Bird Detection System for Airports

IoT Class Project

Antonio Campus
ID 70/90/00344

Nicola Deidda
ID 70/90/00358

Master Degree in Computer Engineering, Cybersecurity and Artificial Intelligence
Faculty of Engineering and Architecture
University of Cagliari
Academic Year 2023-2024

Contents

List of Figures

List of Tables

1 Problem overview

The issue of bird strikes, also referred to as bird-aircraft collisions or bird collisions, is a crucial aspect of airport runway security.
The potential risks that such collisions pose to aviation safety cannot be overstated: bird strikes can lead to severe consequences, including damage to aircraft, injuries to crew members and passengers, and even loss of life.

As such, airport authorities and airline operators alike must take proactive measures to minimize the risk of bird strikes.
This may involve implementing comprehensive bird control strategies, such as habitat modification, flock dispersal, or sonic repellents.
Additionally, regular inspections of the runway area can help to identify and manage potential hazards.

Aviation safety is a critical concern for the industry, and bird strikes have been identified as one of the potential hazards.
Even though smaller birds may seem harmless, they can cause significant damage to an aircraft's critical components, particularly the engines.
Such damage may lead to engine failure, which may further result in emergency landings or accidents. Therefore, it is essential to recognize the potential risks of bird strikes and take necessary preventive measures to mitigate the risks.

2 Anlysis Phase

In this section, first thing first the reference scenario will be defined, to clearly state where the system will operate.
Then, an in-depth analysis of the current state-of-the-art for the bird detection task is required: this will help define what are, currently, the pros and cons of the used technologies.
From this analysis, it will be possible to define the functional and non-functional requirements of the new system, which will be proposed at the end of this section.

2.1 Reference scenario

Being the bird strikes an issue related to airport management, the system is intended for airport personnel who hold the responsibility of ensuring the safety of runways: the categories of personnel encompassed by this system include air traffic controllers, runway maintenance staff, and wildlife control teams.
The number of users involved in the usage of this system may vary, contingent on the size of the airport. Nevertheless, it usually involves multiple personnel.
The personnel involved in this system must be cognizant of the problem and possess the requisite skills to tackle it.

Bird strikes happen on airports' runways, meaning that the system must be specifically designed for deployment in outdoor settings: it must operate effectively under diverse environmental conditions, including varying weather conditions such as rain, snow, fog, differing times of day, and potentially challenging lighting conditions.

As discussed in the Introduction, bird strikes are critical for aircraft safety and security, and could lead to heavy damage: the system is expected to perform efficiently during both routine operations and emergencies, meaning that it must be possible to operate it also with high densities of aviation or aircrafts approaching the runways.
However, it is noteworthy that the airport's security practices involve strict checks: these are presumed to safeguard the on-field sensors from intentional damage.

Connectivity is another critical factor that must be taken into account when considering the implementation of this system.
To this end, the existing communication infrastructure is supposed to be leveraged to minimize latency and eliminate noise, that could interfere with the transmission of data.

Summarizing, the present context is characterized by stressors such as inclement weather conditions, elevated air traffic volumes, and the necessity for uninterrupted and dependable operation, that must not interfere with the capabilities of the system.

2.2 As-is systems analysis

With the reference scenario well defined, it is now possible to analyse the current state-of-the-art, paying attention to the stressors and characteristics emerged above.
The issue of bird strikes, although posing a significant threat to aircraft safety, is mitigated by the utilization of various technologies, which are often combined for optimal results.
Despite pilots and airport personnel being trained to handle such scenarios, it is essential to rely on these technologies to ensure comprehensive and accurate outcomes.
In addition, advancements in the field of machine learning and artificial intelligence are being deployed to enhance the identification capabilities of airport systems, which, in turn, facilitates prompt decision-making by airport personnel: real-time decision-making is now possible due to the advanced identification capabilities of these systems.

Currently, the state-of-the-art solutions implemented can be categorized as follows:

- **Radar-based Systems**: Utilizing advanced radar technology, it is possible to accurately detect the presence of wildlife.
- **Camera-based Systems**: camera systems in conjunction with advanced image processing techniques are used to accurately detect and identify birds on the runways.
- **Acoustic Sensors**: sound detection technologies can be employed to accurately and efficiently identify various species of wildlife.

In the following subsections, an exhaustive analysis of these technologies will be provided, together with their advantages and disadvantages.

2.2.1 Radar-based Systems

The implementation of radar technology in tracking and identifying birds in the vicinity of airports has been of great significance to airport authorities and pilots.
These systems provide valuable information that aids in the avoidance of bird strikes and other aviation hazards, reducing the potential risk of accidents and fatalities.

There exist different typologies of this family of systems, the one specifically designed to deal with this problem is referred to as the "Bird Radar System".

Bird Radar Systems are specifically intended to detect and track birds that are close to airports.
These radars are capable of distinguishing between birds and other objects, providing instant information regarding the location, altitude, and direction of bird flocks.
Advanced bird radar systems rely heavily on algorithms to analyze the collected data and alert airport personnel in real time.

Besides that, other radar-based systems can be employed, implementing flexible architectures to enhance the overall bird-tracking capabilities.

Then, radar data can be combined with other sources of information, such as weather data, to improve the understanding of bird behaviour and movements: these advanced analytics and data fusion techniques help in creating a comprehensive situational awareness picture for airport operators.
In addition, radar-based systems are integrated with automated warning systems that can alert air traffic controllers, pilots, and airport personnel when bird activity raises a potential risk.

Advantages Radar-based systems provide early detection of bird activity, allowing airports and air traffic controllers to take preventive measures well in advance.
Then, data can be integrated with other sources to provide a complete overview of the airport.
These systems can also be configured to suit the specific needs and characteristics of different airports, taking into account factors such as size, geographic location, and bird species prevalent in the area.

Disadvantages Radar systems may sometimes generate false positives, detecting non-bird objects as potential threats. This can lead to unnecessary alerts and interventions.
With that, some radar systems may have limitations in accurately determining the altitude of detected objects and suffer from adverse weather conditions, such as heavy rain or snow.

From a monetary budget point of view, the installation and maintenance of radar systems are highly expensive, so smaller airports with limited budgets may find it challenging to implement sophisticated radar technologies.
Finally, could be challenging to integrate data from radar systems with other systems and databases.

2.2.2 Camera-based Systems

Camera-based surveillance systems depend on visual imaging technologies for the monitoring and identification of avian activity in the airport vicinity.
These systems employ cameras to acquire either still images or video recordings, and subsequently, the captured data undergoes analysis for bird detection and tracking.

Advantages Camera-based systems provide visual confirmation of bird activity, allowing for identification of bird species and assessment of the size of bird flocks: this is more true if high-resolution cameras are used.
In addition, images and videos can be used to analyze bird behaviour, such as flock patterns and flight trajectories, providing insights into potential risks and enabling the use of predictive systems.
Typically, such systems exhibit cost-effectiveness in comparison to radar counterparts, rendering them a viable choice for enhancing runway security at smaller airports.
Then, Camera-based systems can be integrated with other technologies, such as learning to improve the accuracy of the bird detection task.

Disadvantages Cameras need a clean line of sight to detect birds: obstructions, light conditions and bad weather may limit their effectiveness.
Then, if the cameras-based system is improved with machine learning algorithms, the analysis of large volumes of image or video data in real time can be computationally challenging.
Finally, proper maintenance is crucial for cameras to function optimally and prevent image quality degradation from environmental exposure.

2.2.3 Acoustic Sensors

Acoustic sensors operate based on sound wave detection to identify and monitor avian activities.
These sensors discern distinctive acoustic signatures associated with bird vocalizations or wing beats. Subsequently, the acquired data undergoes analysis to ascertain the presence and spatial coordinates of avian entities.

Advantages Acoustic sensors offer non-visual bird detection, providing an alternative to radar and camera-based systems: this can be particularly useful in low-visibility conditions or during the night.
Acoustic sensors exhibit an impressive capacity to encompass expansive areas,

making them an optimal selection for surveilling vast airport spaces and identifying avian activity across a broad spectrum.
Finally, the maintenance requirements of these systems are comparatively lower than other technologies, attributed to the absence of moving parts and resilience to adverse weather conditions.

Disadvantages The existence of ambient noise, such as that emanating from aircraft engines, can compromise the accuracy of the sensors, potentially resulting in false positives or missed detections.
Similarly, adverse weather conditions, such as strong winds or heavy rain, can trivially impact the efficacy of acoustic sensors, potentially diminishing the effectiveness of the system.
In the end, the absence of images or location information afflicts the capability of the system to monitor and track possible patterns.

2.2.4 Summary

The salient features of the aforementioned technologies are succinctly summarized in the accompanying Table 1.

Category	**Advantages**	**Disadvantages**
Radar-based Systems	- Long-range detection - Effective in various weather conditions	- Difficulty in accurately identifying small objects - Limited ability to distinguish between different bird types
Camera-based Systems	- Visual confirmation - Integration with image recognition for species identification	- Highly dependent on visibility conditions
Acoustic Sensors	- Ability to detect bird sounds	- Highly vulnerable to ambient noise

Table 1: Summary of Advantages and Disadvantages of Wildlife Detection Systems

2.3 System requirements

The analysis of the current state-of-the-art allows for defining the requirements the new system must satisfy to perform well in the stated scenario. Last but not least, the satisfaction of these requirements makes the new system able to compete with the ones already in place.

2.3.1 Functional requirements

Below, the Functional Requirements are listed: these describe the capabilities the new system must have.

FR1: The user can log into the system through an authentication process Each user who wants to operate the system must be authenticated: to this end, each user will be provided with credentials. The user will be distinguished into "privileged users" and "standard users".

Input Data provided by users in an authentication form: username and password.

Output If the authentication process succeeds, access to the web application related to the system is provided. Otherwise, an error message is displayed and the access is rejected.

FR2: The authenticated user is able to monitor relevant events Any authenticated user can analyse the data provided by the system through the web application.

Input Data acquired by the system while operating.

Output Charts and summaries, to provide the user with an overview of the system.

FR3: The privileged user is able to customise the system by adding new modules A privileged user can set up the system with new modules, such as new cameras, to fit its needs.

Input Data provided by the privileged user through a guided procedure.

Output A new module is added to the infrastructure.

FR4: The system is able to take a photo at regular intervals At regular time intervals, the system acquires pictures from the cameras on the field.

Input A trigger signal from a time module.

Output An image from each camera.

FR5: The system is able to detect the presence of birds. Each picture acquired by the cameras on the field must be analyzed by an object detection algorithm able to detect birds.

Input A picture from each camera.

Output For each picture, the result of the detection is returned, it can be True or False.

FR6: The system is able to trigger the protection measures if birds are detected and raise an alert. Each time the object recognition process detects a bird in the image, the corresponding actuators must be triggered to activate the protection measures. Also, an alert must be raised.

Input A trigger signal from the image recognition process.

Output The protection measures are activated and an alert is displayed in the web application.

FR7: The privileged user is able to manually activate the protection measures A privileged user can use the web application interface to trigger the activation of the protection measures, overriding the result of the image detection process.

Input The interaction of the privileged user with the web application.

Output The protection measures are activated.

2.3.2 Non-Functional requirements

Together with the Function Requirements listed above, the new system will have the following Non-Functional Requirements, that aim to improve its capabilities and the overall system quality.

NFR1: The object detection process presents a $< high >$ accuracy The process to detect if an image includes some birds should have at least $< high >$=95% accuracy.

NFR2: It is possible to add any number of devices to the system The system should not impose any limit on the number of devices.

NFR3: It is possible to replace each part of the system without additional complexity The system should be composed of non-complex and standard components, $< easy >$ to replace. Where $< easy >$ = removing the broken component and replacing it with a new one, without additional maintenance required.

NFR4: The user interface is $< simple >$ and $< intuitive >$ The users should be able to reach every part of the web application in a $< simple, intuitive >$ manner = less than five clicks required.

2.4 To-be system analysis

From the requirements analysis gathered before, the following additional key features have been defined.

2.4.1 User-friendly interface

As discussed in the sections above, the analysis of complex data patterns can be a challenging and time-consuming task, while the scenario requires the system to operate also in emergencies.

Then, according to **FR1**, **FR2** and **NFR4**, it is mandatory to implement a modern and user-friendly interface that provides a complete overview of the system while remaining easy to comprehend and secure from unauthorized accesses.

Such an interface streamlines the process of analyzing data patterns by presenting a comprehensive view of the system, making it easier to comprehend the information presented, and supporting the decision-making process of the airport personnel.

In addition, according to **FR7**, the privileged user must be able to manually operate the system: a proper interactive procedure it is required to enforce this requirement.

2.4.2 Machine Learning-powered detection

According to **FR5** and **FR6**, the system must be able to identify the presence of aviaries in the images acquired by the cameras on the field.

The camera-based systems, as mentioned above, are part of the current state of the art, however, taking into account the **NFR1**, they do not guarantee a high accuracy in certain challenging conditions. To enforce this requirement, the proposed system will be based on cameras, with the captured images being fed into a machine-learning model to automate the detection process and enhance accuracy.

This approach will enable the system to operate with a high degree of efficiency and effectiveness, reducing the need for manual intervention and minimizing the potential for errors.

2.4.3 Reduced costs

From the "as-is" analysis comes how high could be the cost of implementing an effective bird-detection system, making it difficult for small entities to afford it.

Due to the utilization of cost-effective components, the new system will significantly reduce expenses, thereby rendering it accessible for small airports, which were previously unable to acquire such technology due to its high price point.

Despite the lower cost, the system's proficiency will remain unchanged, ensuring that its functionality and accuracy live up to the standards required by larger entities.

2.4.4 Higher scalability

The successful deployment of the new system lies in its ability to accommodate diverse scenarios.
The requirements **FR3** and **NFR2** define as a crucial aspect of this system the capacity to support the addition of new modules without necessitating complex, supplementary configurations.
Thus, the system will be built with flexibility in mind, allowing for seamless integration of new modules. As such, the system will be able to adapt to changing business requirements and remain relevant in the long term.

2.4.5 Easy maintenance

Upon analyzing previous systems, it has been determined that maintaining large-scale infrastructures can be a highly complex process.
Therefore, according to **NFR3**, the new system must be designed with a higher degree of modularity to enable easy maintenance procedures.
This will not only streamline the maintenance process but also reduce the possibility of unforeseen disruptions and associated costs.

2.5 Available hardware

Before well-defining the new system proposal, an analysis of the available technologies, both in terms of hardware, software and communication technologies is required.
Keeping into account the requirements stated above, this analysis is presented in the following sections.

2.5.1 Development boards

An overview of the available development boards, with their pros and cons, is presented in Table 2.
All these boards could be used to implement the new system, however, due to the requirements in place, the choice must have a proper balance between costs, computation resources and vendor support.
Following, a brief presentation of each one is provided.

Arduino Uno This is a popular and open-source development board designed to build electronic projects.
It features digital and analog input/output pins, USB connectivity for programming, and power and communication protocols.
Arduino Uno is known for its ease of use, versatility, and the large community of users.
Differently from analogous boards, it does not feature any wireless connectivity.

Arduino Nano The Arduino Nano is a compact board, serving as a smaller alternative to the Arduino Uno.
It shares many features with it but comes in a more compact form factor.
It is commonly used in applications where a smaller footprint is desired, such as in embedded systems and projects with limited space.

ESP8266 This is a low-cost, compact Wi-Fi module with an integrated microcontroller. It is widely used for Internet of Things applications.
The limited cost made this board affordable and easy to use, while its connectivity allows for wireless communications.

Raspberry Pi The Raspberry Pi is a single-board computer that features a compact design and is equipped with USB ports, HDMI output, audio jacks, GPIO pins, and an SD card slot for storage.
It presents higher computational capabilities than the board from Arduino families, allowing for extensive customizations.

From the pros and cons analysis in Table 2, the chosen development board is the ESP8266.
While the lack of computational resources could lead to the choice of a Raspberry

Board	Pros	Cons	Cost
Arduino Uno	– Compact form factor – Large community support and available extensions, allowing extensive customization to fit different scenarios – Stable and reliable, based on a well-known architecture	– Limited resources in terms of memory and processing power, which impose the use of extension modules – No built-in wireless module, so a Wi-Fi additional module is required	$25.00
Arduino Nano	– Compact form factor	– Limited I/O pins, compared to Arduino Uno. It can satisfy the requirements of basic projects but is unfeasible for complex ones. – No built-in wireless module, additional extensions are required	$20.00
ESP8266	– Compact form factor – Computational capabilities similar to Arduino Uno – Built-in wireless and serial connectivity	– Limited number of GPIO pins – Limited amount of memory	$5.00 - $10.00
Raspberry Pi	– Powerful processing capabilities and expandable storage – Built-in connectivity, allowing to have all modules built in the same board – Versatility in the development process, allowing the coexistence of different modules, technologies and architectures	– High Power Consumption: Raspberry must be connected to the power grid or equipped with properly-sized batteries.	$35.00

Table 2: Available boards, with pros and cons

Pi, its customizability and its connectivity features made this board a proper choice for the implementation.

2.5.2 Sensors and actuators

The sensors that can be used in the new system, with their pros and cons, are listed in Table 3.
Instead, the analysis of available actuators is provided in Table 4.
Sensors and Actuators must be chosen keeping in mind the environment where they will operate: airports' runways are noisy places, in terms of sounds and interferences.
In addition, to exploit the already-on-place infrastructure, sensors and actuators must be able to connect to WiFi and operate without the need for additional support.
Last but not least, the power consumption must be constrained.
Also in this case, the requirements must be taken into account: a reliable, cost-effective, easy-to-maintain and camera-based system is mandatory.

From Table 3 the best choices that fit the **FR4** appear to be the OV7670 and the EXP32 camera modules.
While other available sensors present characteristics that do not fit the requirements, such for instance the possibility of high accuracy in image recognition, these two modules feature interesting capabilities, that will be tested to make a considered choice.

Concerning the actuators, the state-of-the-art requires the use of air cannons. While this will be possible during on-field tests, for test purposes, buzzers will be employed.

Experimental test To properly choose one camera module, it is necessary to set some benchmarks.
According to **FR4**, **FR5**, **NFR1** and **NFR3** the following benchmark have been defined:

- **Latency in acquiring the image**: the image must be acquired at regular intervals, meaning that no time extensive computation is allowed.
- **Image quality**: the image must have a good quality to enforce the object detection process, allowing the exploitation of the accuracy of the machine learning model.
- **Module complexity**: the additional complexity must remain as low as possible.

Initially, the OV7670 implementation has been evaluated.
The camera is able to capture good-resolution images, able to be correctly classified by the detection system.

Despite the image quality, access to the camera is made by implementing low-level operations: this increases the software's complexity and degrades its maintainability.
In addition, due to the lack of compatibility, the OV7670 must be connected to an Arduino Uno, meaning that additional connectivity modules are required.
Connecting an ESP8622 WiFi module, the Arduino is able to communicate through WiFI. However, a critical bottleneck remains.
The camera communicates with the Arduino Uno using a serial protocol: during the experiments, a good balance between acquiring latency and image quality has been achieved reading at 500k bps from the UART interface.
While this speed is affordable by a more powerful device, the ESP8266 is not able to deal with it, having a theoretical maximum serial speed of 9600 bps.
Doubling the clock frequency, during the experiments it was possible to reach the serial speed of 1Mbps: despite that, the latency in the communication was excessive to satisfy the requirements.

From this, the experiments moved to the ESP32 camera. This module features built-in WiFi connectivity and higher resources compared to the other one. Being sold with an Arduino-compatible system on a chip, the camera can be directly programmed with high-level instructions.
The experiments show that the image quality is as high as in the previous case, while the latency and complexity drastically decrease.

Summing up, the camera chosen is the ESP32. Due to its characteristics, the devices do not require additional WiFi capabilities, meaning that the devices with the camera will be made only by this module, further reducing the complexity.

2.5.3 Communication technologies and protocols

The analysis of the communication technologies is provided in Table 5, while protocols are listed in Table 6.
From this point of view, the system requires reliable communication technologies, well-standardized and defined, with additional security capabilities, due to the critical nature of the system.

To choose proper communication technologies and protocols to further analyse the reference scenario and the requirements.
Devices must be placed on the runways, meaning that, to avoid intrusive infrastructures, wireless communication must be exploited.
According to **FR1** and **FR2** underline the necessity of a web application, to which the connection must be performed using HTTP protocol.
Then, **FR5** requires the sending of the image to a central server, where the object detection is performed: to achieve this task, HTTP will be employed, due to its security features and well-defined structure. Instead, to satisfy **FR6** and **FR7**, a publisher/subscriber protocol response is better, due to the possibility of activating more devices with one command. For this reason, MQTT has been

chosen.

2.6 System proposal

In the following subsection, the new system will be defined taking into account the choices and the considerations reviewed in the sections above and better explaining some technical details.

Starting from the requirements, modularity and maintainability are two key features of the new system. To enforce them, it is appropriate to separate the different subsystems into individual modules. Applying the "*divide et impera*" approach, it will be possible to independently manage the subsystems, making them almost independent from other nodes and any central entity.

The system will be composed of two modules: one with a camera, and the second with an actuator. In addition, a central entity is required to act as a conjunction between them: this entity must be able to serve the on-field devices and the users that want to access the system.

The module with the camera (the Detector) will periodically acquire an image from the field, then it will send it to a server able to perform image recognition using a machine learning model. As mentioned before, the chosen camera is the ESP32, which does not require any additional modules to perform the assigned task.
The images will be embedded in an HTTP request and then sent to the server: this protocol can exploit an API system, useful for managing the classification task. Its overhead represents for sure a disadvantage, but it is negligible due to the computational capabilities of the camera module and the time interval available to send the image to the server.
With this first device, the system satisfies the requirements **FR4**, **FR5** and **NFR1**. Finally, the extremely simple implementation of the camera satisfies **NFR3**.

The actuator will receive a command from the server when the ML module confirms the bird's presence on the field.
With the command received, the actuator will active its buzzer
As for the Detector, the Actuator will exploit the existing WiFi infrastructure to communicate with the server. As mentioned before, MQTT has been chosen as communication protocol
The communication will be managed by an ESP8266 WiFi module, able not only to communicate over WiFi but also to manage additional hardware components.
With this implementation, requirements **FR6** and **FR7** are managed. As for the Detectors, the simplicity of the system further enforces the **NFR3**.

The system will also implement a web application, allowing the users to analyse

the current scenario and the alerts' history, as requested by **FR2**. According to **FR1**, authentication is mandatory to access the application, meaning that a database is required to properly store their credentials, together with other useful information about the system. Users will be divided into standard and privileged users: the former will access the web application to analyse the alerts and the statistics, while the latter will be also able to manage the system architecture, for instance, adding or removing devices without any kind of constraints.
With these features in place, **FR3** and **NFR2** are accomplished.
All the interfaces in the web application must be designed taking into account **NFR4**.

The web application will interface with the server, providing an updated overview of the system to the users.

The privileged users will also have access to a Telegram bot, which will provide security features related to the web application, such as OTP codes for privileged actions.

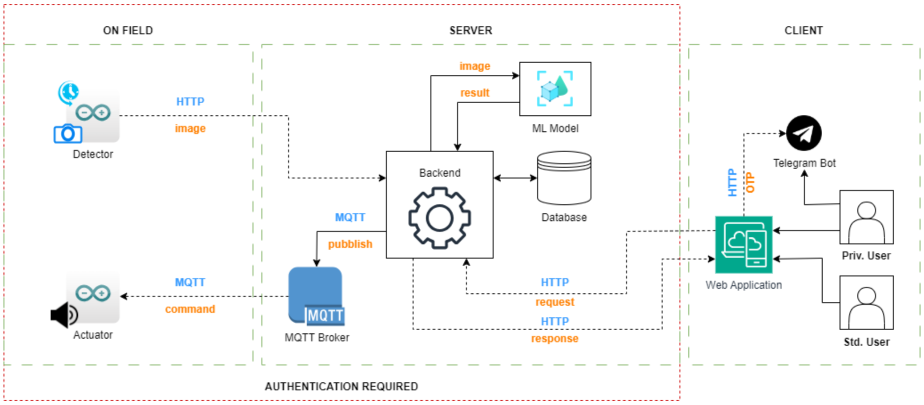

Figure 1: Architecture overview: the dashed arrows represent WiFi communications, while the solid arrows are the actions and the exchange of data in physical/logical channels. On the arrows there are in blue the communication protocol, then in orange, the data exchanged

3 Design phase

This section presents an in-depth analysis of the design aspects of the proposed system, covering hardware, software, and overall infrastructure.
The aim is to provide a detailed account of the technical aspects of the system's design and to highlight the key components that make it work seamlessly.

In terms of hardware, the system incorporates components previously analysed and chosen: the goal is to make them interact together limiting the system's complexity and enforcing the requirements.
The software aspects of the system are designed to guarantee robustness, focusing on modularity and flexibility to accommodate future enhancements.

The schema proposing the whole architecture overview is presented in Figure 1.

There, for the sake of clarity, the overall architecture has been divided into three logical areas: On Field, Server, and Client.

- On field: this area includes the devices on the airport's runways, such as the Detectors and the Actuators
- Server: it is the link between the different components of the system. Each component inside this area must be considered as a software module embedded in the server's application program.
- Client: this area contains the user-side technologies, such as the web application and the Telegram bot.

It must be underlined that the interactions in the first two areas must be authorized through credentials and token exchanges.

About the communication protocols As previously mentioned, the subsystems are required to communicate in different manners, according to their tasks.

The HTTP protocol is well-suited for the communication between the Detectors and the Server because of its stateless nature and its seamless integration with all kinds of web technologies. These features allow each Detector to send images independently from their history or the server's history while guaranteeing possible future improvement in the implementation.
In addition, the encryption capabilities allow for secure communication, fundamentals when dealing with sensitive data such as pictures from the airport's runways.

Instead, for the server-actuator communication MQTT has been chosen: its publisher-subscriber nature helps manage a large amount of Actuators in the same area with only one message.
MQTT's topics will represent different areas of the scenario: an activation command, will activate each actuator in the area.

3.1 "On field" systems

This section provides an in-depth analysis of the Detector and the Actuator implementations.
These two embedded systems are the interface between the system and the real world, meaning that they are exposed to weather conditions and adversary situations: their robustness, in terms of hardware and software, must be guaranteed.

Each subsystem will be analyzed in terms of hardware and software implementation, as well as the communication procedure with the central server.

3.1.1 The Detector

The Detector is the subsystem with the task of periodically acquiring an image from the runway, and sending it to the server for the bird recognition task.

The algorithm Figure 2 proposes a flowchart about the Detector's implementation.
When the ESP32-Camera module is booted, the setup phase starts, trying to connect to the WiFi Network. If this operation succeeds, then the camera is initiated.

The camera's initiation aims to set the connection between the two SoCs: if an error occurs during this phase, the whole module must be restarted. Instead, if the camera is correctly initialized, its configuration parameters can be

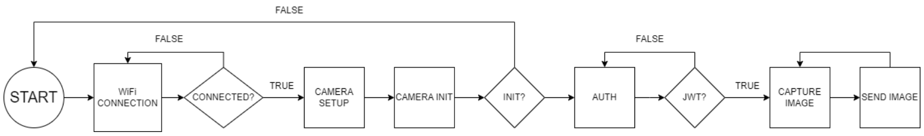

Figure 2: Flowchart of the Detector's implementation

set: these parameters include the contrast, the brightness and the colour correction settings.
The camera initiation phase is the most critical: its failure means that the camera will not be able to satisfy its task, then requirements.

When connected and initiated, the ESP32 module authenticates itself to the server sending an HTTP request with its credentials. These credentials are stored inside the software.
If the sent credentials are correct, the server provides the Detector with a secure token. This procure has been schematized in Figure 3.

At the end of the setup phase, the camera is ready to acquire an image. The image is acquired and sent to the server periodically, embedding it in an HTTP request, together with the secure token previously obtained.

Hardware wirings The Detector does not require particular connections to other hardware modules. The ESP32 Camera module already come out with its subsystems connected:

- The camera, an OV2640, is connected to its SoC through a bus.
- The camera's SoC is connected to the development board by connecting their pins one to one.

The ESP32 Camera module's specifications are listed in Table 7. Instead, its schematic is provided in Figure 4.
About the different components of the ESP32 module, they are shown in Figure 5.
The overall hardware design appears to be compact, robust and simple while allowing for further customization.

3.1.2 The Actuator

The Actuator must scare birds in the runways either when a Detector locates them, or when an authorized operator activates it.

The algorithm Figure 6 describes the algorithm of the actuator.
When the device starts, visual feedback is provided through an LED. As in the Detector case, also the Actuator starts setting its wireless connectivity. Then,

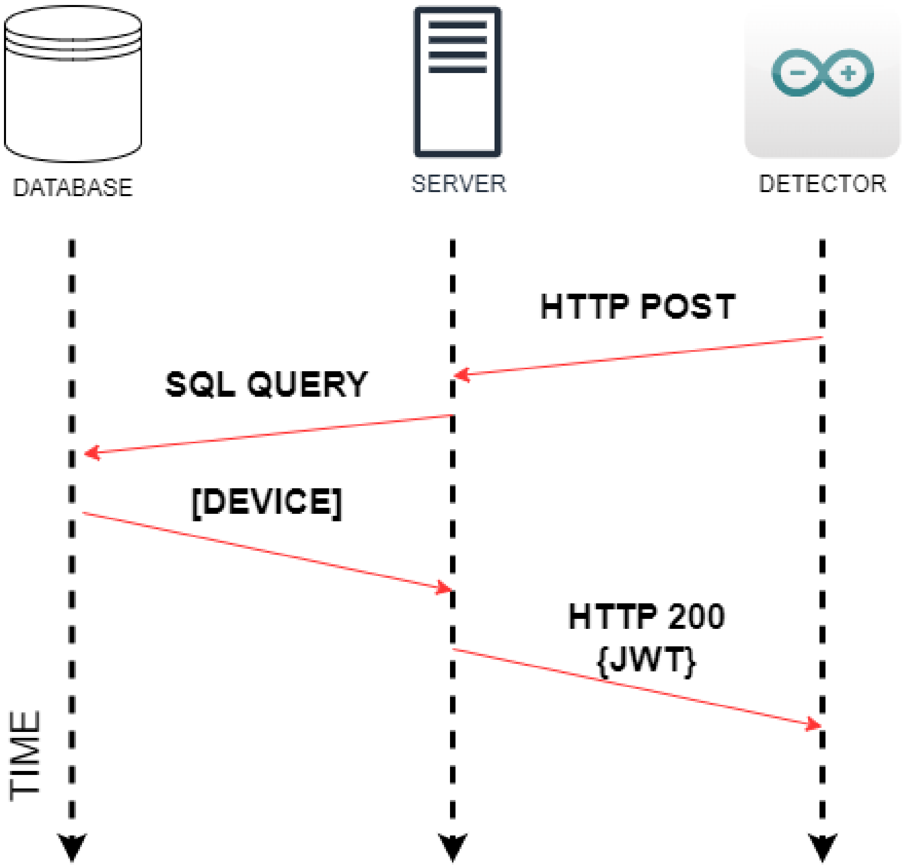

Figure 3: Detector authentication procedure

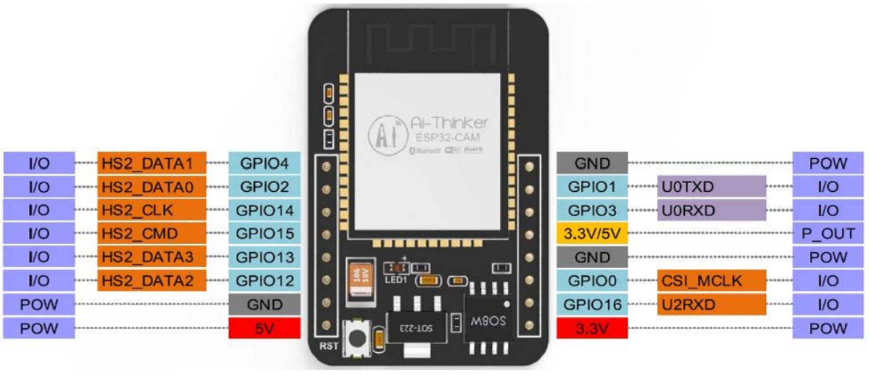

Figure 4: ESP32 schematic with pins

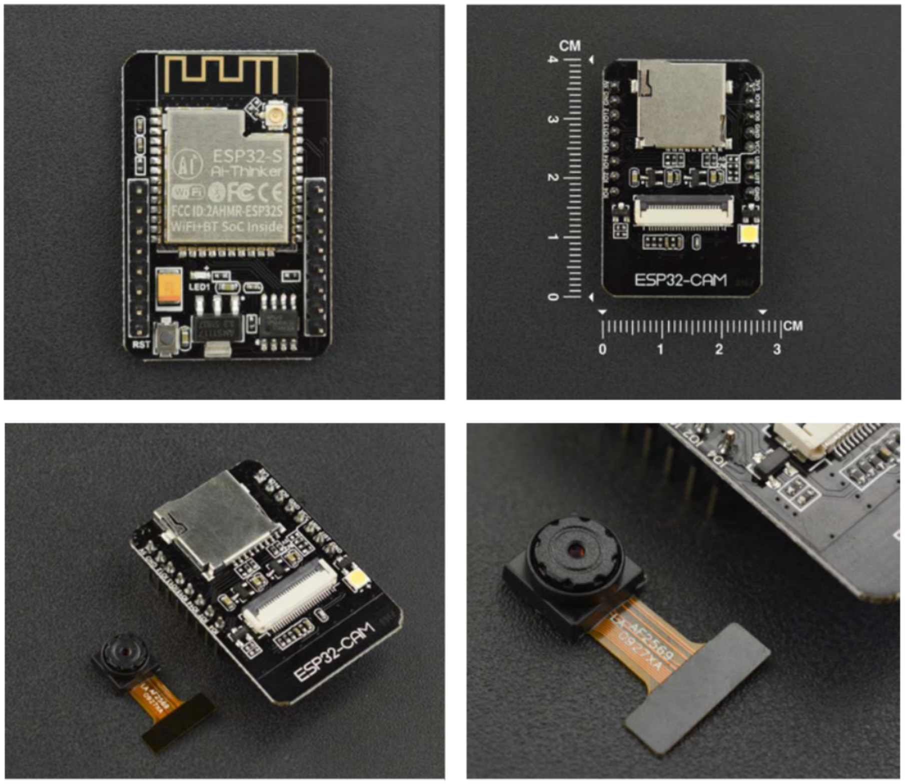

Figure 5: Detailed image of ESP32 module's components

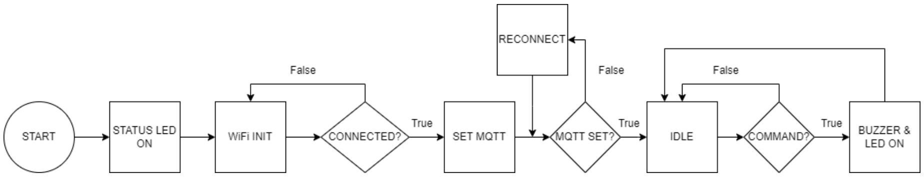

Figure 6: Flowchart of the Actuator's algorithm

it subscribes to the MQTT's topic corresponding to its area and will remain in an idle status while a command is received.
Receiving a command on the topic means the Actuator must activate its buzzer: it will remain active for a time period. This time-limited activation will not limit the effectiveness of the device: if the target flock remains on the runway, the detectors will, again, detect it the actuators will receive another activation command.

Hardware wirings Wirings for the Actuator module are described in Figure 7.
The circuit is composed by:

- The ESP8266 WiFi module, is able both to connect to the server and manage other hardware components.
- A buzzer, is used to emit sounds to scare the birds in the area. As previously mentioned, this module may maybe too simple but helpful for testing purposes.
- Two LEDs: one active when the ESP8266 is configuring itself, and the other activates when an activation command is received.

3.2 Server

As it is possible to see from Figure 1, the Server is made by several subsystems that cooperate to manage the incoming data and requests.

These subsystems are connected by a server, which is in charge of managing the available routes and exposing the APIs. While it has been denoted that HTTP-enabled software is required, it was not defined the specific framework employed.
The goals are still maintainability, simplicity and robustness: the best compromise found is to implement a Flask server.
Flask is the core component of the server program. With it, the server includes:

- The Machine-Learning model, to perform the bird-recognition task
- The database, to store the application's data

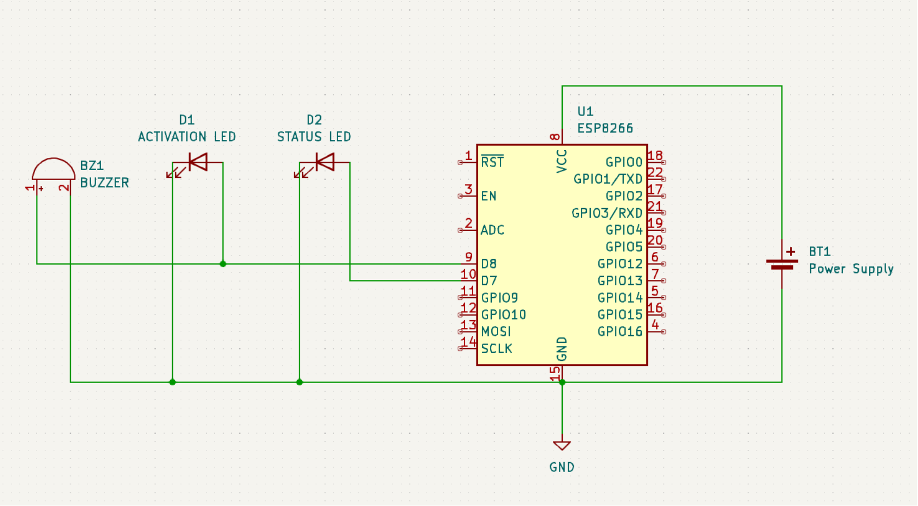

Figure 7: Schematic of the Actuator's circuit

- An MQTT Broker, to manage the MQTT exchanges between the server and the actuators.

3.2.1 The Flask application

Flask is a micro web framework written in Python that is designed to be lightweight and flexible. It provides the necessary tools and features to build web applications by simplifying the process of handling HTTP requests, routing, and managing application logic.

The Flask application is available on server port 5000. The APIs exposed, are described in Table 8. Each endpoint, except the `/api/users/login` one, must be accessed using the authorization token received with the login's successful response.

Securing the APIs accesses Data stored in the application and accessed throw the APIs are considered critical, as critical as the operation that the server can perform: for instance, adding or removing a new device.
The login is mandatory before requesting any other resources to guarantee a proper security level.

The login procedure provides the user with a secure token, containing some useful data, then signed with a secret key.
This data can be accessed by the application, while their integrity and authenticity are enforced through the signature.
This technology is referred to as JWT (JSON Web Token). Figure 8 summarizes

the authentication procedure of a client that wants to access the APIs.

3.2.2 The Machine Learning model

The image received from the field is forwarded by the Flask backend to the machine-learning model for the bird-recognition task. In the project, this model is also referred to as "Bird Detector".

The Bird Detector model is based on the "CenterNet" object detection model and trained on the Microsoft Common Objects in Context (COCO) dataset, a widely used benchmark for detection tasks. Its robust infrastructure, together with its high capabilities, satisfies the requirements in terms of accuracy and object detection.
Technically speaking, the key motivation for choosing this model as the basis of our detection algorithm is that "Centernet" focuses on detecting objects based on their center points: this eliminates the need for anchor boxes, making it computationally efficient while preserving accuracy.
The backbone architecture instead, is the ResNet50, a Residual Network with 50 layers, and Feature Pyramid Network. Resnet is a deep convolutional neural network architecture, while FPN helps capture multi-scale features, which is crucial for object detection.

3.2.3 The database

The nature of the data processed by the application necessitates the selection of a Relational Database Management System (RDBMS) that is capable of managing structured data. To this end, SQLite3 has been chosen to simplify the overall architecture and data management.
This open-source, serverless, and self-contained RDBMS does not require a separate server and stores the entire database in a single file, thus optimizing resource utilization.
Additionally, SQLite3 is cross-platform and can be installed within pre-existing infrastructures, thereby augmenting the portability of our system.
The Entity-Relations schema of the database is provided in Figure 9. Instead, information about data types is provided in Table 9.

3.2.4 The MQTT Broker

As previously stated, in situations where there are multiple devices in a given area, it becomes necessary to send identical commands to multiple actuators. This requirement can be fulfilled by employing the publisher/subscriber protocol such as MQTT. The use of this protocol ensures that the message is delivered to all the intended actuators simultaneously, without the need for individual command delivery.
Employing the publisher/subscriber protocol such as MQTT not only simplifies the process of sending commands to multiple actuators but also significantly

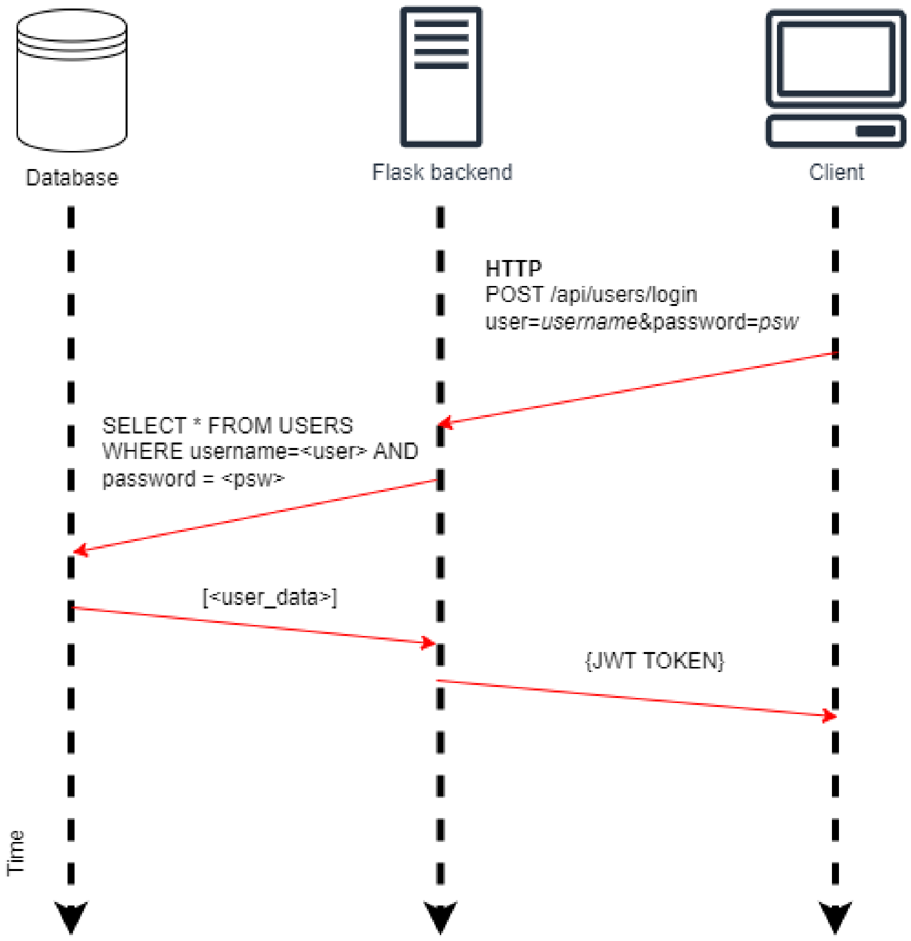

Figure 8: Schema of the authentication procedure. If the authentication succeeds, the client obtains the secure token

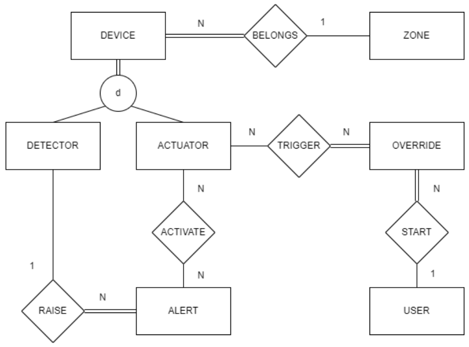

Figure 9: Database's schema

reduces the probability of errors in the command delivery process.
MQTT is a lightweight protocol, suitable for constrained environments, such as actuators, that supports persistent connections, allowing clients to stay connected to the broker even when they are not actively sending or receiving messages. This last feature allows the actuator to remain connected to the broker while waiting for commands.
Finally, MQTT provides the scalability features the system requires, being able to serve multiple subscribers on multiple topics.
Figure 10 schematizes the MQTT broker behaviour when a Detector belonging to a specific zone sends an image that triggers the ML model.

About the Zones Zones are logical sets the system uses to aggregate different devices together.
Each zone can have as many detectors and as many actuators as the users want. When the ML model detects birds in the picture, all the actuators in the Detector's zone will be activated.

3.3 Client

From the client side, the system distinguishes two different types of users: privileged users and standard users.
While the paragraphs below give an overview of these categories, their permissions are summarized in Table 10.
It must underlined that in the table, permissions marked with * required a multi-factor authentication with the Telegram Bot.

Standard users These users can access the web application to overview the system situation, analysing the logs and its structure, but without editing it.

Privileged users These users are allowed to act as standard users. In addition, they are able to manage the system infrastructure, add and remove

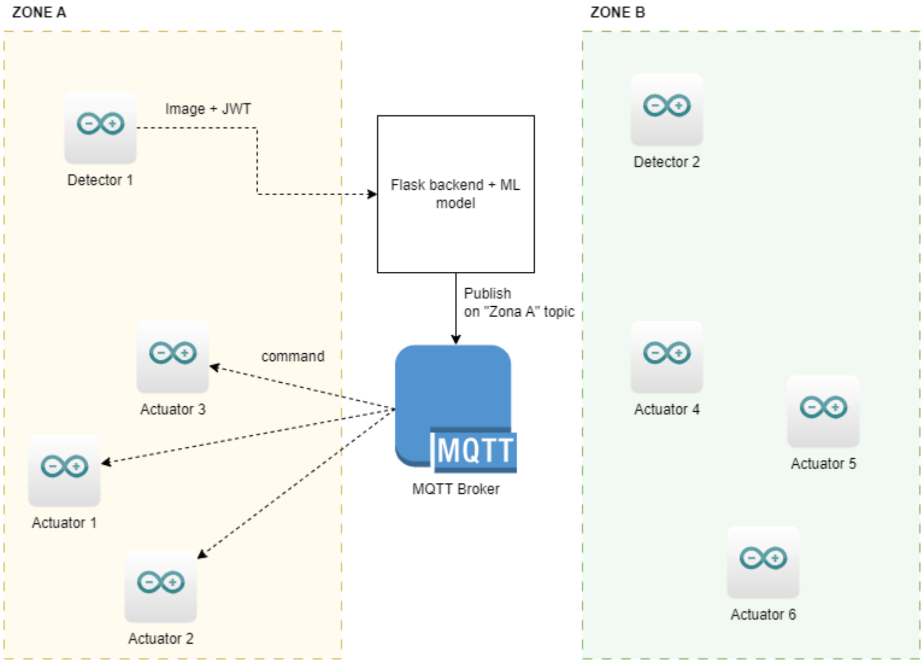

Figure 10: MQTT broker schema: the activation command is received only by the actuators subscribed to the interested zone, got from the detector which sent the image.

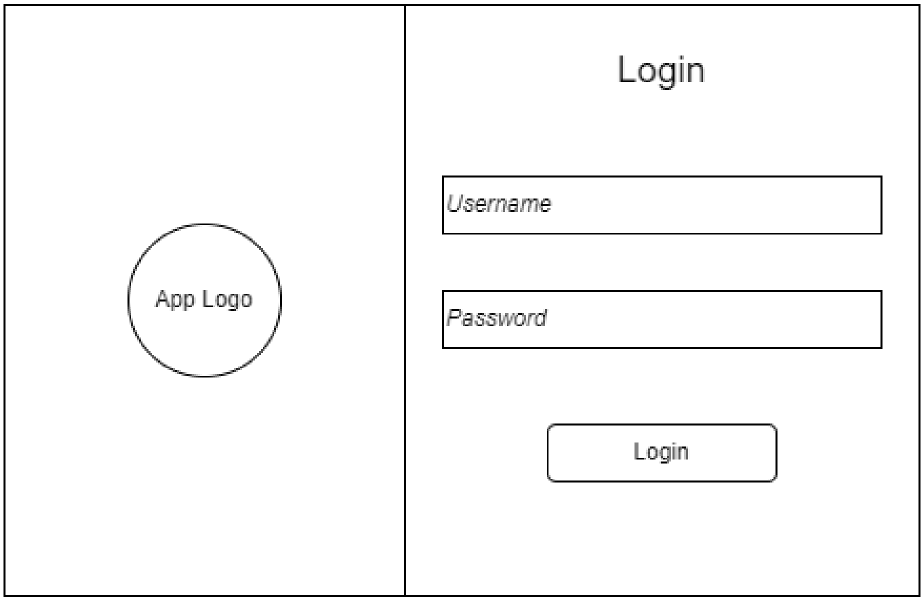

Figure 11: Schema of the login form provided while attempting to access the web application

components, and set up new users.
To enforce this operation's security, privileged users have access to the Telegram bot, which provides them with OTP codes.

3.3.1 The web application

The web application provides an interactive and intuitive interface between users and the system. These characteristics are mandatory to satisfy the requirements gathered during the Analysis phase.
Access to the web application is enforced through a login procedure, where the users are requested to enter their username and password to be authenticated. A mock-up of the login form is provided in Figure 11. It must be underlined that, to strongly enforce the web application's security, no "Remind me" option is allowed. In addition, from the login page, no further navigation in the web application is permitted.

Providing the correct credentials allows the users to access the inner pages of the application. The available pages are:

- Dashboard: provides an overview of the current status of the system, with some data shown, together with the last alerts and the last detections.
- Alerts: provides the list of all alerts managed by the system. These alerts could or could not lead to an actuator activation.

- Statistics: provides some useful statistics about the system, such as the number of alerts per day or the number of faults.
- System: includes the list of detectors, actuators and zones. Allows the privileged users to add one of them through a guided procedure.
- Admin: a page reserved for privileged users, allowing them to modify the system's users.
- Logout: unset the current user's session.

Dashboard The dashboard page is designed to provide users with a complete and easy-to-understand overview of the current status of the system.
While other pages present more detailed data, the dashboard must provide a way to quickly understand what is happening in the field. The page gives suggestions about:

- The number of zones inserted in the application.
- The number of detections in the last 24 hours, is useful to quickly understand if there are some strange flocks' behaviour in the area.
- The number of manual overrides issued by the privileged users during the day. A higher number of these may indicate the presence of faults in the detectors in a certain area or the lack of accuracy of the detection model.
- How many active detectors there are, to keep track of the current monitoring infrastructure.
- How many suspicious faults have been detected, computed as the number of detectors that do not interact with the server for a while. In addition to the number of manual overrides, this information is useful for estimating the system's health.

Those suggestions are provided at the top of the page to allow the users to rapidly check in case of emergencies.
Then the page shows a brief history of the last alerts and last detections. A mock-up of this page is provided in Figure 12.

Clicking on the suggestions, the user is redirected to the page with corresponding details: these pages are described below. The pages not described are the "Overrides" and the "Faults" pages because these are available only from the dashboard.
The schema of the Overrides page is shown in Figure 13. Instead, the schema of the Possible Faults page is in Figure 14.
The reason these pages are not shown in the navigation bar is that to simplify the user interface, in it are present only the pages commonly used. With the application correctly working, these pages are not supposed to be frequently accessed.

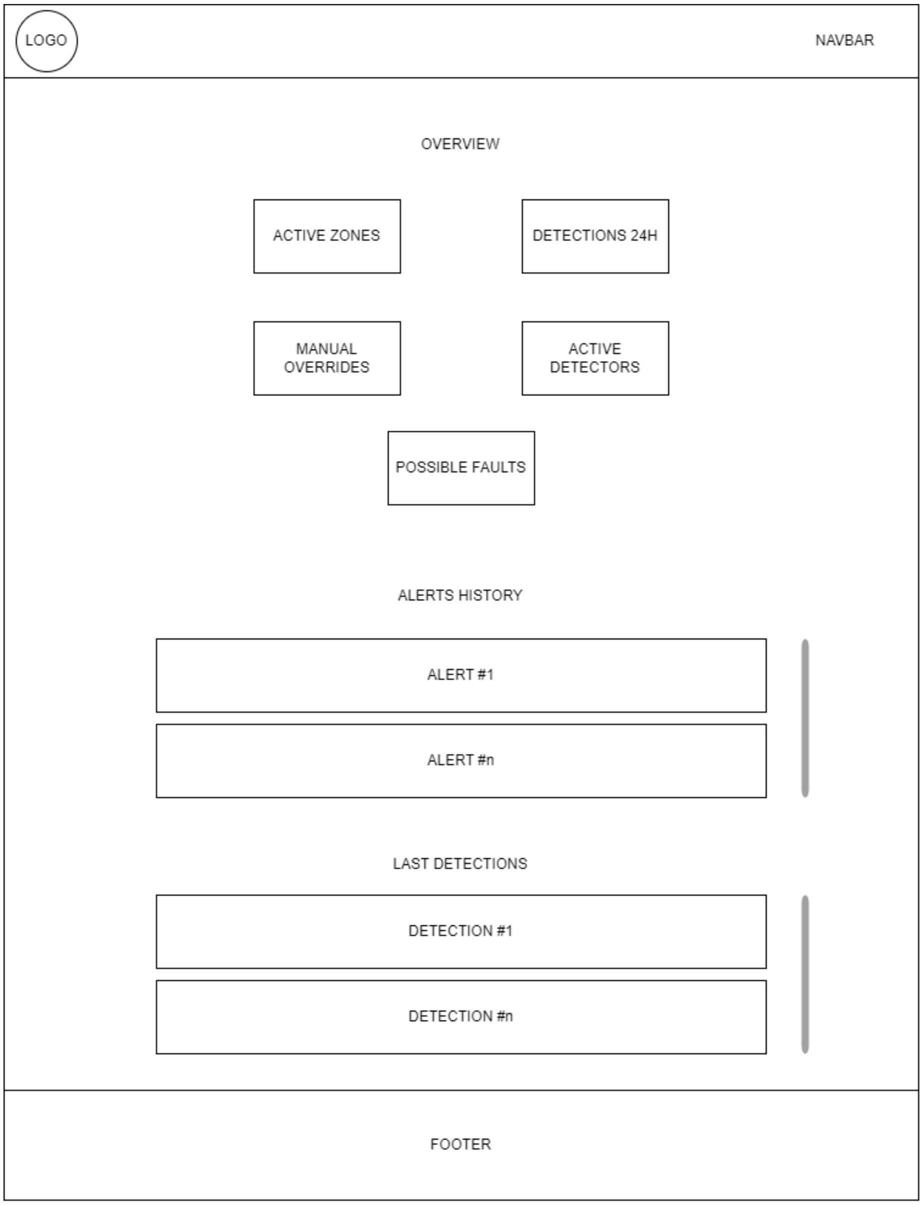

Figure 12: Draft of the dashboard page

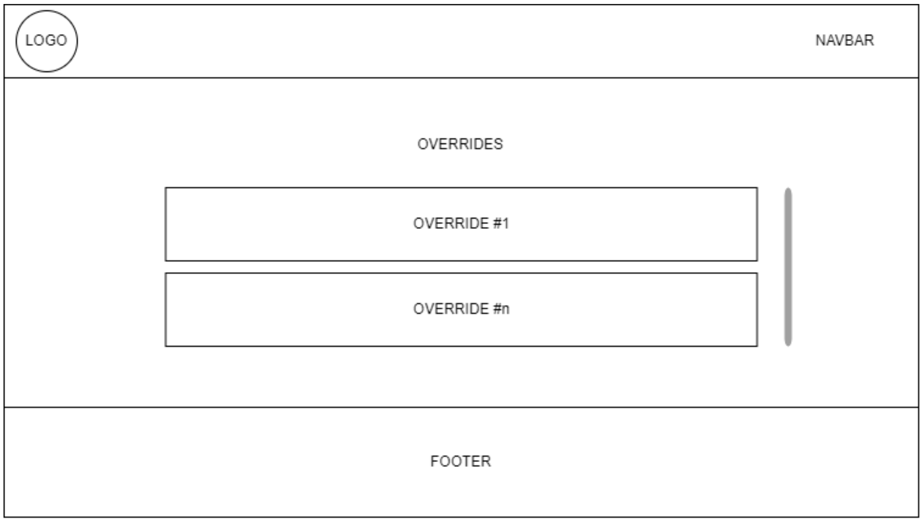

Figure 13: Mock-up of the Overrides page

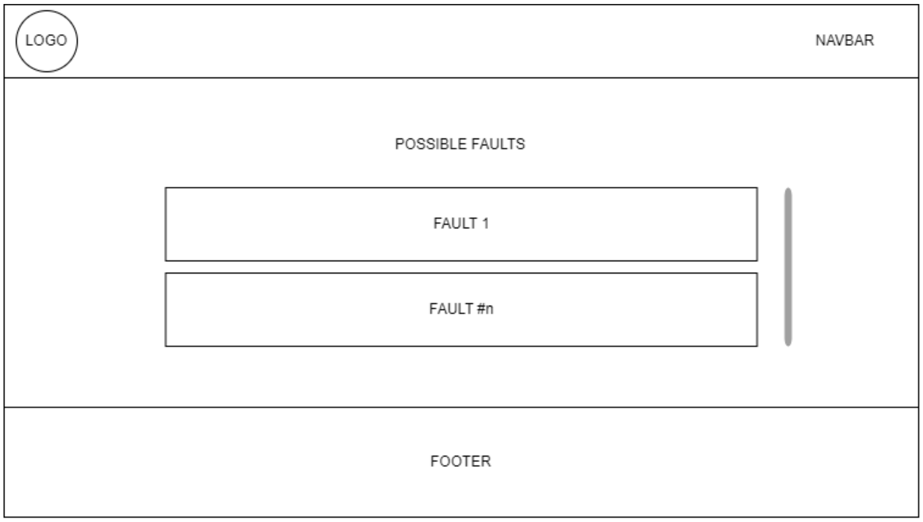

Figure 14: Mock-up of the Possible Faults page

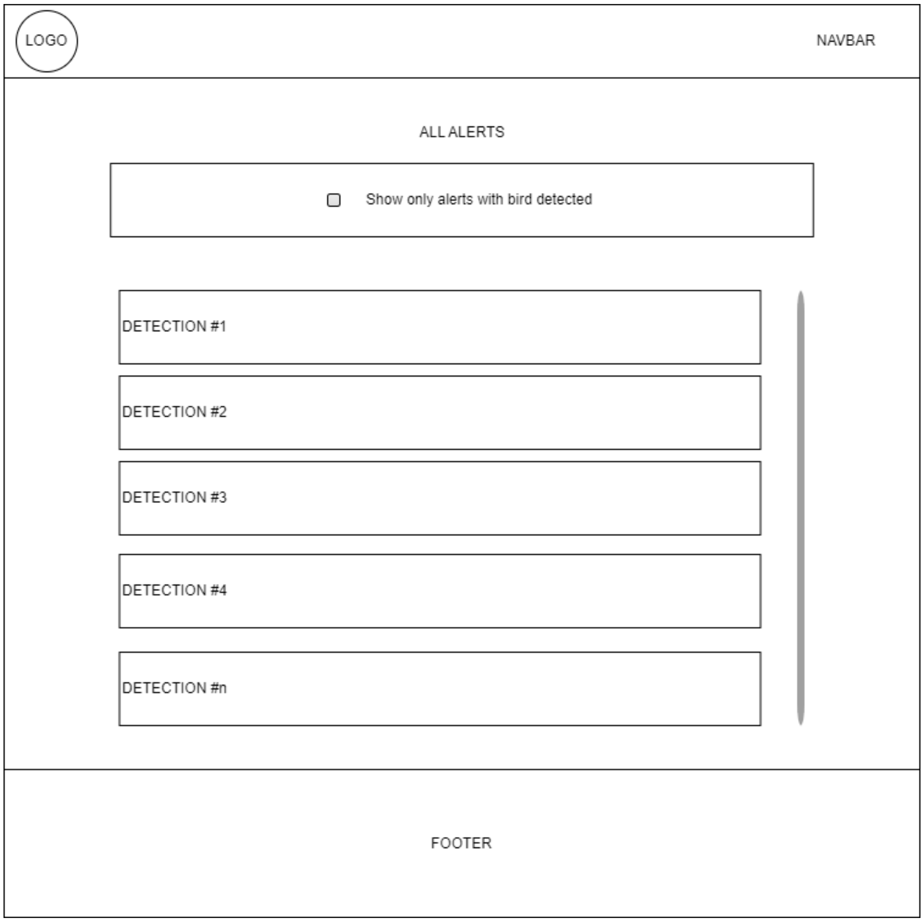

Figure 15: Draft of the Alerts page

Alerts This page provides the list of all the detections performed by the Detectors on the field during the cycle of life of the system.
The detections displayed there, include both detections with birds detected and not detected. In this way, it is possible to overview the activity on the field from one single page, simplifying the users' experience. Naturally, the user can filter the results to show only the alerts where birds were detected.
A mock-up of this page is provided in Figure 15. By clicking on each detection card, its details will be shown in a simple pop-up panel. This allows the users to remain on the page while having a complete description of a detection.

Statistics The present web page offers an array of statistical analyses to the users, that aim to provide statistical information related to the system.
The users can peruse the statistical data furnished on the page and gain insight

into the functioning of the system.
The analysis performed includes:

- Detections per detectors in the current day, useful to determine flock patterns in the runways.
- Activations of the actuators in the current month, divided into automatic and override activations. With this analysis, a user can understand if the system's capabilities are degrading.
- Total number of detections per month. As for the first case, it is useful to observe the trend of bird activity in the airport.
- Total activations per zone. This information is provided to support decision-makers in determining if more devices on the field are required.

As in the previous cases, its mock-up is presented (Figure 16).

System This page allows users to oversee the devices that are currently connected to the system, as well as the zones that have been configured.
For users with privileged access, there is an additional capability to add new devices and configure new zones. This feature can be leveraged by authorized personnel to augment the existing network infrastructure as per the organizational requirements.
Figure 17 provides a mock-up of the System page. Instead, the guided procedure to add a new device is described in Figure 18.
To add a new device, both an id and a passcode must be provided: those data must correspond to the data provided within the device. Both are strings without any blank or special characters.
Similarly, it is possible to set a new zone: in this case, the only input required is a brief description of the zone. The mockup of this form is provided in Figure 19. In this case, no constraints are applied to the user's input.
It must be underlined that to complete the procedures described above, the user must provide a valid OTP code, received on the Telegram bot.

By clicking on the devices or zone cards, the user will be able to verify some useful data:

- For the detectors, general information, date of last detection, date of last activity and detection rate are provided. In addition, from this page, a privileged user can remove the detector. A mock-up is provided in Figure 20.
- For the actuators, the page shows general information and the possibility of performing an override or removing the actuator. A reference schema is provided in Figure 21.
- For the zones a pop-up shows the description.

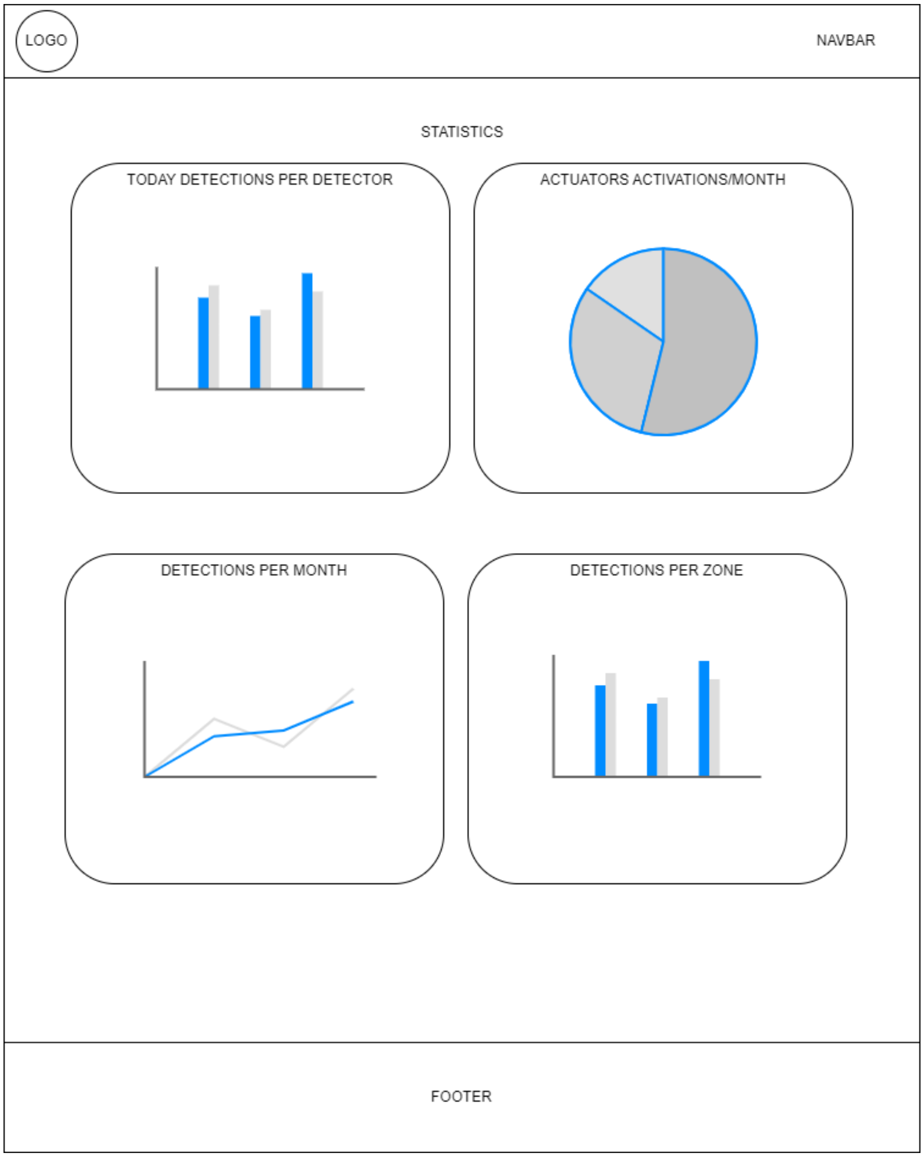

Figure 16: Draft of the Statistics page

It must be highlighted that to remove a device or perform an override, the OTP code must be provided.

Admin dashboard In addition to the pages that are accessible to all users, privileged users are granted access to an administration page.
This page serves as a platform for adding new users to the system or removing the existing ones A mock-up of the page is provided in Figure 22.
To add a new user, the privileged user must insert the username and the password, and then the OTP code is required. A schema about this procedure is provided in Figure 23.
To enforce the application's security, the password must have at least 8 characters, at least one must be in uppercase, a number and a special character.
Finally, to remove a user, the privileged account must click on the card a insert the received OTP code. Naturally, a user can't remove his account.

3.3.2 The Telegram bot

The Telegram bot provides an additional security layer to the system, sending privileged users OTP codes to perform critical actions on the system.
These OTP codes are randomly generated and last for three minutes in the default application's configuration.
If the code is not provided in time, the critical procedure is interrupted.
The bot does not have a user interface, it only sends messages with the OTP codes when needed, exploiting Telegram's APIs.

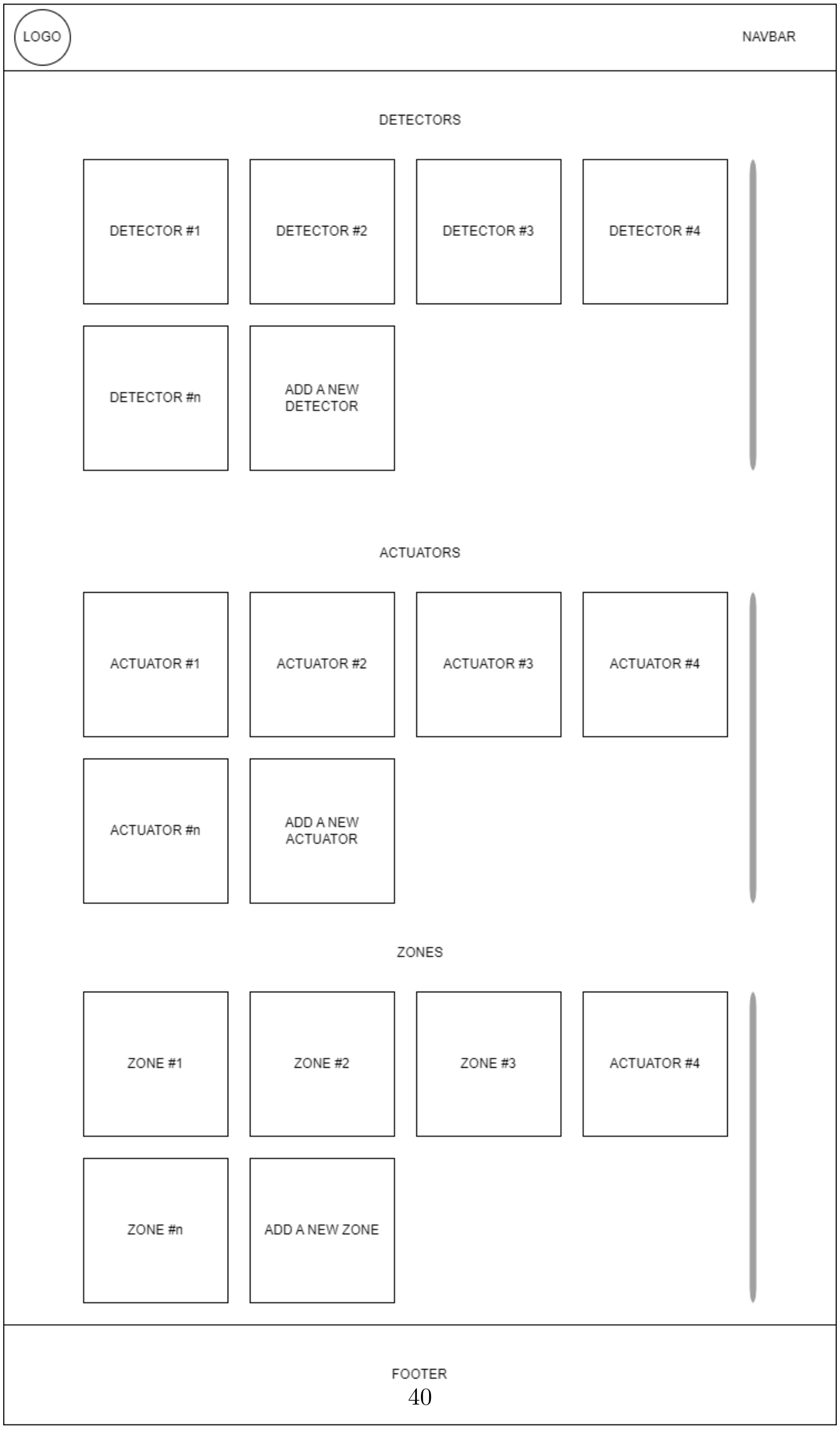

Figure 17: Draft of the System page

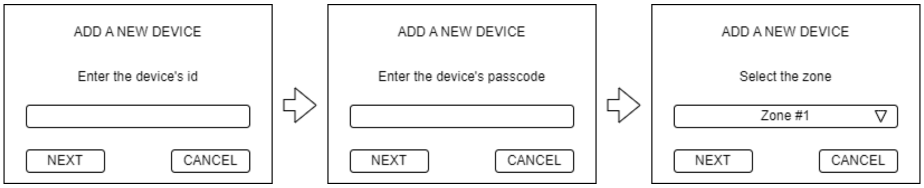

Figure 18: Guided procedure to add a new device

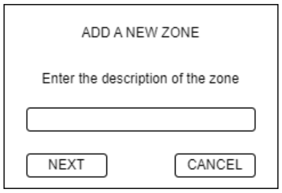

Figure 19: Form to add a new zone

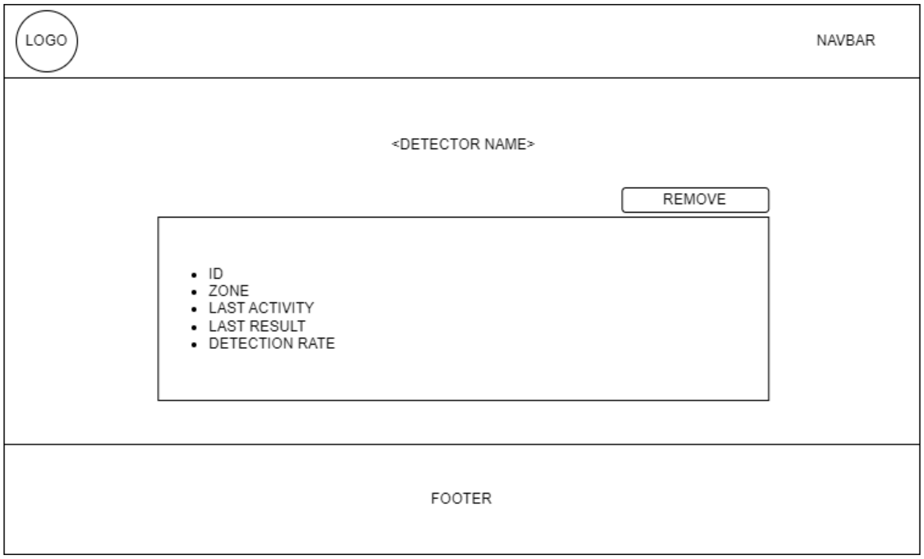

Figure 20: Mock-up of the page containing the detector's details

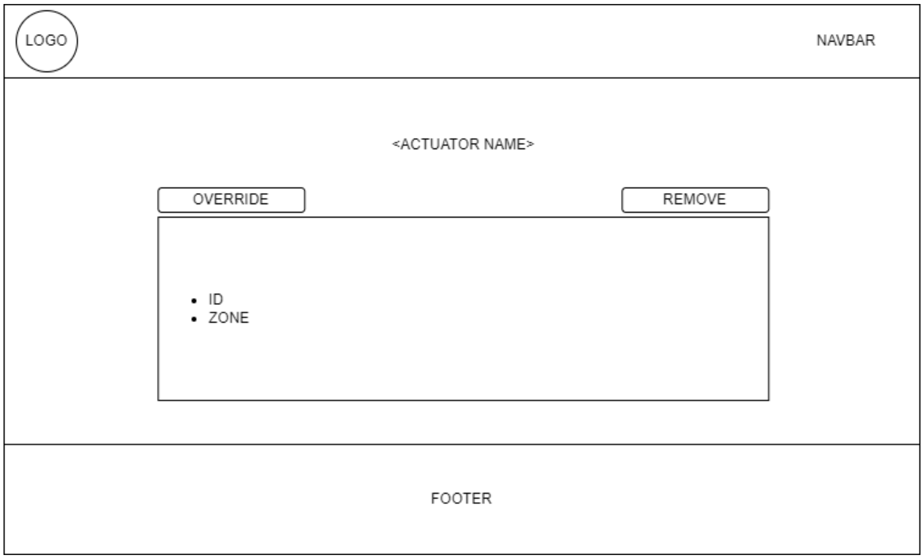

Figure 21: Mock-up of the page containing the actuator's details

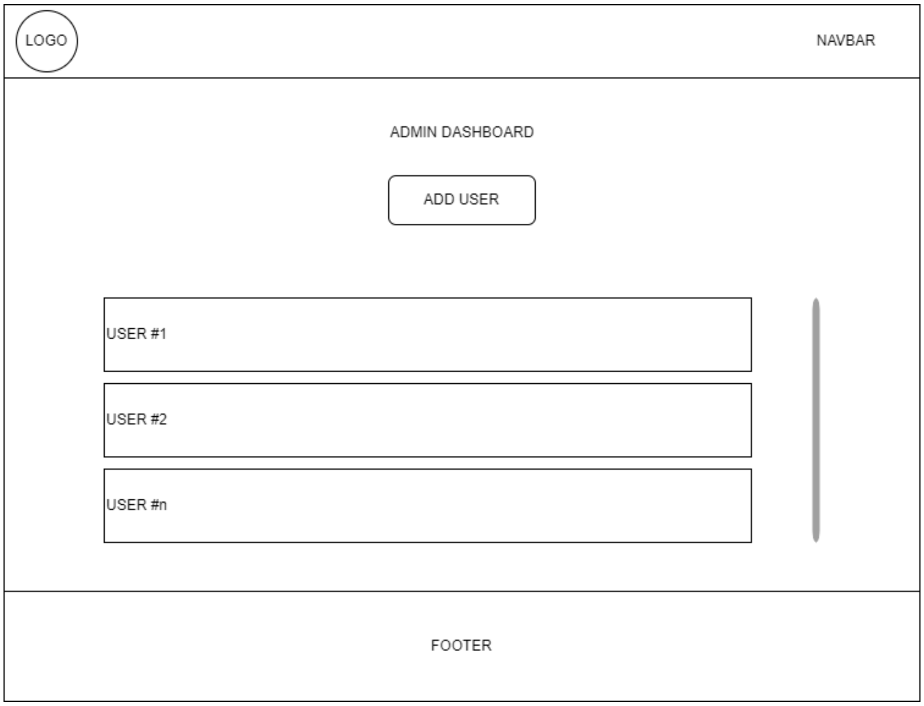

Figure 22: Mock-up of the Admin Dashboard page

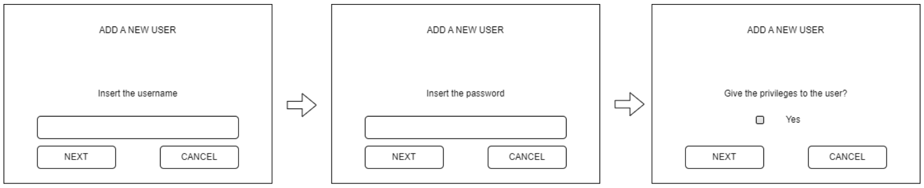

Figure 23: Mock-up of the procedure to add a new user

Sensor	**Pros**	**Cons**	**Cost**
Passive Infrared (PIR) Sensor	– Detects motion and heat – Low power consumption – Simple integration with development boards	– Limited range and field of view – Cannot provide detailed images, so unfeasible for the image recognition task	\$5 - \$15
Camera with IR Filter	– Captures high-resolution images – Can provide visual data for bird recognition – Infrared capability for low-light conditions	– Higher power consumption compared to PIR sensors, more capable batteries are required – May require additional processing for image analysis	\$20 - \$50
Acoustic Sensor	– Captures bird vocalizations – Non-intrusive and can cover a wide area – Suitable for audio-based bird presence recognition	– Limited to detecting vocalizations, not visual identification – Background noise may affect the accuracy, this must be taken into account since the runways are noisy environment	\$10 - \$30
Raspberry Pi Camera Module	– Small form factor – High-quality images and video, suitable for the image recognition task – Directly connects to Raspberry Pi	– Limited compatibility outside the Raspberry Pi ecosystem	\$20 - \$30
Arducam Mini Module	– Compact size – Various resolutions and lenses available – Compatible with multiple development boards	– May require additional adapters for certain boards, increasing the overall costs and the system's complexity	\$15 - \$40
OV7670 Camera Module	– Low cost – Lightweight – Suitable for simple projects	– Basic features – Low-level management is required, making it hard to program – Requires high baud rates, non-compatible with some boards or modules	\$5 - \$10
ESP32 Camera Module	– High-quality images and video stream – Built-in connectivity modules – Suitable for advanced projects – Can be upgraded with an external memory card	– Higher costs with respect to OV7670	\$10 - \$15

Table 3: Comparison of Sensors and Cameras for Bird Presence Recognition

Actuator	**Pros**	**Cons**	**Cost**
Air Cannon	– Provides a strong, directional stimulus – Effective for scaring birds away from a specific area – Adjustable pressure for varying force – De-facto standard	– Requires a compressed air source – May be noisy and potentially disruptive – High power consumption	\$50 - \$200
Buzzer	– Audible alarm to deter birds – Simple to integrate and control – Low power consumption	– Limited range compared to physical deterrents – May not be as effective for all bird species – Emitted sound can be hidden by louder sounds in a noisy environment	\$5 - \$15
LED Lights	– Visual deterrent to scare birds – Can be programmed for different light patterns – Low power consumption compared to other light sources	– Effectiveness may vary based on bird species – Limited impact in well-lit areas	\$2 - \$10
Ultrasonic Repeller	– Non-audible to humans, avoid to bother airport's personnel – Can cover a wide area	– Effectiveness may vary among bird species – Limited range and penetration through obstacles	\$20 - \$50

Table 4: Comparison of Actuators for Bird Presence Recognition

Communication Technology	**Pros**	**Cons**
Wi-Fi	– High data transfer rates – Commonly available and widely supported – Does not require a physical infrastructure	– Limited range compared to other technologies – Power-hungry for continuous operation – Can be disrupted by other devices
Bluetooth	– Low power consumption (Bluetooth Low Energy - BLE)	– Limited range compared to other technologies – Data transfer rates may be lower than Wi-Fi
LoRa (Long Range)	– Long-range communication capability – Low power consumption for battery-operated devices – Suitable for outdoor applications	– Low data transfer rates compared to Wi-Fi – Not suitable for high-bandwidth applications
Zigbee	– Low power consumption – Mesh networking for extended coverage – Suitable for short to medium-range communication in IoT applications	– Limited data transfer rates – Interference with other devices on the same frequency
Cellular (3G/4G/5G)	– Wide coverage area with cellular networks – High data transfer rates (4G/5G)	– Relatively higher power consumption – Requires a cellular data plan – Requires a proper infrastructure, that may be not in place
Satellite	– Global coverage – Can provide continuous communication in some scenarios	– Higher latency compared to terrestrial technologies – Costly infrastructure and communication plans

Table 5: Comparison of Communication Technologies for Bird Presence Recognition

Communication Protocol	**Pros**	**Cons**
MQTT	– Lightweight and efficient for low-bandwidth networks – Ideal for sensor and actuator communication in IoT – Publish/subscribe model allows for scalable implementations	– May not be suitable for high-frequency data – Requires a centralized MQTT broker
CoAP	– Designed for constrained devices and low-power networks – Lightweight protocol suitable for IoT applications – RESTful architecture for simple communication	– Limited to constrained environments – Security considerations for certain implementations
HTTP/HTTPS	– Universal and widely supported protocol – Suitable for web-based applications and APIs – Secure communication with HTTPS	– Higher overhead compared to lightweight protocols – May not be the most power-efficient for constrained devices
Zigbee	– Mesh networking for extended coverage – Designed for low-data-rate applications in IoT	– Limited data transfer rates compared to other protocols – Possible limited coverage – Vulnerable to interference from other devices on the same frequency

Table 6: Comparison of Communication Protocols for Bird Presence Recognition

Feature	**Detail**
Clock	Up to 160MHz
RAM	520 KB
Connectivity	WiFi, Bluetooth, UART, I2C, PWM
TF Card	Max 4GB
Image format	JPEG, BMP, GRAYSCALE
Power consumption	180mA@5V without flash

Table 7: ESP32 Camera details

Endpoint	**Method**	**Parameters**	**Description**
`/api/users`			
`/`	GET, POST		Return the list of users
`/login`	POST	user, password	Validate the user credentials. If are correct, the user is authenticated and a secure token is retrieved.
`/register`	POST	user, password	Insert a new user
`/remove`	POST	userId	Remove a user
`/listoverrides`	GET		Get the list of overrides
`/override`	GET, POST	zone	Overrides the actuators in a given zone
`/dashboard`	GET		Get the data for the user's dashboard
`/api/zones`			
`/`	GET		Get the data about the zones
`/add`	POST	info	Set a new zone
`/api/devices`			
`/`	GET		Get the data about all devices
`/login`	POST	idname, passcode	Authenticates a device
`/remove`	POST	deviceId, type	Removes a device
`/add`	POST	idname, passcode, zone, type	Add a new device
`/classify`	POST	file	Send an image to the ML model

Table 8: Flask APIs Endpoints

Field	**Datatype**	**Properties**
`users`		
id	INTEGER	PRIMARY KEY AUTOINCREMENT
user	TEXT	NOT NULL
pass	TEXT	NOT NULL
isAdmin	BOOLEAN	NOT NULL
`overrides`		
id	INTEGER	PRIMARY KEY AUTOINCREMENT
userId	INTEGER	FOREIGN KEY (userId) REFERENCES user(userId)
zone	TEXT	NOT NULL
time	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP
`devices`		
id	INTEGER	PRIMARY KEY AUTOINCREMENT
idname	TEXT	NOT NULL
passcode	TEXT	NOT NULL
zone	INTEGER	FOREIGN KEY (zone) REFERENCES zones(id)
`detectors`		
id	INTEGER	PRIMARY KEY AUTOINCREMENT FOREIGN KEY (id) REFERENCES devices(id)
`actuators`		
id	INTEGER	PRIMARY KEY AUTOINCREMENT FOREIGN KEY (id) REFERENCES devices(id)
`alerts`		
id	INTEGER	PRIMARY KEY AUTOINCREMENT
zoneId	INTEGER	FOREIGN KEY (zoneId) REFERENCES zones(id)
deviceId	INTEGER	FOREIGN KEY (deviceId) REFERENCES devices(id)
status	BOOLEAN	
time	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP
`zones`		
id	INTEGER	PRIMARY KEY AUTOINCREMENT
info	TEXT	NOT NULL

Table 9: Tables' datatypes

Permission	**Standard user**	**Privileged user**
Can log into the web application	Yes	Yes
Can oversee the system status and access data analysis such as logs and statistics	Yes	Yes
Can oversee the system architecture, accessing the component's details	Yes	Yes
Can modify the system architecture, both in terms of detectors, actuators and zones *	No	Yes
Can add or remove users *	No	Yes

Table 10: Differences between the categories of users