Transactions on Energy Systems and Engineering Applications

Article

To eggspace and beyond: design and implementation of an autogyro-CanSat with IoT purposes using AWS

Franco Rivadeneira[1], Kioshi Kiyan[1], Victor Huayapa[1], Diego Godinez[1], Nicole Perez[1], Abel Hinostroza[1], Sebastian Acosta[1]and Diego Arce[2]

[1] Faculty of Science and Engineering, Pontificia Universidad Catolica del Peru, Lima, Peru.
[2] Engineering Department, Pontificia Universidad Catolica del Peru, Lima, Peru.
* Correspondence: franco.rivadeneira@pucp.edu.pe; kioshi.kiyan@pucp.edu.pe; huayapa.victor@pucp.edu.pe; diego.godinez@pucp.edu.pe; naperezs@pucp.edu.pe; harvi.hinostroza@pucp.edu.pe; sebastian.acostat@pucp.edu.pe and diego.arcec@pucp.edu.pe

Received: 21 January 2024; Accepted: 25 September 2025; Published: 05 November 2025

Abstract: In the context of a lack of educational tools for learning space technologies and satellite development, CanSats were created as an educational tool. This article proposes the mechanical, electrical and software design of a CanSat with an autogyro descent system where the novelty is the implementation of AWS IoT services and Node-RED to store, manipulate and display in real-time the collected weather data. This picosatellite design is capable of safeguarding the integrity of the CanSat's payload where a chicken egg will be placed during the flight and landing phases. Often, other designs of CanSats use local servers implemented on the computer or laptop of the team for storage and display of the data. This makes it more difficult to share the information to people without access to the computer where the server was specifically deployed. The use of AWS services for the Internet of Things is very useful in sharing and displaying the collected information to the public interested in the collected weather data. One of the AWS services implemented allows data subscription through Gmail. The findings made in this paper hold implications for applications involving the transportation and safe landing of delicate payloads in space exploration missions. As a result of the implementation of this design, the separation between the secondary and primary load was successfully achieved and the weather data was transmitted.

1. Introduction

Space missions are more common daily due to humans' enthusiasm for the universe's exploration. Given that interest, in 1998, Professor Bob Twiggs of the Space Development Laboratory of Stanford University proposed the idea of making a soda can-sized satellite [1], now known as CanSat, as a method

How to cite this article: Rivadeneira, Franco; Kiyan, Kioshi; Huayapa, Victor; Godinez, Diego; Perez, Nicole; Hinostroza, Abel; Acosta, Sebastian; Arce, Diego. To eggspace and beyond: design and implementation of an autogyro-CanSat with IoT purposes using AWS. *Transactions on Energy Systems and Engineering Applications*, 6(2): 628, 2025. DOI:10.32397/tesea.vol6.n2.628

of testing the communication or the mechanisms of the system [2]. In addition, the CanSat was also created as an educational tool to encourage students to learn about space technology [3, 4] .

A CanSat is a device with a can's dimensions, designed to accomplish a determined mission, such as image processing or measuring variables [5]. The Picosatellite rises with a rocket, drone, or helium balloon until an altitude of less than one kilometer. Then the Cansat comes down with a parachute, autogyro propellers, or another method to reduce speed. And before, over, and after its trip the CanSat transmits information to the ground station [6].

The CanSat's telemetry is vital to know its conditions in real-time as the altitude, velocity, and acceleration employing the IMU and GPS [7]; and the battery voltage to know the actual state of the Picosatellite. Additionally, it allows the user to send commands to activate actuators. Moreover, it can send images [8] and atmosphere variables like pressure, temperature, methane, and butane among others [9]. Therefore, this article proposes the use of IoT to save the sensors' measurements, have better access to them, and avoid the loss of information.

2. State of Art

For the search of previous projects, priority was given to CanSats focused on data transmission and data processing or that implemented an autogyro descent system. As is shown in Figure 1, the CanSat commnly have two descent mechanism, autogyro and using parachutes. In [10] a CanSat is developed especially for measuring and visualizing environmental variables. For this purpose, environmental sensors are implemented to measure temperature, pressure, and relative humidity. For the data transmission, it is used two radiofrequency LoRa transceptors RFM95W-915MHz. One is connected to the main circuit of the CanSat, and the other is connected through a serial connection to the Ground Station PC. In addition, the development of a graphical interface is presented, which allows the visualization of the environmental data collected during the flight and the selection of the serial port for communication. However, a downside of this design is the need to manually press a button to store the data in an external file.

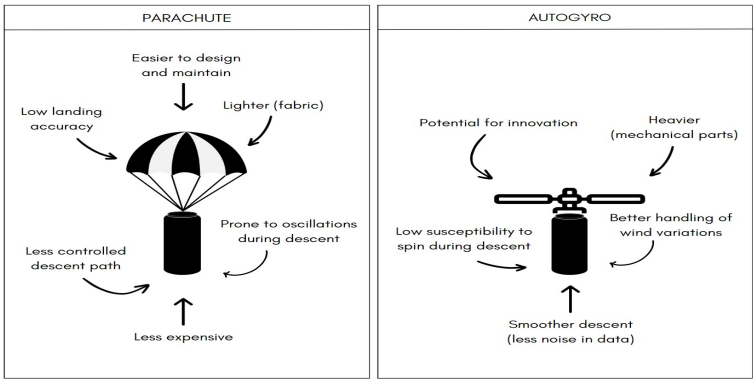


Figure 1. CanSat descent mechanism.

In [11] an interesting arrangement is proposed for the release of the autogyro mechanism blades. After being released from a rocket, this CanSat releases the descent mechanism at an altitude between 670-725 meters. The CanSat controller receives the signal from the altitude sensor and commands the servomotor to rotate a cross-shaped lid enclosing the blades inside the container. Then the autogyro mechanism comes out of the container due to the force of gravity. Another design of a descent mechanism is proposed in [12]. An innovative design for the blades inspired by maple seeds is used in this paper. The physical structure is composed of a lightweight wing frame that maintains a 10-degree angle of attack for increased lift force, a wing axle that provides rigid support to the wing frame and stores a part of the electrical subsystem, and

a PCB capsule that mimics the heavy part of the seed. It features a hook for a servo at the opposite end, connecting to the payload release mechanism of the container.
In [3] an 11-centimeter-high CanSat is presented. Due to the limited space, they placed the electronics on three layers of round PCBs on top of each other. They divided the connections into a sensor layer, another layer where the controller and the elements necessary for its operation are located, and finally, a layer where the GPS module and antenna are placed. This design is particularly interesting since one of CanSat's constraints defined for this application is a height of no more than 22 centimeters to contain the mechanisms, electronics, and an egg chamber. A different structural design for a CanSat is suggested in [13]. This paper proposes a CanSat Out-Of-The-Shelf modular design. To assemble this CanSat, you have to stack a command module and one or several sensor modules, with the condition that one of them is of the nose cone type. The command module uses an Arduino Nano as the main controller and a 443 MHz radio frequency transmitter. Among the nose cone sensor modules are one with 4 laser rangefinders, one with 4 ultrasonic rangefinders and another with an FPV camera. Then there are 2 sensor modules in the form of a cylinder: the model with 4 laser rangefinders and another that was to serve as a container for a motion capture system. In the end, it was only used to facilitate internal wiring for transmitting both data and power to the subsequent module in the stack.

3. Methodology

The design of the system is divided into three components: mechanical design, electrical design, and software architecture design. To realize this project, the ADD Methodology [14] was employed. Consequently, the following usage cases were established, some of them are taken from the "Concurso Iberoamericano de Satélites Enlatados 2023" [15].

- UC1: The picosatellite must be divided into a primary and secondary payload.
- UC2: The main mission is to protect an egg from the landing impact of the CanSat. Additionally, it should measure, transmit, and save the data on the pressure, temperature, and inclination of the CanSat.
- UC3: An autogyro mechanism must be used to reduce the CanSat's descending velocity to 6 m/s.
- UC4: After the landing, the CanSat should keep transmitting information for another 20 seconds. And be able to localize the secondary load.
- UC5: Registered data must be uploaded to a highly available Cloud Database.
- UC6: If the uploaded data to the database has an outlier read of temperature (Higher than 35°C) an alert must be sent to the general user.

On the other hand, ensuring an optimal design also necessitates addressing all constraints. In this study, these constraints pertain to dimensions, weight, and the type of energy source. Accordingly, the design constraints are as follows:

- CON1: Weighing less than 500 grams. The maximum diameter must be 10 cm and the maximum height, less than 22 cm.
- CON2: It is mandatory to employ a 9V square battery for each payload.

3.1. Mechanical Design

To the success of the mission, the following critical tasks were identified: the separation of the loads defined in UC1, the deployment of an autogyro mechanism, the protection of the electronics and the protection of an egg. Therefore, three subsystems were considered for the CanSat design and development: secondary load, autogyro, and the internal structure for the protection of the electronics and egg. Before

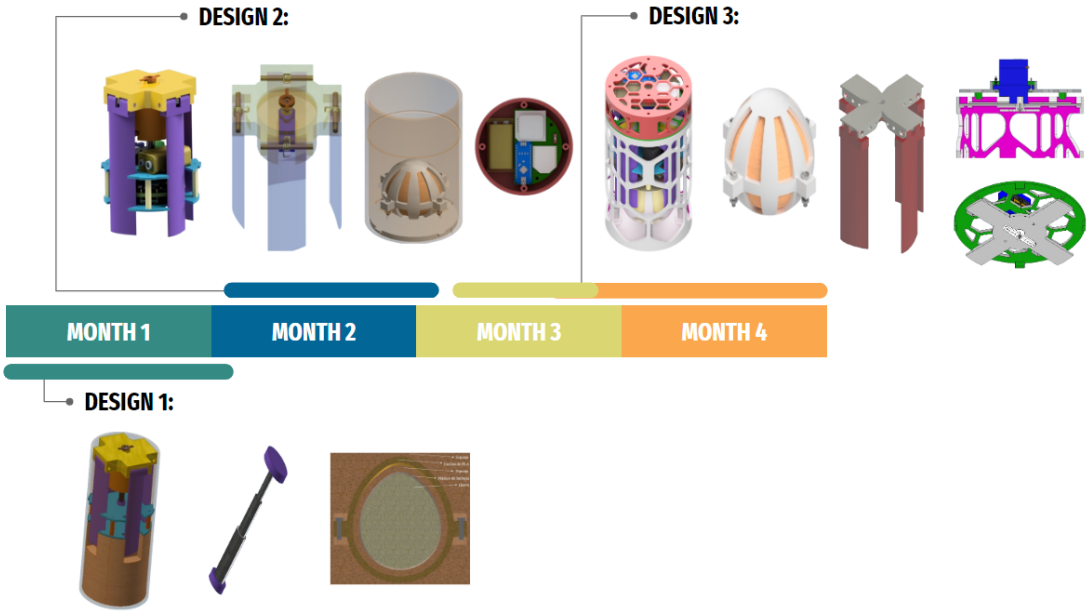


Figure 2. CanSat design timeline.

describing the final mechanical design, it is important to highlight the iterative process of the CanSat design. Hereafter, the evolutionary line of the design and implementation of the system is presented in Figure 2.

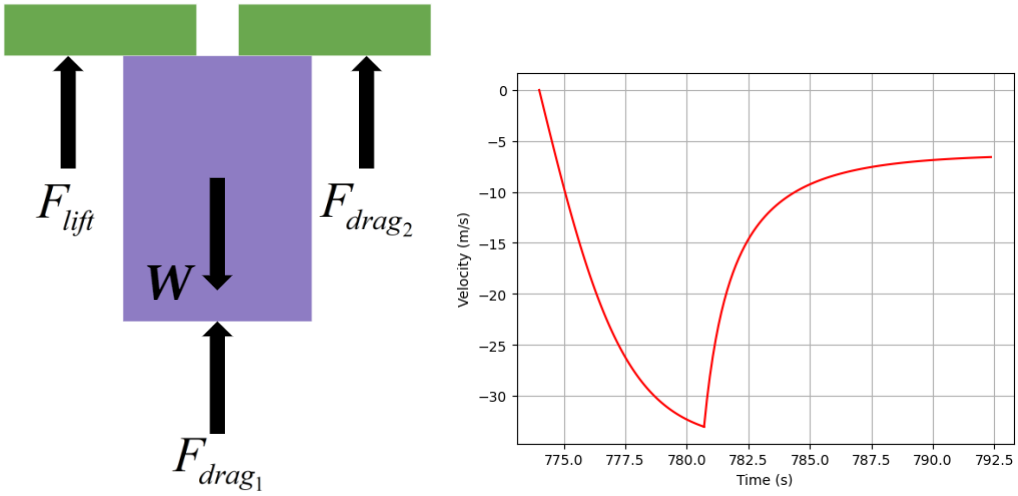
Firstly, Design 1 focuses on the 3D modeling of the structure, material, size, and shape of the components to be used. In design 2, the 3D modeling incorporates electronic components, which leads to changes in some materials due to the weight restriction. For the last phase, the deployment mechanism, the enclosure, and the egg protection system are optimized. The subsequent paragraphs present the final version of each subsystem in detail.

For the secondary load subsystem, a 3mm wall thickness top coating is considered, which also has 50mm diameter hexagonal holes with rounded edges, in order to reduce weight without affecting consistency.

The autogyro subsystem is the mechanism responsible for reducing the descent speed of the payload by optimizing the airflow that comes into contact with the propellers during descent. Deceleration, generated by the air thrust forces on the propellers, is determined by various mechanical factors, such as the profile, quantity, and size of the propellers. For the selection of the propeller profile, commercial equipment propellers such as drones, turbines, and autogyro propellers in traditional helicopters were used as a reference due to their efficiency in optimizing the force generated by the airflow. For this reason, the NACA 2412 profile was selected. The design of the propellers considers the necessary deceleration, optimal angle of attack, and available space within the primary payload. Finally, using the Airfoil Tools website [16], the smallest dimensions that meet the constraints were chosen.

For the theoretical analysis of the performance of the designed autogyro, a Free-Body Diagram with the forces acting on the system is proposed [17], Figure 3a. These forces consider ideal environmental conditions and the spatial arrangement of the CanSat during descent. Solving the dynamic equation, the velocity versus time graph of the CanSat is obtained, as seen in Figure 3b. In this graph, the terminal velocity is observed to stabilize at a speed of 6 m/s, accomplishing the requirement established in UC3.

In a preventive and predictive manner, another analysis is done about what the behavior would be like during the descent in typical and/or critical non-ideal situations.



(a) Free-Body Diagram. **(b)** Ideal velocity vs time.

Figure 3. Theoretical model of the descending behavior.

a) Very low wind condition:
In the worst case there is no rotation in propellers and then no lift force, in addition due to action of V_c there must be drag force, but its value should be small because of the shape of the body, which does not have a wing profile in V_c direction,then C_d is considered = 0.2. After FBD:

Figure 4. FBD under very low wind condition.

Making simplifications:

$$a_y = \frac{A_{cansat} * \rho * V_c^2 * C_d/2 + 4 * A_h * \rho * V_c^2 * C_d/2 - W}{m}. \quad (1)$$

Table 1. Poor wind parameters.

Parameter	Symbol	Value
Drag Constant due to Freefall	C_d	≈ 0.2
Air Density	ρ	$0.91\ \mathrm{kg/m^3}$
Descendant Speed	v	$25\ \mathrm{m/s}$
Propeller Surface Area	A_h	$0.003\ \mathrm{m^2}$
CanSat Surface Area	A_{cansat}	$0.008\ \mathrm{m^2}$

Replacing values:

$$a_y = -7.54 m/s^2.$$

It is observed that the acceleration will always be negative, which indicates that in very low wind conditions the body could not brake.

b) Condition of the body inclined $\beta°$ with respect to the vertical:

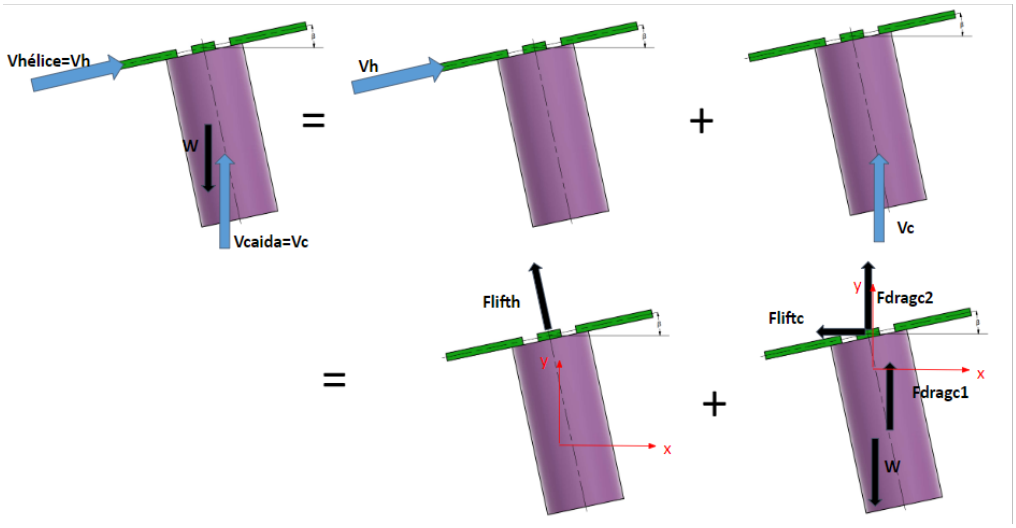


Figure 5. FBD under β inclination condition.

There:

$$a_y = \frac{F_{dragc1} + F_{dragc2} + F_{lifth} * Cos\beta - W}{m}, \tag{2}$$

$$a_x = \frac{F_{lifth} * Sen\beta + F_{liftc}}{m}. \tag{3}$$

If β is negative and exceeds 10° the body will stall, since under these conditions C_d » C_l On the other hand, if β is positive, the body has freedom of rotation up to a certain angle, for $\beta = 10$:

Table 2. Inclination $\beta = 10$ Parameters.

Parameter	Symbol	Value
Lift Constant due to Freefall	C_{lc}	0
Lift Force due to Freefall	F_{liftc}	$0 \cdot \cos(\beta)\ N$
Lift Constant due to Freefall	C_{dc}	≈ 0.2
Drag Force in cylinder due to Freefall	F_{dragc1}	$0.447 + 19.34 \cdot \sin(\beta)\ N$
Drag Force in propellers due to Freefall	F_{dragc2}	$0.683 \cdot \cos(\beta)\ N$
Lift Constant due to Propellers	C_{lh}	0
Lift Force due to Propellers	F_{lifth}	0

Then:

$$a_y = -0.043 \approx 0 m/s^2 \tag{4}$$

$$a_x = 0 \tag{5}$$

Approximately from this value onwards, the body is not able to support its own weight ($a_y < 0$). In both cases, the effect of inclination allows the body to move horizontally in exchange for a reduction in lift. There is also a 20° inclination robustness. This means that if it goes outside these limits, the body will have a chaotic behavior.

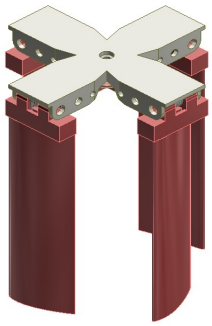


Figure 6. CanSat's autogyro.

The four propellers are attached to a rotor and a bearing that allows relative movement between the payload and the rotor, ensuring that the propellers rotate without dragging the primary payload (Figure 6). Additionally, for the autogyro system to exit the primary payload, there is an expulsion mechanism based on the storage of elastic potential energy contained in a compressed spring. This spring connects the internal structure of the primary payload with an adapted piece that connects to the bearing, which in turn connects to the rotor. For the assembly, the autogyro's spring is compressed using a cross-shaped plate, whose edges fit into an interior groove of the main container. The expulsion mechanism can only be released when a servomotor, located in the secondary payload, is activated and the edges of the plate are aligned longitudinally to a path for its escape, as shown in Figure 7.

On the other hand, regarding the internal structure, priority was given to the space occupied by electronic components and the egg protection system. Two levels were arranged, distributing the necessary spaces to accommodate all components. One of the platforms is made of 2 mm thick acrylic, and for the second platform, the geometry of the electronic board was adapted, with nylon separators. The structure is connected to the casing using nylon bolts.

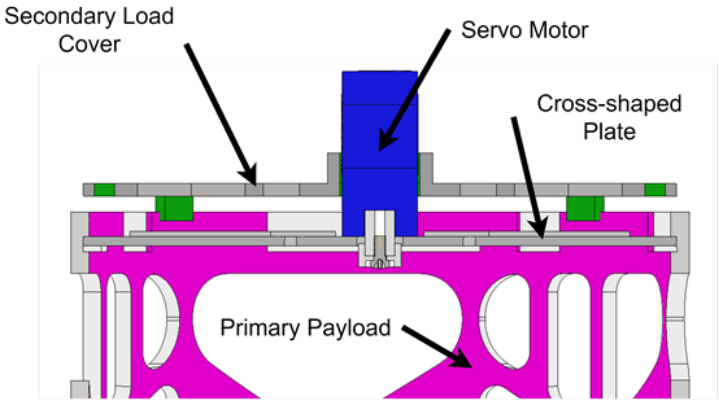
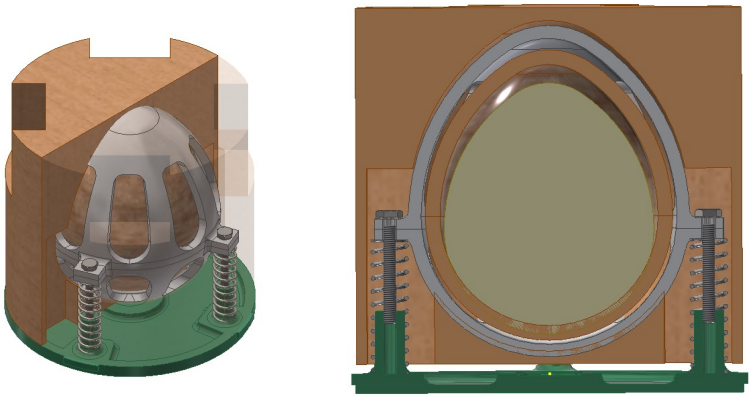


Figure 7. Detail of critical plate for autogyro subsystem.

As for the egg protection system, it consists of a layer of EVA foam and two symmetrical TPU shells with four fins adapted to accommodate screws. These fins allow for securing the egg-shaped shells and, in turn, connecting the shell with four axes that are further attached to the base of the main casing of the primary payload. The eggshell can move freely using the axes as a guide. Additionally, to dampen the impact of the eggshell at the moment the payload collides with the ground, there is a spring on each axis, reducing the instantaneous acceleration of the shell upon impact [18]. Figure 8 provides a more detailed view of the element distribution.



(a) Egg's protection isometric view. **(b)** Egg's protection section view.

Figure 8. Egg's protection system.

The assembly of all the aforementioned subsystems is shown in Figure 9.

In Figure 11, the mechanical analysis of the egg protection system is observed. The egg/electronics protection system consists of a layer of EVA foam and 2 symmetrical TPU shells bolted together. To distribute the energy received by the system during impact in a better way, springs are used, these are aligned through shafts containing a nylon bolt and threaded holes orthogonal to the bottom base whose limits are the base itself and the plates joining the protective half-shells of the egg.

It is known that the most critical case occurs immediately after the collision. Here it is expected for deformation and stress to be the highest. In the impact a distribution of energy takes place, this is divided into potential energy in springs, kinetic energy of the egg plus some energy lost in the form of work. For

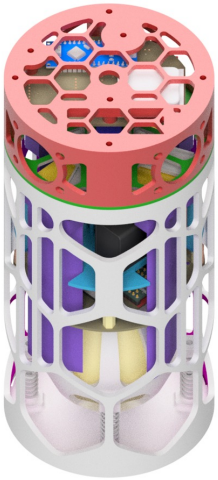


Figure 9. CanSat assembly.

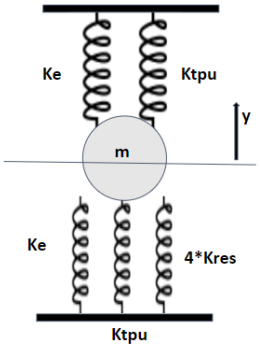


Figure 10. Egg protection simplified to mass-spring system.

the few first moments, it is valid to simplify the system to one of mass-spring as seen in Figure 10 . Which implies that there are no losses at work.

Then:

$$\frac{1}{2}mv^2 = \frac{1}{2}m\frac{\mathrm{d}y'^2}{\mathrm{d}t} + \frac{1}{2}k_{eq}y^2, \tag{6}$$

where:

$$Keq = (\frac{1}{4 * Kres + Ktpu + Ke} + \frac{1}{Ktpu + Ke})^{-1}. \tag{7}$$

After replacing constants and solving the differential equation we get the behaviour of all the system during the few first instants and the maximum deformation it can reach:

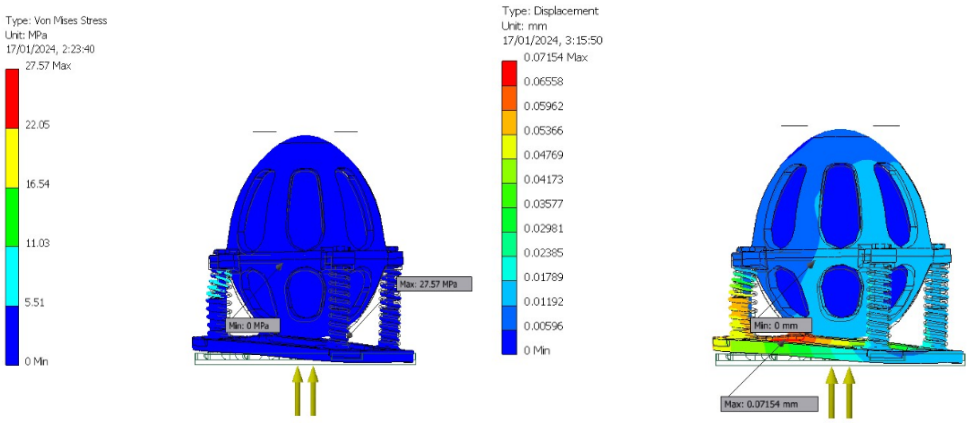
$$y(t) \approx 0.000533 \sin(11252.93 \cdot t), \tag{8}$$

$$y_{max} \approx 0.000533m = 0.533mm. \tag{9}$$

For the collision simulation in software, it is assumed that the collision is vertical, an expected case since if it were horizontal it would be interpreted that the system does not function properly. The parameters are those shown in the table below.

Table 3. General specifications.

Specification	Value
Terminal Velocity	6 m/s
CanSat Mass	500 g
Impulse	3.0 kg · m/s
Time	0.3 s
Force	12.0 N
Area	6720.1 mm^2
Pressure	0.0018 MPa



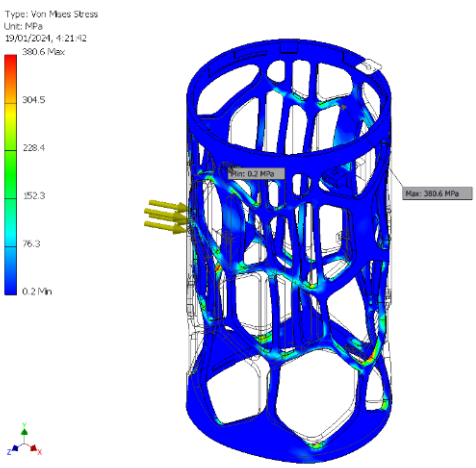
(a) Von Mises Stress. **(b)** Displacement.

Figure 11. Results of vertical collision simulation for egg protection system.

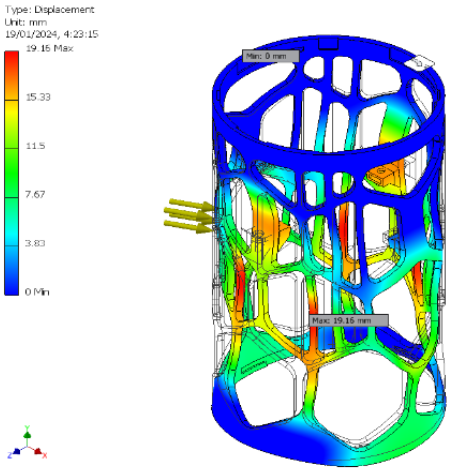
Regarding the mechanical characteristics of the simulation, the entire area above the egg protection subsystem is assumed to be fixed parts. All the pieces have been assigned with their corresponding materials except for the sponges that have been approximated. Regarding the simulation results, the efforts reflect that the impact energy is consumed in the spring and the cover, the same occurs for the deformations, as seen in Figure 11, which is desirable. Out stand that in both simulation and mass-spring system the maximum deformation is under 1 mm and its values are not too far one to another.

The following section presents stress tests using Autodesk Inventor software, considering the von Mises criterion. The analysis is conducted at the moment the system impacts the ground since it is the moment when the greatest load is generated on the CanSat. Utilizing the maximum force generated at the base of the CanSat due to the impact, an analysis of the main components is performed, whose functions include protecting the payload. Applying a pressure equivalent to a 12 N force, obtained based on the values from Table 1. The directions of the arrows, horizontal and vertical, represent the critical cases to which the body would be subjected. In Figure 12, the internal structure is observed, which is attached to the main casing through four internal screw-mounted fins.

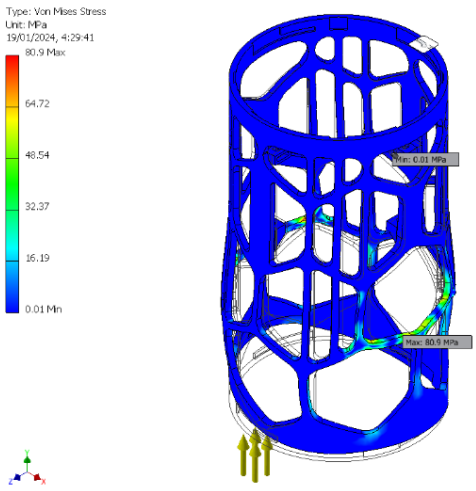
The secondary load, which disconnects from the main load when deploying the autogyro system, is shown in Figure 13, Those results follow the same order as the main casing. It is important to mention that the autogyro system does not have a rigid connection with the payload, so the impact it receives is not considered, assuming that it is not relevant if it breaks since its main function is to reduce the speed during the CanSat descent.



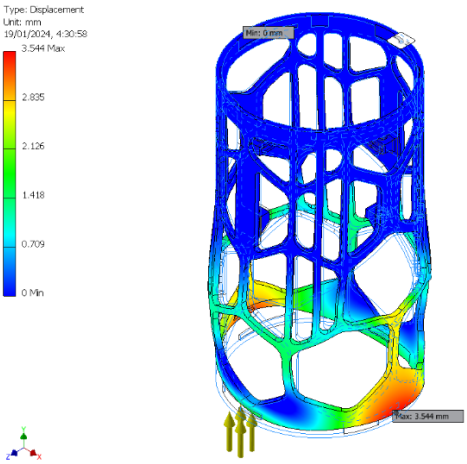
(a) Horizontal load Von Mises Stress.



(b) Horizontal load Displacement.

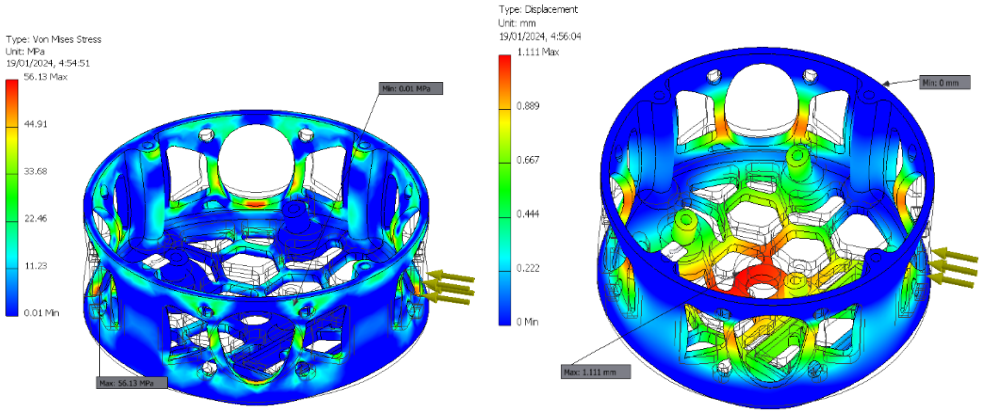


(c) Vertical load Von Mises Stress.

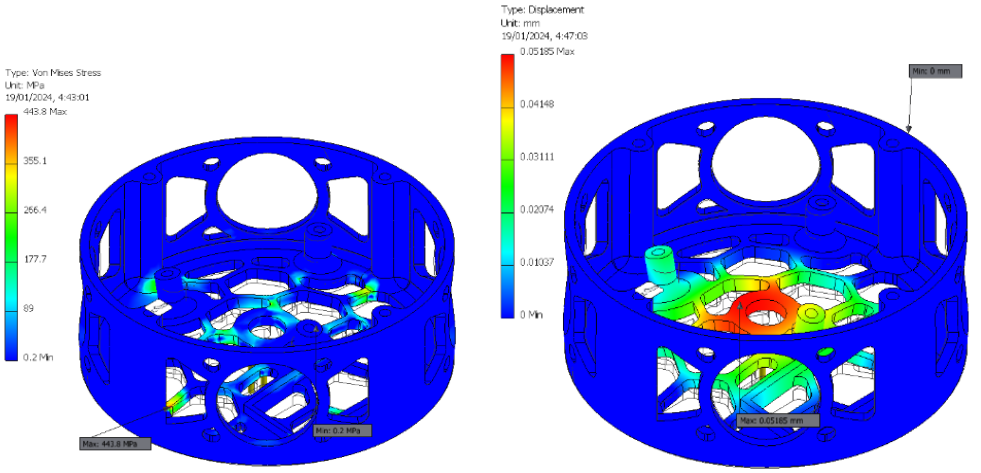


(d) Vertical load Displacement.

Figure 12. Results of collision simulation for the main casing.



(a) Horizontal load Von Mises Stress. **(b)** Horizontal load Displacement.



(c) Vertical load Von Mises Stress. **(d)** Vertical load Displacement.

Figure 13. Results of collision simulation for secondary load.

3.2. Electrical Design

The electrical architecture comprises two circuits: the primary load circuit and the secondary load circuit. The following components were selected based on accessibility, power consumption, size, weight and price: GPS NEO-6M for localization; XBEE S2C for data transmission; GY-91 for inertial, pressure, and temperature measurements; Teensy 4.1 and Arduino Nano for computing in the primary and secondary payloads, respectively; Micro Servo SG90 for the release mechanism; and 9V batteries for power supply. Figure 14 shows the components and connections of both circuits.

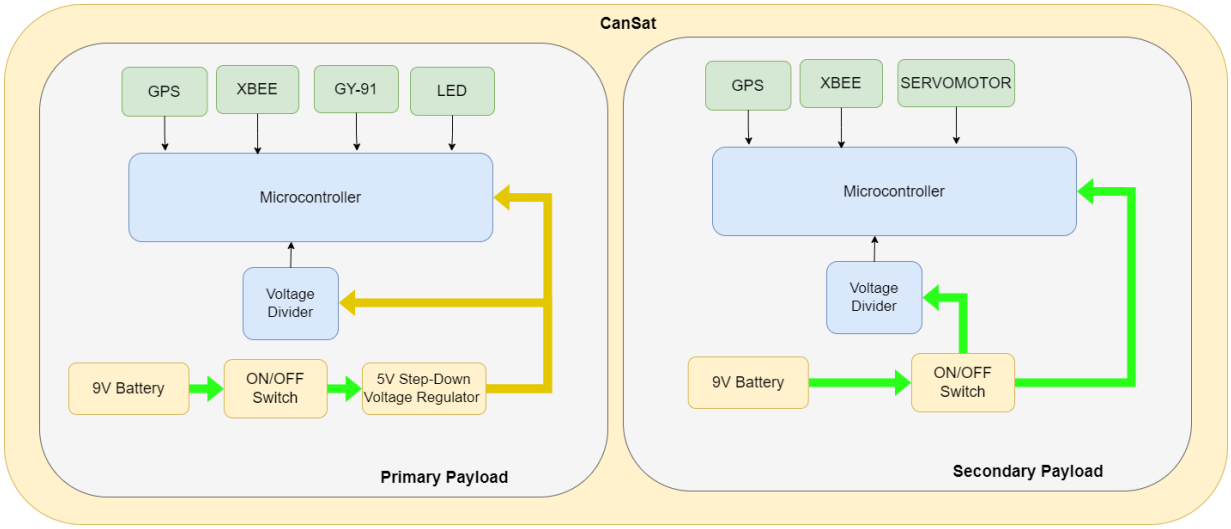


Figure 14. Block diagram of the electrical architecture.

The system controls each stage of the mission using a Finite State Machine model, as shown in Figure 15. The mission consists of five stages or states: Start, Ascend, Free Fall, Gyro Fall, and Landing. The conditions for transitioning between states are also shown in Figure 15. In the Start state, the system sets the current altitude measurement as the ground reference. After the CanSat is lifted 2 meters above the ground, the program transitions to the Ascend state. It remains in this state until the maximum altitude is reached. Once a negative velocity is detected on the vertical axis, the Dropped condition is activated, and the system transitions to the Free Fall state. The Gyro Fall state is reached when the system detects an altitude of 250 meters above the ground, at which point the autogyro subsystem is deployed. For security purposes, this condition is detected using two criteria: the pressure sensor and the time elapsed since the start of the Free Fall State. The Landing State refers to the final 10 meters of falling, the impact, and the time until it is turned off. The FSM model for this CanSat design is mathematically described as follows:

- **States**: $S = \{\text{Start}, \text{Ascend}, \text{Free\_Fall}, \text{Gyro\_Fall}, \text{Landing}\}$
- **Input Alphabet**: $\Sigma = \{1, 2, 3, 4, 5\}$
  - 1 = raised above ground level
  - 2 = dropped
  - 3 = 250 meters from the ground
  - 4 = 10 meters from the ground
  - 5 = 1-minute delay
- **Initial State**: $s_0 = \text{Start}$
- **Final States**: $F = \emptyset$

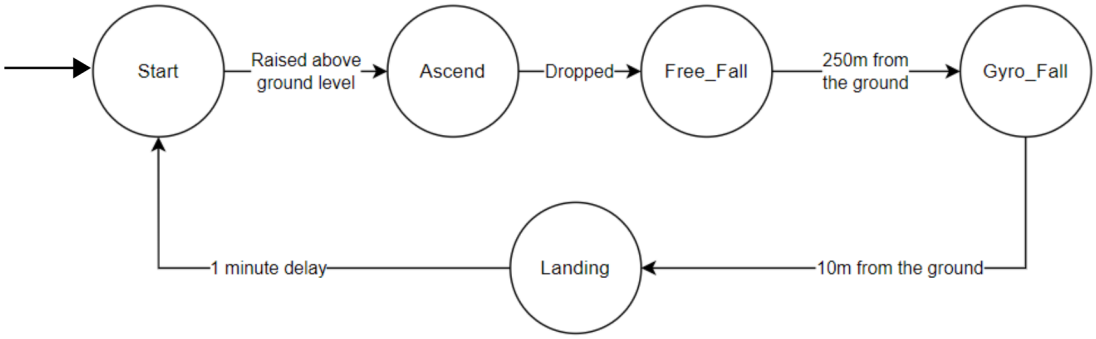


Figure 15. Finite state machine diagram.

- **Transition Table**:

Current State	Input	Next State
Start	1	Ascend
Ascend	2	Free_Fall
Free_Fall	3	Gyro_Fall
Gyro_Fall	4	Landing
Landing	5	Start

3.3. Software Architecture Design

One of the basic missions of a picosatellite is the transmission of data collected during the descent to the ground station. For that reason, an optimal telemetry system must be designed to accomplish the requirements established previously. This section will go into the design of the telemetry system and how the data is transmitted from the sensor to the graphical interface. In addition to detailing the implementation of AWS Internet of Things services in the telemetry system and the design of the graphical interface.

The communication system is formed by two main subsystems as Figure 16 shows: the CanSat and the Ground Station. The CanSat is equipped with multiple sensors such as a gyroscope, accelerometer, pressure, temperature, and altitude sensor, using a microcontroller to process all acquired data and then send it to a receptor using an RF transmitter. On the other hand, the ground station consists of an RF receiver to obtain the data transmitted by the CanSat, a laptop that reads the data obtained, and a web server.

As Figure 17 shows, the first step of the workflow starts when the CanSat measures the environment data conditions. The collected data is then encoded by the microcontroller as a JSON string and transmitted using a network of radio frequency modules. Additionally, due to the significant distance between the CanSat and the Ground Station, a Yagi antenna is connected to the Ground Station radio frequency module to increase the range of the signal and ensure that the telemetry data is received by the Ground Station. The data is transmitted through the RF receiver module to the serial port of the Ground Station computer to be received by the MQTT broker of the computer where the Node-RED is running. Then, the workflow done in Node-RED acts as an MQTT client to receive and process the telemetry data, as well as display the information in real-time through the dashboard built in Node-RED. Additionally, a .csv file was generated to store all the data received during the entire execution time of the workflow.

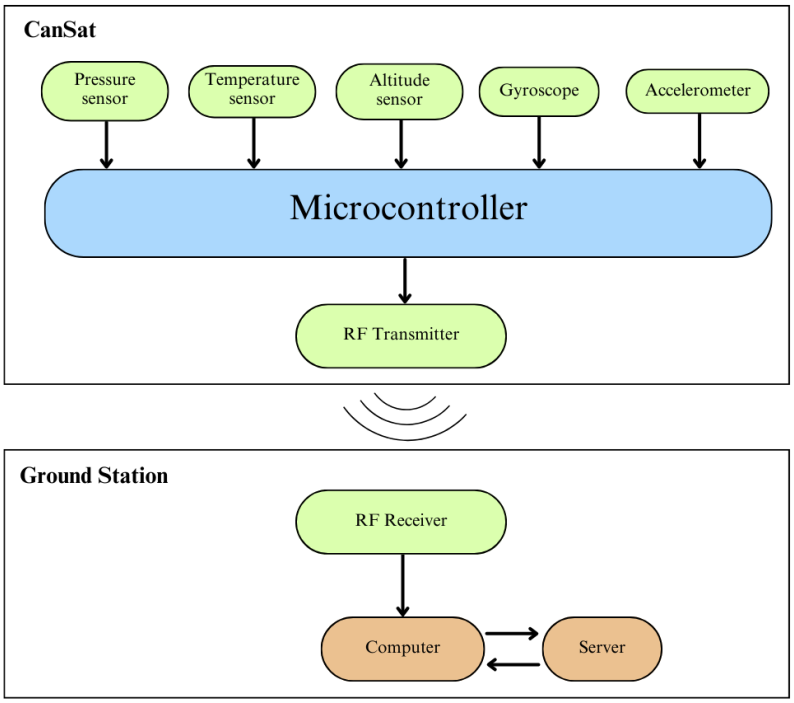


Figure 16. Communication system.

The Node-RED programming tool is employed to process real-time data from diverse sensors on the CanSat for interface design. This entails reading information received via the XBee RX node and extracting details from the transmitted JSON object. Various functions such as the "function block" and "JSON block" from Node-RED are applied to retrieve specific data such as temperature, pressure, altitude, acceleration on the X, Y, and Z axes, distance, secondary payload direction, and battery percentage. After obtaining the data, additional functions using sensor fusion and Kalman Filter are utilized to calculate inclination on the X and Y axes and velocity on the Z axis. Subsequently, the calculated values are saved in a .csv file for future data analysis. The graphical representation of the real-time data acquired from the sensors is depicted in Figure 18. This comprehensive process ensures the effective handling and analysis of CanSat's telemetry data. Finally, the latest update of the user interface was the implementation of the "Push to AWS" button, which can store all the collected data on a safety database.

The AWS layer or system was established to meet the requirements associated with cloud operations. The initial emphasis is on implementing the use case (UC6), which entails securely storing sensed data in a database that is highly available and secure. This implementation leveraged the AWS DynamoDB service, a tool renowned for its exceptionally high availability, safety and scalability [19, 20]. Furthermore, to fulfill the requirement of notifying the user via email when the CanSat registers an outlier value for temperature (UC7), the AWS Simple Notification Service (SNS) is employed. Finally, for the orchestration of these services, AWS IoT Core will be utilized. This Amazon service has been previously employed in similar works, demonstrating its efficiency.[21–23]. Fusing all AWS services used in this study, the software architecture simplified is shown in Figure 19.

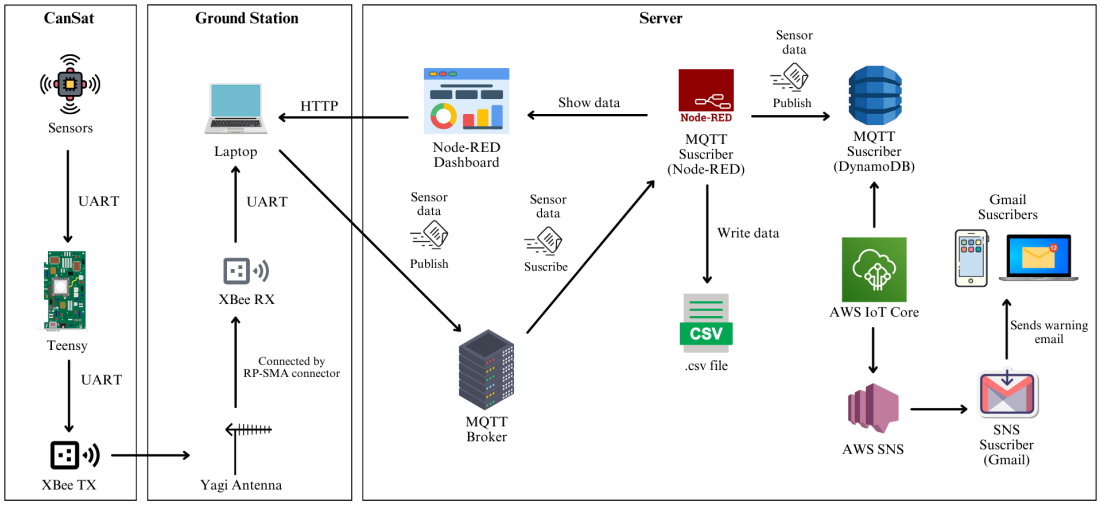


Figure 17. Flow diagram of the data transmission.

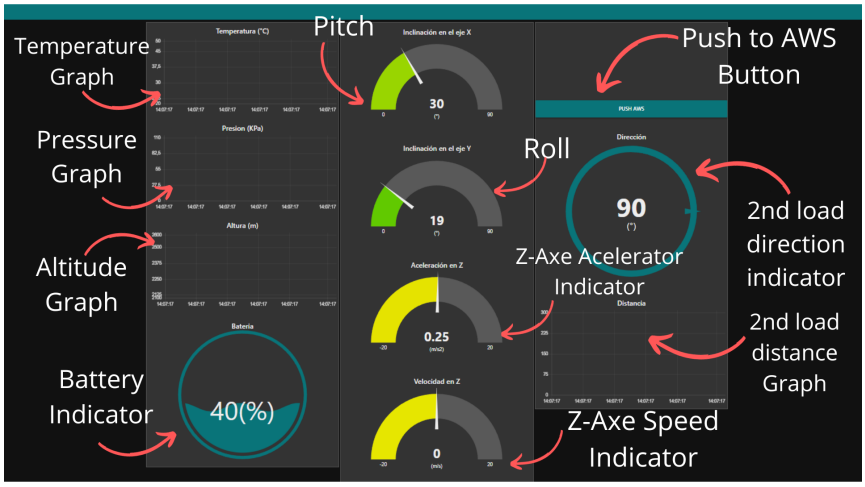


Figure 18. Graphic user interface.

4. Validations and Testing

The objective of this section is to explain the planted test which will evaluate how efficient is the presented design in this study. In addition, tests were done in different locations such as Mexico for the Data Collection test and Perú for the data storage and warning tests.

4.1. Data Collection

For this test, the integrated system of the satellite was implemented, as depicted in Figure 20, assembling both mechanical and electronic components. Subsequently, the CanSat was lifted to an altitude of 400 meters above the ground using a drone and then released in free fall. Meanwhile, the ground station was actively receiving the data from the CanSat.

4.2. Data Storage Test

For this test, it is essential to have already processed the data to verify the functionality of storing the data in an AWS DynamoDB table. As depicted in Figure 21, initially, a first user must press the

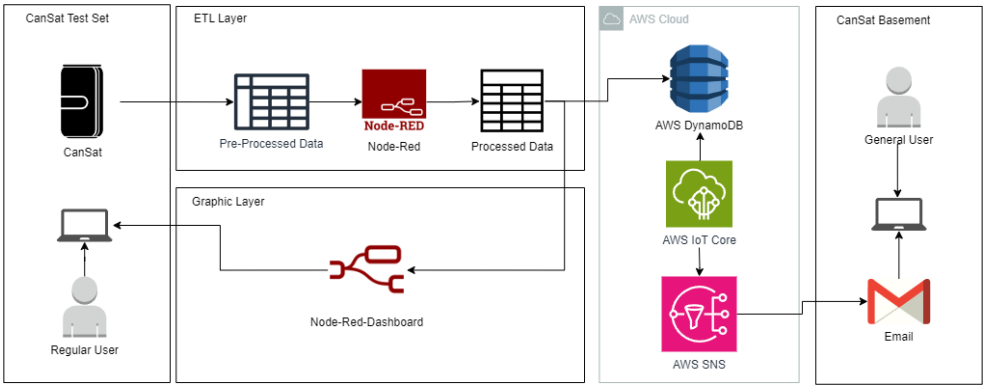


Figure 19. AWS CanSat architecture.

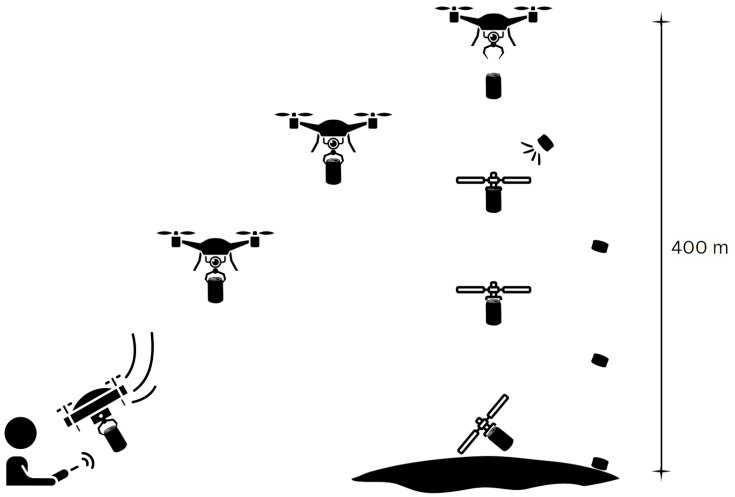


Figure 20. CanSat deployment.

"AWS Button" showed in 18, which, in turn, uploads the data to the cloud, specifically to an AWS IoT Core service. Subsequently, a second user logs into the AWS account where the database is hosted and checks the reception and storage of the summary obtained in the previous test. According to this, the CanSat has registered parameters such as altitude, pressure, and temperature.

4.3. Warning Test

In this final test, enabling the interface to send the recorded data to the cloud using AWS services is crucial. The process commences with the initial user utilizing the interface option to process and subsequently upload the data to the cloud. It is important to emphasize that the input data must include at least one reading with a temperature of 35°C or higher. Subsequently, a second user, who has linked their email account to the AWS SNS Service, must verify the effectiveness of the warning system for an exceeded temperature reading from sensors of a specific CanSat as is shown in Figure 22.

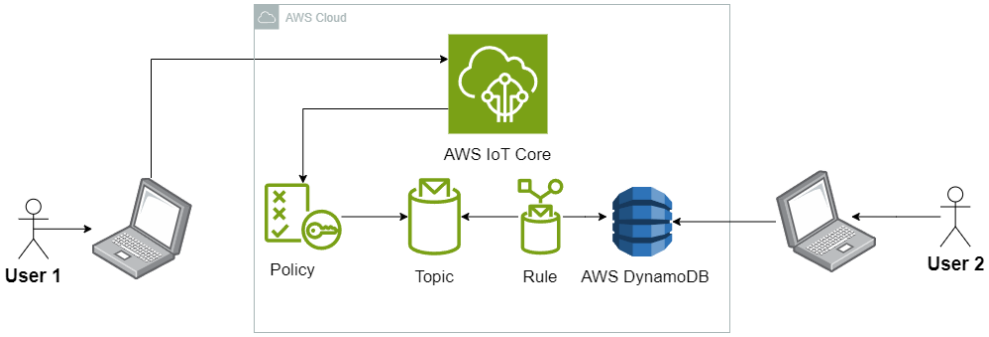


Figure 21. DynamoDB test.

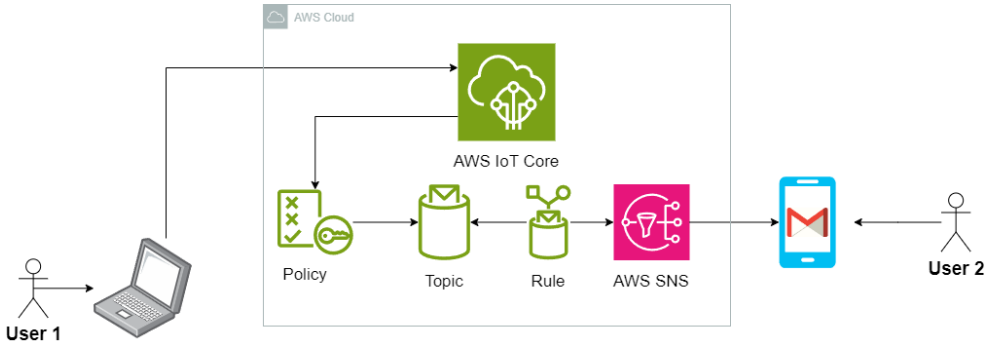
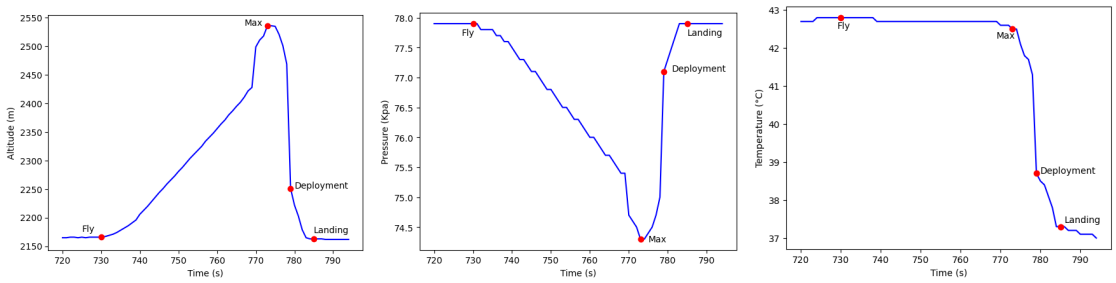


Figure 22. SNS alert test.

5. Results and Discussion

In this section, the results obtained from the tests mentioned in the preceding section are presented. It is important to note that the results are accompanied by a thorough analysis.

Throughout the CanSat mission, data was successfully transmitted and recorded, yielding over 700 readings from the equipped sensors. As illustrated in Figure 23a, the graph depicting altitude over time is aligned to the expected shape of the curve. The four main turning points of the mission can be identified in the graph: flight, drop, deployment and landing. During the lifting phase, between the flight and drop, there is a linear height increase over time that is interrupted at 2425 meters and is resumed in the last seconds before the drop. Another almost vertical behavior is present during the descent, where the altitude goes from 2450 meters to 2250 meters in apparently one second. Considering that the plot was constructed assuming all data was consecutive, with one second between each point, these behaviors are explained due to the loss of some data during this time. In Figure 23b, the descent of pressure is shown as the CanSat ascends, and how it returns to the initial value upon landing. Finally, the temperature curve shown in Figure 23c does not display a logical trend, as the temperature reaches up to 40°C. This is due to the placement of the temperature sensor, which was positioned very close to the microcontroller, causing it to be affected by the considerable amount of heat emitted.



(a) Altitude vs time. **(b)** Pressure vs time. **(c)** Temperature vs time.

Figure 23. Data collected during the mission.

Another important aspect of the mission was the control of the descending speed. During the testing, the CanSat was spinning uncontrollably around its radial axis, which exposed the autogyro subsystem to unfavorable working conditions. A center of mass located close to the bottom of the system would have solved this issue. Figure 24 shows the velocity of the system over time, derived from the altitude graph recorded and, in dashed lines, the ideal velocity. Despite the unexpected behavior, there was a noticeable reduction in the descending speed, reaching 14 m/s at the collision moment.

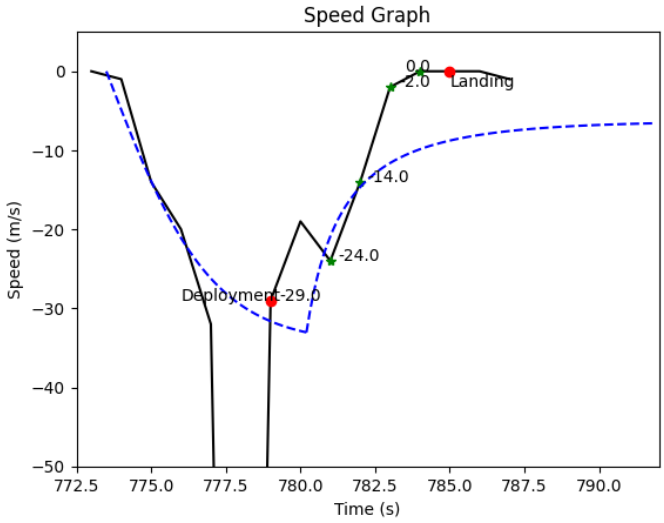


Figure 24. Calculated velocity and ideal velocity vs time graph.

The design contemplated a terminal velocity of 34 m/s during the first phase of the descent, and a final terminal velocity of 6 m/s after the release of the autogyro, as can be observed in the ideal curve. Considering both graphs, the transmission of data must have been interrupted for around 4 seconds, since that interval of time results in an average speed closer to the ideal value. Continuing the analysis, the ideal curve depicts an abrupt change in velocity when the autogyro is deployed, and this tendency is present in the data collected. The effects of the autogyro, which confirm the success of the deployment subsystem, are appreciated during seconds 778 and 779, which correspond to an altitude of approximately 2300 meters, 150 meters above the ground in contrast to the expected 250 meters. Further experiments should be conducted to test the reliability of this subsystem and the robustness of the program. However, the system could not stabilize as the theoretical calculation suggests, presumably due to the uncontrolled spinning and the delay in the autogyro deployment.

According to the data storage explanation provided in the previous chapter, the test was executed following the previously outlined steps. Consequently, the data was successfully stored in the database using the AWS IoT Core and AWS DynamoDB services, as depicted in Figure 25. It is important to mention that, although data such as pressure, temperature, and altitude associated with a CanSat ID and date were added, this database does not store satellite-specific characteristics, such as its battery status, inclinations in different axes, and the position of the secondary payload. This exclusion is intentional to optimize data flow in storage.

id *(Cadena)*	date *(Cadena)*	altitude	pressure	temperature
4	04-01-24, 18:38:45	2670	74	20
4	14-01-24, 12:32:26	2664	74.4	25
2	04-01-24, 18:38:23	2350	76	20
3	14-01-24, 12:38:33	2345	75	23
5	24-12-23, 11:10:58	2264	76.3	27
1	28-12-23, 15:18:36	2246	77.1	26
1	13-01-24, 19:21:54	2175	78.2	24
1	14-01-24, 14:30:46	2172	72	70

Figure 25. DynamoDB Storage.

Regarding the test for alerting in case of detecting an atypical temperature reading, it was implemented similarly to the data storage test. As illustrated in Figure 26, the user received an alert for an atypical data point indicating a temperature reading of 70 degrees Celsius. This was the only atypical data point shown in the database in Figure 25, leading to the conclusion that the requirement to inform the user of an atypical data point was successfully implemented. It is worth mentioning, however, that while the user was alerted, the message format is not entirely user-friendly. Therefore, in future versions of the IoT system, the presentation of alerts will be improved to facilitate the quick interpretation of anomalous data.

Figure 26. SNS Email Alert.

6. Conclusions and Future Works

The presented CanSat accomplished the objective of measuring and transmitting the temperature and pressure of the atmosphere during the mission, along with information about its orientation and acceleration. Despite the loss of some data during the mission, the data recorded was enough to understand the behavior of the Cansat and provided insights for future improvements. First of all, the component location and the structure will be redesigned to place the center of mass closer to the bottom of the system. Second, the mechanisms for the deployment of the primary and secondary load will be improved to ensure its reliability and robustness. Third, the program in the embedded system will be optimized in order to guarantee that every task is executed in the correct time. Fourth, the autogyro mechanism will be tested separately to confirm that it produces a terminal velocity of 6 m/s. Finally, a timestamp will be added to the reception of the data to improve its quality and identify the cause of the data loss.

Regarding the IoT architecture implemented, AWS provides the resources to store and easily manage the data recollected from the CanSat. This allows the project to be escalated for multiple CanSats operating simultaneously, measuring environmental conditions of different places and centralizing the information for further analysis. For future works, the software architecture will become more intricate by incorporating additional AWS services, such as AWS Cognito and AWS OpenSearch. This expansion aims to create a web user interface for monitoring CanSat registrations.

Acknowledgments

We would like to thank the following groups for their financial support in the development of this work. Pontificia Universidad Católica del Perú, Grupo de Robótica PUCP, Faculty of Sciences and Engineering, Core Facilities - FABCORE and Technology Innovation Group (GIT) PUCP. Furthermore, we would like to thank the Runtu Team for their collaboration with this study.

Funding: This research received no external funding.

Author contributions: Introduction, Diego Godinez; State of Art, Nicole Perez and Diego Godinez; Mechanical Design, Victor Huayapa, Diego Godinez and Abel Hinostroza; Electrical Design, Franco Rivadeneira and Kioshi Kiyan; Software Design, Franco Rivadeneira and Nicole Perez; Validations and Testing, Franco Rivadeneira; Results and Discussion, Kioshi Kiyan and Franco Rivadeneira; Conclusion and Future Works, Kioshi Kiyan and Franco Rivadeneira.

Disclosure statement: The authors declare no conflict of interest.

References

[1] Yasuyuki Miyazaki and Mohammed Khalil. Cansat pico size artificial satellite, 2017.

[2] Michał Ostaszewski, Kazimierz Dzierzek, and Łukasz Magnuszewski. Analysis of data collected while cansat mission. In *2018 19th International Carpathian Control Conference (ICCC)*, pages 1–4, 2018.

[3] Tanvir Hasan Raian, Jahirul Islam, Saiful Islam, Rafiul Azam, and Sutapa Debnath Jahidul Islam. An affordable cansat design and implimentation to study space science for bangladeshi students. *2020 IEEE Region 10 Symposium (TENSYMP)*, pages 1205–1208, 6 2020.

[4] M C Ferraz, M C Pereira, M Greco, and E Peiró. Design of a generic platform for an educational cansat. In *Proc. of the 1st IAA Latin American Symp. on Small Satellites*, 2017.

[5] Carrington Chun, M. Hassan Tanveer, and Sumit Chakravarty. The cansat compendium: A review of scientific cansats. *Machines*, 11(7), 2023.

[6] N Sreenivasaraja, Mr J Jeffrey Jackson Raj, B Nandhagobalan, and S Kavin. Design and fabrication of advance level cansat used for measuring the atmospheric parameters and gps tracking system. 4:1264–1269, 2 2018.

[7] Shankar Bhattarai, Ji-Seong Go, and Hyun-Ung Oh. Experimental cansat platform for functional verification of burn wire triggering-based holding and release mechanisms. *Aerospace*, 8, 7 2021.

[8] Angel Colin. A pico-satellite assembled and tested during the 6th cansat leader training program. *Journal of Applied Research and Technology*, 15:83–91, 2 2017.

[9] Efren Bautista-Linares, Enrique A. Morales-Gonzales, Mario Herrera-Cortez, Esther A. Narvaez-Martinez, and Jaime Martinez-Castillo. Design of an advanced telemetry mission using cansat. In *2015 International Conference on Computing Systems and Telematics (ICCSAT)*, pages 1–4, 2015.

[10] Oscar Alberto Jaramillo, Raúl Camacho Briñez, and Juan Camilo Tejada. Design of a cansat for measuring environmental variables. *Publicaciones e Investigación*, 13:31–38, 7 2019.

[11] Shindy Atila Putri Merlyn Inova Christie Latukolan Edwar Kusmadi Rizki Pratama Ramadhan, Aditya Rifky Ramadhan. Prototype of cansat with auto-gyro payload for small satellite. *2019 IEEE 13th International Conference on Telecommunication Systems, Services, and Applications (TSSA)*, pages 243–248, 10 2019.

[12] Prakhar Shukla, Raj Mishra, Uzair Ahmad Sardar, and B. Mohapatra. Satellite design for cansat with autorotatig payloads. *2022 4th International Conference on Advances in Computing, Communication Control and Networking (ICAC3N)*, pages 2385–2392, 10 2022.

[13] Carrington Chun, Uday Patel, M. Hassan Tanveer, Kevin Dallesasse Tom Swift, and Sumit Chakravarty. Crafting cansats: A novel modular design paradigm for scientific cansats. *2023 IEEE 20th International Conference on Smart Communities: Improving Quality of Life using AI, Robotics and IoT (HONET)*, pages 68–72, 12 2023.

[14] Rob Wojcik, Felix Bachmann, Len Bass, Paul Clements, Paulo Merson, Robert Nord, and Bill Wood. Attribute-driven design (add), version 2.0 software architecture technology initiative. 2006.

[15] PEU. Guía de la misión espacial concurso iberoamericano satélites enlatados 2023, 2023.

[16] Airfoil Tools. Naca 4 digit airfoil database search.

[17] Alejandro Ventura, Johan Nuñez-Quispe, Godo Sanchez, and Jafet Santivañez. Cansat payload with autogyro for descent in experimental rocket flights: Development, cfd analysis, and preliminary test on free-fall. *Journal of Physics: Conference Series*, 2235, 5 2022.

[18] Sultan Nur Bulut, Mahircan Gül, Can Beker, İbrahim İlge İpek, Ömer Eren Can Koçulu, Çınar Topaloğlu, Nurullah Dinçer, Ahmet Kırlı, Hasan Fatih Ertuğrul, and Celal Sami Tüfekci. Model satellite design for cansat competition. In *2013 6th International Conference on Recent Advances in Space Technologies (RAST)*, pages 913–917, June 2013.

[19] P.N. Shankaranarayanan, Ashiwan Sivakumar, Sanjay Rao, and Mohit Tawarmalani. Performance sensitive replication in geo-distributed cloud datastores. In *2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pages 240–251, 2014.

[20] Juan García Puertas. *Edge Computing in IoT: A Study on AWS IoT Greengrass and DynamoDB*. PhD thesis, Universitat Politècnica de València, 2023.

[21] Justin Waterman, Hyeongjun Yang, and Fadi Muheidat. Aws iot and the interconnected world – aging in place. In *2020 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 1126–1129, 2020.

[22] Nadia Imtiaz Jaya and Md. Farhad Hossain. A prototype air flow control system for home automation using mqtt over websocket in aws iot core. In *2018 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, pages 111–1116, 2018.

[23] Aman Kumar Singh, Abdullah Ahmed Arifi, R Harikrishnan, Bhuvi Datta, and Shivali Amit Wagle. Iot based home automation using app aws. In *2022 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI)*, pages 1–9, 2022.