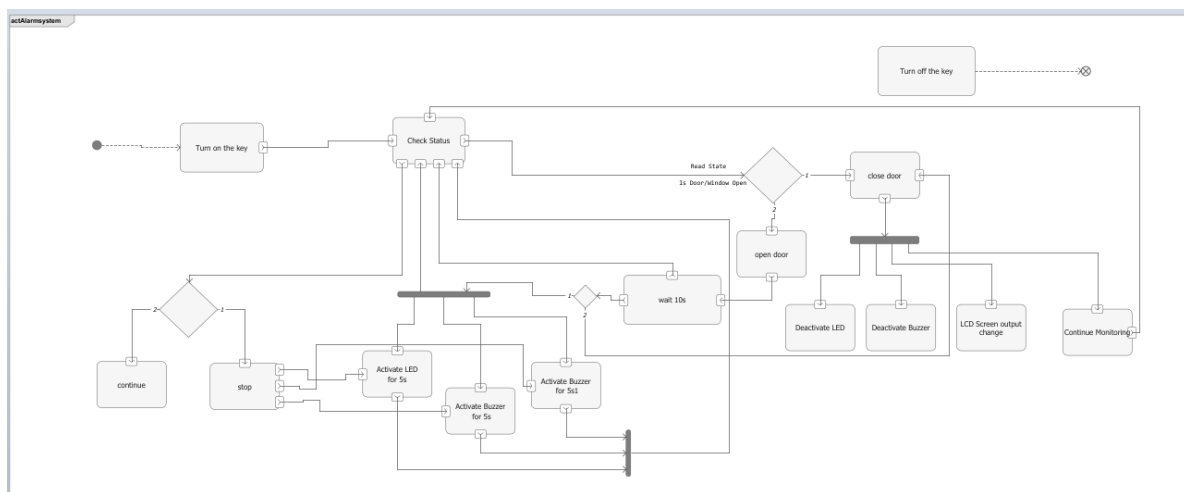
- **System Off State:** The initial state where the system is turned off and inactive.
- **System Initialized:** The system is turned on and ready to monitor the door or window status.
- **Door/Window Opened State:** Represents the state when the door or window is detected to be open.
- **Door/Window Closed State:** Represents the state when the door or window is closed.
- **Wait for 10s:** A temporary state where the system waits 10 seconds after the door or window is opened before taking further action.
- **Turn on the buzzer/LED for 5s:** An action state where the buzzer or LED is activated for 5 seconds to alert that the door is still open.
- **Turn off the buzzer/LED when the time finish:** Represents the state where the buzzer or LED is turned off, either because the time is over
- **Turn off the buzzer/LED before the time finish:** Represents the state where the buzzer or LED is turned off, either because the door has been closed or the alarm cycle has paused.



## 2. Transitions:

- The system starts in the **System Off State**.
- When the system is powered on, it transitions to the **System Initialized** state.
- The system then continuously checks the door/window status:
- If the door/window is opened, the system transitions to the **Door/Window Opened State** and waits for 10 seconds.  
After 10 seconds:
  - If the door/window is still open, the system transitions to the **Turn on the buzzer/LED for 5s** state and activates the alarm.
  - If the door/window is closed, the system transitions to the **Door/Window Closed State** without activating the alarm.
- After sounding the alarm for 5 seconds, the system transitions back to:
- **Wait for 10s** if the door/window remains open (repeating the alarm cycle).
- **Door/Window Closed State** if the door/window is closed, turning off the alarm.

### 4.1.2.4. Activity diagram



#### Turn on the Key:

- The system is activated by turning on the key.
- This action begins the process and transitions to the next activity.

#### Check Status:

- The system checks the current operational state:
  - Is the system ready to monitor?
  - Are all components functioning properly?

#### Read State (Is Door/Window Open?):

- The system continuously checks the state of the door or window using a sensor (e.g., reed switch and magnet).
- **If the door/window is closed:**
  - The system continues monitoring without triggering an alarm.
- **If the door/window is open:**
  - Proceed to the "Wait 10 Seconds" activity.

**Wait 10 Seconds:**

- A timer starts, allowing a 10-second for the door to be closed.
- If the door/window is closed within this period:
  - The system resets without triggering an alarm.
- If the door/window remains open after 10 seconds:
  - Move to the alarm activation process.

**Activate Buzzer/LED for 5 Seconds:**

- The buzzer sounds to alert about the open door/window.
- This process repeats every 10 seconds if the door remains open.
- **If the door is closed during the alarm:**
  - Deactivate both LED and buzzer, reset the system, and change the LCD screen's output.
- **If the door remains open:**
  - The buzzer and LED activation cycle continues.

**Close Door:**

- The door or window is closed, signaling the system to deactivate the alarm and reset the monitoring state.

**Deactivate LED and Buzzer:**

- The alarm indicators (LED and buzzer) are turned off.
- The system stops the alarm cycle and transitions to a normal monitoring state.

**LCD Screen Output Change:**

- The LCD screen updates to current state.

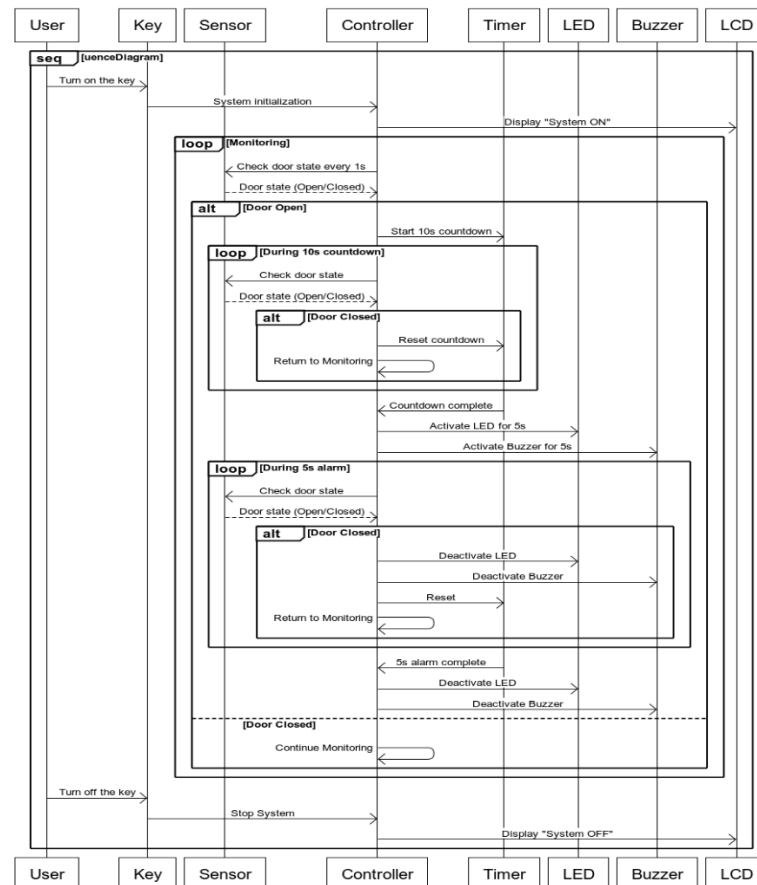
**Continue Monitoring:**

- The system resumes monitoring the door/window state.

**Turn off the Key:**

- The system is deactivated by turning off the key, ending the process.

#### 4.2.2.4. Sequence Diagram



#### Actors and Components:

- **User:** Initiates and stops the system by turning the key.
- **Key:** Represents the physical or logical interface for the user to turn the system on or off.
- **Sensor (Reed Switch):** Monitors the state of the door (open or closed).
- **Controller (STM32F0):** Processes input from the sensor and manages the alarm logic.
- **Timer (Timing Logic):** Controls time delays, including the 10-second countdown and buzzer activation timing.
- **LED (Visual Alarm):** Provides visual feedback when the alarm is triggered.
- **Buzzer (Audible Alarm):** Produces an audible alarm when the door remains open.

#### Process Flow:

- **System Initialization:**
  - The system is activated when the user turns the key.
  - The controller enters a **monitoring loop**, continuously checking the state of the door via the reed switch.
- **Door Monitoring:**

- The controller detects the **door state** (open/closed) in each loop iteration.
- If the door remains **closed**, the system continues monitoring without further actions.

- **Door Open Detected:**

- When the door is opened, the controller starts a **10-second countdown** using the timer.
- During the countdown, the system continues to monitor the door state.
- **If the door is closed during the countdown:**
  - The timer is reset.
  - The system returns to monitoring mode.
- **If the door remains open for 10 seconds:**
  - The timer signals the completion of the countdown.
  - The system activates the **LED (visual alarm)** and **buzzer (audible alarm)** for 5 seconds.

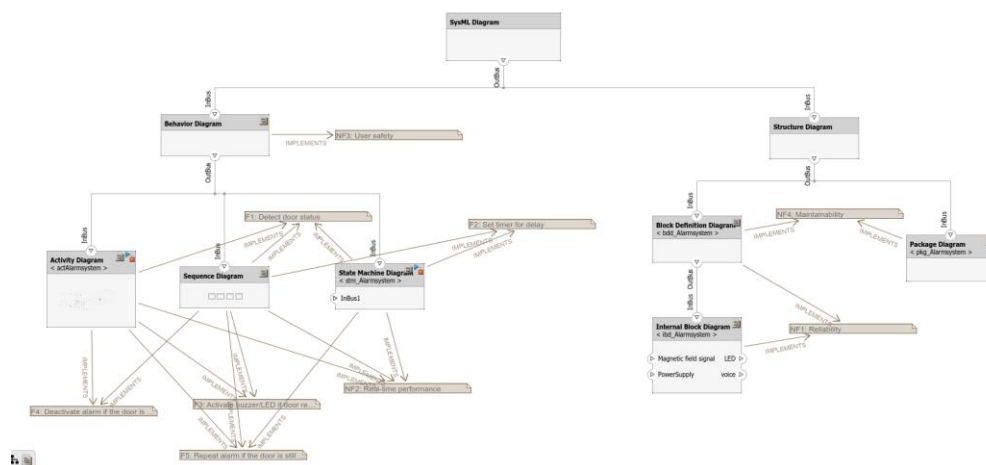
#### During the Alarm (5 seconds):

- The system continues to monitor the door state:
  - If the door is closed during the alarm:
    - The LED and buzzer are deactivated.
    - The system resets and returns to monitoring mode.
  - If the door remains open:
    - The 5-second alarm cycle completes.
    - The LED and buzzer deactivate temporarily, but the system prepares for a new alarm cycle (repeats every 10 seconds).

#### System Stop:

- The user can turn off the key at any time to stop the system.
- All components (sensor, timer, LED, buzzer) are deactivated.

#### 4.3. Link between diagrams and requirements.



## 5. Code Implementation

### 5.1. Setup

#### 5.1.1. Configuration

##### Microcontroller:

- STM32F0xx series is used for this project due to its low power consumption and integrated peripherals.

##### Hardware Components Configured:

1. **Reed Switch:**
  - Pin: GPIO\_PIN\_5
  - Port: GPIOB
  - Purpose: Detects the state of the door (open or closed).
2. **LED Indicators:**
  - Green LED: GPIO\_PIN\_6, GPIOB (indicates normal conditions).
  - Red LED: GPIO\_PIN\_7, GPIOB (indicates alarm conditions).
3. **Timer (TIM1):**
  - TIM1 Channel 3 is used for time-dependent functionalities, including delays for alarms and state transitions.
4. **I2C Communication:**
  - Peripheral: I2C2.
  - Purpose: Communication with the LCD module.
  - Slave Address: 0x4E.
5. **Alarm System Logic:**
  - Uses the reed switch to trigger the state of the LEDs and buzzer.
  - Delays are controlled using the timer configuration.

#### 5.1.2. Library Integration

The following libraries are integrated into the project for seamless operation:

##### 5.1.2.1. I2C-LCD.h

This library enables communication between the microcontroller and the LCD module via the I2C interface.

##### Functions:

- `lcd_init()`: Initializes the LCD in 4-bit mode with a specific function set, display control, and cursor control.
- `lcd_send_cmd()`: Sends commands to the LCD (e.g., clear screen, set cursor position).
- `lcd_send_data()`: Sends data (characters) to the LCD for display.
- `lcd_put_cur(row, col)`: Sets the cursor position on the LCD screen.
- `lcd_clear()`: Clears the LCD screen by writing blank spaces to the entire display.
- `lcd_send_string(str)`: Sends a string to be displayed on the LCD.

**Code Integration:** The `i2c_lcd.h` library interacts with the I2C2 handler (`hi2c2`) for sending commands and data.

##### 5.1.2.2. Timer

**Purpose:** This configuration utilizes TIM1 Channel 3 for timing-dependent operations such as:

1. Controlling the buzzer and LED blinking intervals.
2. Generating precise delays during the alarm cycle.

##### Code Highlights:

- The `HAL_TIM_Base` library is used for timer initialization and interrupt handling.
- The `HAL_TIM_MspPostInit()` function ensures proper timer initialization after base configurations.

### 5.1.3. Equations

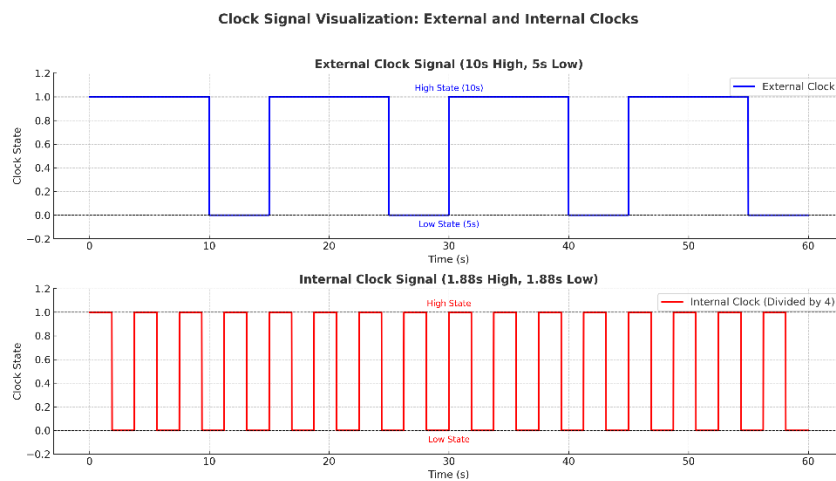
For the timing of the alarm system:

- **Buzzer ON Delay:** 5 seconds (controlled by TIM1).
- **Buzzer OFF Delay:** 10 seconds (repeats until the door is closed).

**Equation for Timer Delay:**

$$T_{delay} = \frac{PSC \times ARR}{f_{clk}}$$

- PSC: Prescaler value.
- ARR: Auto-reload register.
- Fclk: Clock frequency of the timer.



## 5.2. Code Explanation

### 5.2.1. Code Overview

#### 1. Initialization and Setup

- **HAL\_Init():** Initializes the HAL library for hardware abstraction.
- **SystemClock\_Config():** Configures the system clock settings.
- **Peripheral Initialization:**
  - **GPIO:** Sets up the input/output pins for LEDs and the reed switch.
  - **I2C:** Configures communication with the LCD.
  - **TIM1:** Configures the timer for periodic interrupts and PWM for the buzzer.

#### 2. Main Functionalities

- **Buzzer Activation:**
  - Uses a timer interrupt to control the buzzer and LED behavior.
  - Alternates between active and inactive states based on timing logic.
- **LCD Display:**
  - Displays system status ("ON" or "OFF") based on the reed switch state.

- **Reed Switch:**
  - Monitors the state of a door or window and updates the system accordingly.

### 5.2.2.Key Functions

#### **activate\_buzzer()**

- Starts the buzzer using PWM and turns on the red LED to indicate an alert state.

#### **deactivate\_buzzer()**

- Stops the buzzer but keeps the red LED on, signaling the system is still active.

#### **set\_system\_state(uint8\_t is\_open)**

- Updates the LCD display and LED status:
  - **System ON:**
    - Displays "System is ON."
    - Turns on the red LED and off the green LED.
  - **System OFF:**
    - Displays "System is OFF."
    - Turns on the green LED and off the red LED.

#### **handle\_buzzer\_logic()**

- Manages the buzzer's timing:
  - Activates for 10 seconds.
  - Deactivates for 5 seconds, then repeats until stopped.

#### **handle\_sensor\_state()**

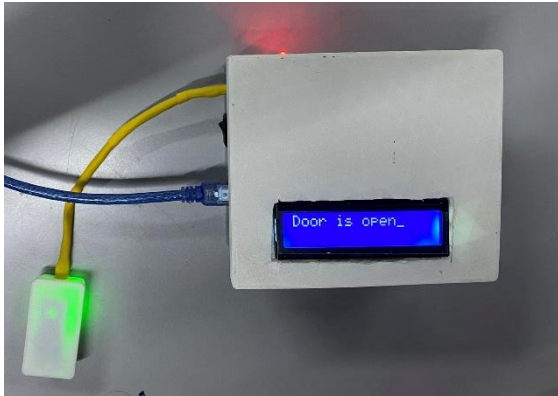
- Reads the reed switch state:
  - **Open:** Sets the system to "ON."
  - **Closed:** Sets the system to "OFF."

#### **HAL\_TIM\_PeriodElapsedCallback()**

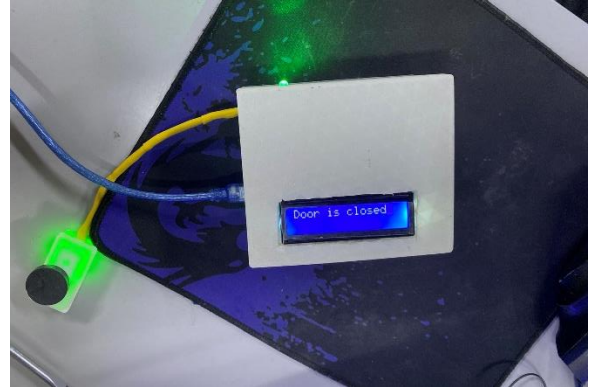
- Triggered by the timer interrupt.
- Calls handle\_buzzer\_logic() to manage the buzzer's behavior.

## 6. Results

When the system is activated:



When the system is Shut down





## **7. Summary**

### **7.1. Recap of Research Goals and Achievements:**

The primary goal of the project was to develop an efficient alarm system using the STM32F0 microcontroller. The system aimed to monitor door activity through a reed switch and magnet, provide visual feedback with an LED, and sound an alarm under specific conditions. The system successfully:

- Detected door status changes.
- Implemented a timed delay of 10 seconds after the door opens before triggering an alarm.
- Enabled a buzzer to sound in cycles (5 seconds ON, 10 seconds OFF) until the door is closed.
- Incorporated modular code and diagrams (sequence, use case, BDD, IBD, and package diagrams) for clear design and functionality.

### **7.1. Challenges Encountered and Their Solutions:**

- Sensor Sensitivity Issues: Initial inconsistencies in reed switch detection were resolved by fine-tuning the pull-up resistor values and optimizing the input pin configuration.
- Timing Delays: Interrupts and timer peripherals were used to achieve precise timing intervals for delays and buzzer operation.
- Sensor Malfunctions: During testing, the reed switch, experienced mechanical or electrical failures, disrupting system functionality. This was resolved by keeping spare components for immediate replacement to minimize downtime. Investigating and addressing root causes such as improper handling, wiring errors.
- Code Organization: Ensured modular and reusable code through structured use of C programming conventions and standardized embedded system design techniques.

### **7.2. Potential Future Enhancements:**

- Wireless Notification: Integrate a Wi-Fi or Bluetooth module to send notifications to a smartphone if the alarm is triggered.
- Battery Backup: Add a battery backup system to ensure functionality during power outages.
- Advanced Features: Incorporate additional sensors, such as motion detectors, for broader security coverage.
- Mobile App: Develop a mobile application for remote control and monitoring of the system.
- Machine Learning Integration: Use algorithms to detect unusual patterns in door activity and alert the user of potential security threats.

## 8. References

- [1] "SysML Diagram Tutorial | SysML.org," SysML.org. <https://sysml.org/tutorials/sysml-diagram-tutorial/>
- [2] "Block Definition Diagram - an overview | ScienceDirect Topics," www.sciencedirect.com. <https://www.sciencedirect.com/topics/computer-science/block-definition-diagram>
- [3] "Internal Block Diagram - an overview | ScienceDirect Topics," www.sciencedirect.com. <https://www.sciencedirect.com/topics/computer-science/internal-block-diagram>
- [4] "Package Diagram | Introduction, Elements, Use Cases and Benefits," GeeksforGeeks, Nov. 22, 2023. <https://www.geeksforgeeks.org/package-diagram-introduction-elements-use-cases-and-benefits/>
- [5] Lucidchart, "UML Use Case Diagram Tutorial," Lucidchart.com, 2019. <https://www.lucidchart.com/pages/uml-use-case-diagram>
- [6] "Activity Diagram - Activity Diagram Symbols, Examples, and More," Smartdraw.com, 2022. <https://www.smartdraw.com/activity-diagram/?srsltid=AfmBOopnx1JYfkDJyf3CRlk79Bt1jLEZfFrN5pu0UXcX4JBrpHBkp0zo>
- [7] "System Composer," www.mathworks.com. <https://www.mathworks.com/products/system-composer.html>
- [8] "Reed Switch Sensor Module," Components101. <https://components101.com/modules/reed-switch-sensor-module>
- [9] "Rocker Switch DPST 4 Pin ON-OFF Black Button Black Housing - KCD1-4-201," Evelta Electronics, 2024. [https://evelta.com/rocker-switch-dpst-4-pin-on-off-black-button-black-housing-kcd1-4-201/?srsltid=AfmBOopPovz-2LRdU4bcbfjHL7pby\\_6bidDtXDJH3MQIG4u9H-JGY\\_Cq](https://evelta.com/rocker-switch-dpst-4-pin-on-off-black-button-black-housing-kcd1-4-201/?srsltid=AfmBOopPovz-2LRdU4bcbfjHL7pby_6bidDtXDJH3MQIG4u9H-JGY_Cq)
- [10] ArduinoModules, "KY-012 Active Buzzer Module," ArduinoModulesInfo, Sep. 26, 2016. <https://arduinomodules.info/ky-012-active-buzzer-module/>
- [11] alldatasheet.com, "STM32F0DISCOVERY PDF," Alldatasheet.com, 2016. <https://www.alldatasheet.com/datasheet-pdf/view/464193/STMICROELECTRONICS/STM32F0DISCOVERY.html>

## Appendices

<https://github.com/yassinmtibaa/Magnetic-Door-Window-Sensor.git>