Federal University of Technology - Paraná - Brazil - UTFPR
Academic Department of Electronics – DAELN
Academic Department of Informatics – DAINF
Computer Engineering Course
Integration Workshop 3 (ELEX22) – S71 – 2025/1

Technical report

SoyIA: Automated Soybean Grain Classifier

João Henrique Campos Soares
Lucas Emanuel Veiga Marques
Marcos Vinicius Tanganelli Lara
Nicolas Barbieri Sousa
Robson Borges dos Santos
Rudinei Kuznier Silva

June - 2025

Summary

Abstract

The SoyIA project presents the development of an automated soybean grain classifier designed to improve the manual classification process currently used in Brazil, as established by the Ministry of Agriculture. The system integrates mechanical, electronic, and software components to capture, process, and analyze soybean samples. It uses computer vision with a YOLOv11 model running on a Raspberry Pi 5 to identify different grain classes, including good, moldy, burnt, and smashed seeds. The structure includes a sieving mechanism for impurity removal, an analysis surface with vibration control, and an intuitive web interface developed in React for user interaction. A MySQL database stores classification results, while communication with an ESP32 microcontroller handles motor control and IMU data for surface leveling. Despite the inability to gather a sufficient number of germinated seeds due to poor planting conditions, the system successfully met nearly all functional requirements. Future improvements include dataset expansion and enhanced data augmentation techniques to increase model robustness and support additional grain classes.

1. Introduction

Agribusiness is an important sector of the Brazilian economy, accounting for nearly 30% of the GDP. Its share has been growing over the past decades, and one product in particular stands out within agriculture: soybeans. Brazil is the world's largest producer, having produced 167.87 million tons in 2024, representing a total of 53.942 billion dollars in exports [5]. In this context, the Ministry of Agriculture establishes regulations for the classification of soybeans that are traded, dividing them into those intended for raw consumption and those for other uses [14]. Currently, the classification process is carried out manually, which can lead to inconsistent or biased results.

1.1 Proposed solution

In order to mitigate the issues arising from the manual classification of soybean grains, SoyIA: An Automated Soybean Grain Classifier was developed. This system uses computer vision to evaluate the grains in a sample and also separates impurities from the grains through a sieving process. Figure 1 shows the overall operation: a user connected to the system's Wi-Fi network accesses the address soyia.local through any web browser. The user pours the seeds to be classified onto the sieve and starts the classification process by clicking the "Begin analysis" button on the website. After that, the system filters out small impurities and classifies the grains. It is necessary to wait for the system to finish analyzing the entire sample; the results can then be viewed, showing the quantities of good, moldy, crushed, and burnt seeds. The user can also view the photos taken during the process, as well as the model's inferences about them.

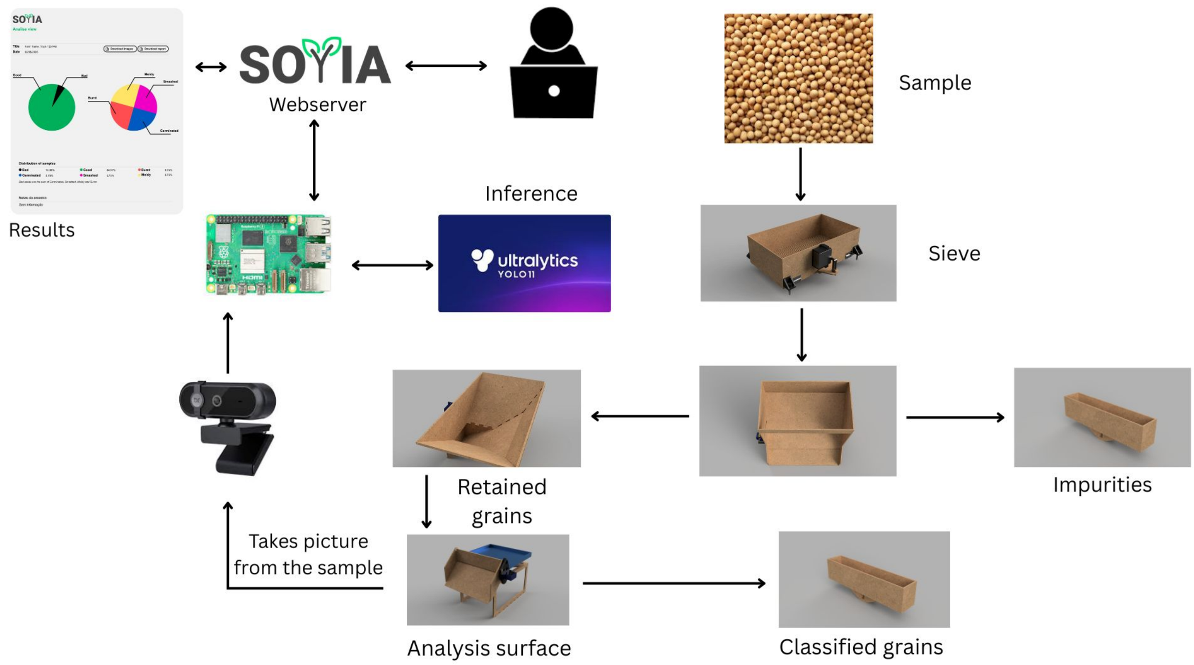

Figure 1: Block diagram.

2. Project specification

2.1 Requirements

This section presents the main requirements established by the SoyIA project team during the planning phase, taking into account the available resources and the intended system functionalities. The requirements were divided into functional and non-functional categories and organized according to the system's subsystems: software, mechanical, and hardware. Each requirement was carefully defined to ensure that the system complies with the soybean grain classification process established by the Ministry of Agriculture.

2.1.1 Mechanical requirements

Requirement ID	Description
Mechanical FR 1	The system must receive the seeds in an input compartment.
Mechanical FR 2	The system must sieve the sample.
Mechanical FR 3	The system must separate impurities from the sample.
Mechanical FR 4	The system must distribute the seeds on an analysis surface in such a way that none are on top of each other when the photo is taken.
Mechanical FR 5	The system must distribute the seeds on an analysis surface in controlled quantities.
Mechanical FR 6	The system must remove seeds from the analysis surface after classification.
Mechanical FR 7	The system must dump the already processed seeds into a reservoir for the user to remove them.

Table 1: Main Mechanical Functional Requirements

2.1.2 Hardware requirements

Requirement ID	Description
Hardware FR 3	The system must capture images of soybean seeds for classification.
Hardware FR 4	The system must illuminate the surface to capture photos of soybean seeds.
Hardware FR 5	The system must control vibration mechanisms.

Table 2: Main Hardware Functional Requirements

2.1.3 Software requirements

Requirement ID	Description
Software FR 1.1	The system must identify defect-free seeds.
Software FR 1.2	The system must identify moldy seeds.
Software FR 1.3	The system must identify burnt seeds.
Software FR 1.4	The system must identify smashed seeds.
Software FR 2	The system must allow the user to register a sample to be classified.
Software FR 3	The system must list the classified samples.
Software FR 4	The system must generate a classification report for each sample.

Continued on next page

Table 3 – continued from previous page

Requirement ID	Description
Software FR 5	The system must show statistics of classified samples.
Software FR 6	The system must indicate whether the system is leveled.
Software FR 7	The system must increase the model's accuracy by 85%.

Table 3: Main Software Functional Requirements

3. Development

3.1 Mechanical development

The mechanical design was created using the Fusion 360 software [2]. The mechanical system was divided into six parts: the sieve (Figure 2), the impurity separator (Figure 3), the dispenser (Figure 4), the analysis surface (Figure 5), and the output compartment (Figure 9). The webcam is attached to the structure along with the LED strips (Figure 8). The assembled structure is shown in Figures 6 and 7.

The sieve has a 12V vibration motor mounted underneath to perform the shaking process. Since the motor is eccentric, its rotation causes the surface to vibrate, separating dirt from the grains. At this stage, the separator has its ramp tilted toward an output compartment, where the impurities are discarded.

Right after that, the ramp opens to let the grains fall into the dispenser. Once the sieving process is complete, the sieve surface is opened by two servo motors. The dispenser also has a servo motor at its bottom, which allows the seeds to drop in a controlled manner onto the analysis surface.

The analysis surface is equipped with a 5V vibration motor underneath, enabling the grains to spread out and preventing overlap. A servo is attached to the analysis surface so that it can be tilted to direct the seeds into the output compartment.

It is important to note that the vibrating mechanical parts are attached to the structure using springs and 3D-printed supports to ensure an efficient shaking process. Nearly the entire mechanical structure was assembled using 3mm MDF sheets, which were laser-cut.

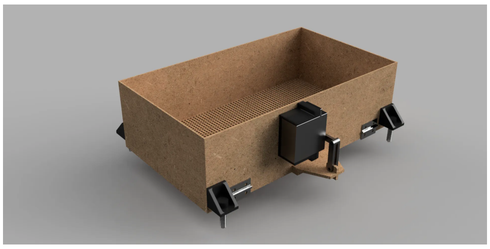

Figure 2: Sieve.

Figure 3: Impurity separator.

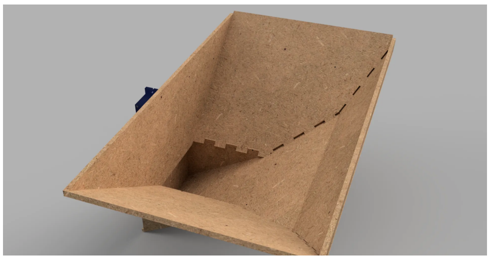

Figure 4: Dispsenser.

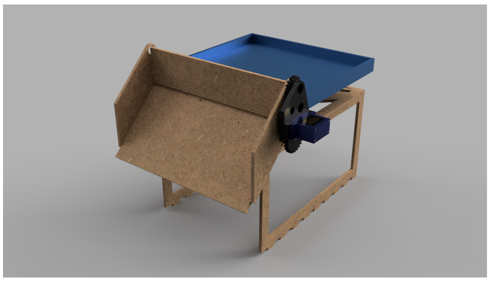

Figure 5: Analysis surface.

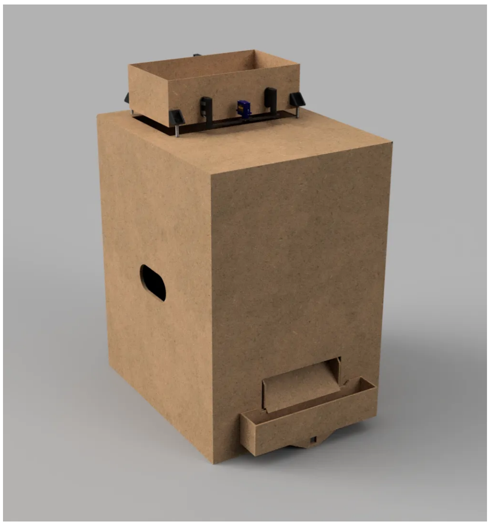

Figure 6: Assembled structure (front view).

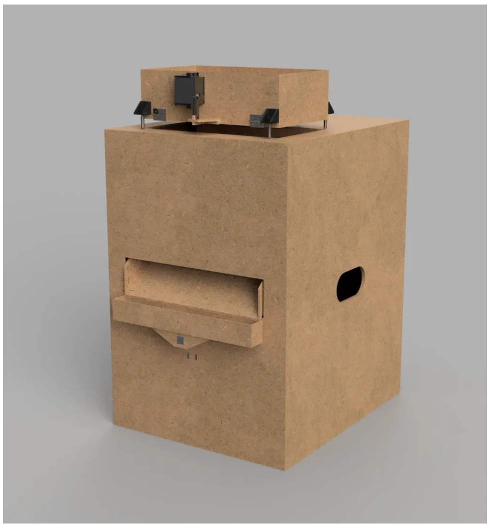

Figure 7: Assembled structure (rear view).

Figure 8: Camera and led strips.

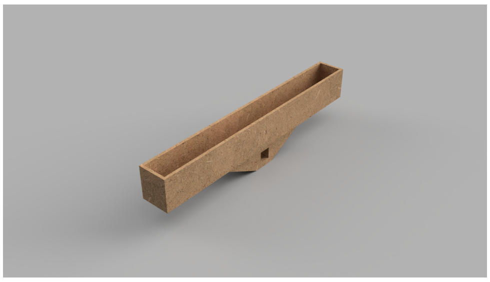

Figure 9: Output compartment.

3.2 Electronic development

The electronic system is responsible for motor control, synchronization of photo capture, and system lighting and leveling. The system is powered by a 12V/20A power supply. Since most components operate at 5V, two 12V to 5V step-down converters were added: one for powering the logic circuit (Raspberry Pi 5 [12], ESP32 Dev Kit V1 [6], WC577 2k webcam [15] and a MPU-6050 IMU [10]), and another for powering the analog circuit (motors and LED strips). The Figures 10, 11 and 12 show the portion of the schematic related to the system's power electronics.

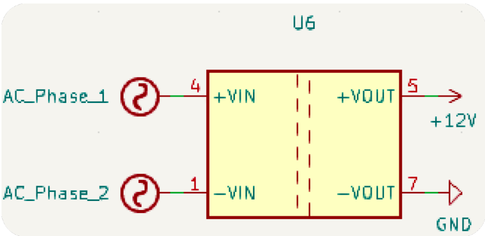

Figure 10: Power supply.

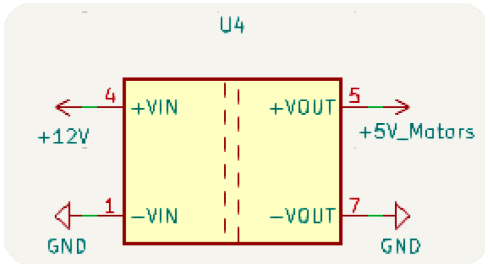

Figure 11: Step down 3A (12V to 5V).

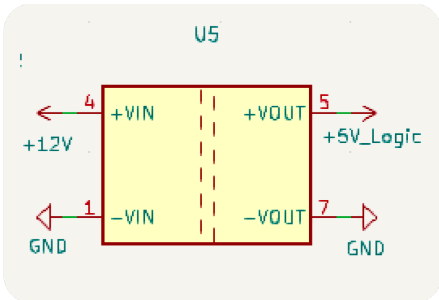

Figure 12: Step down 6A (12V to 5V).

The Raspberry Pi 5 is powered by the output of the higher-current step-down converter. Connected to it via USB ports are the webcam and the ESP32—both of which are powered directly by the Raspberry Pi. A communication protocol was developed between the Raspberry Pi and the ESP32 to synchronize activities such as starting an analysis, requesting images, and ending an analysis. An IMU was connected to the ESP32 via the I2C bus to indicate whether the mechanical structure is level. The Figures 13 and 14 show a portion of the schematic related to the digital circuit.

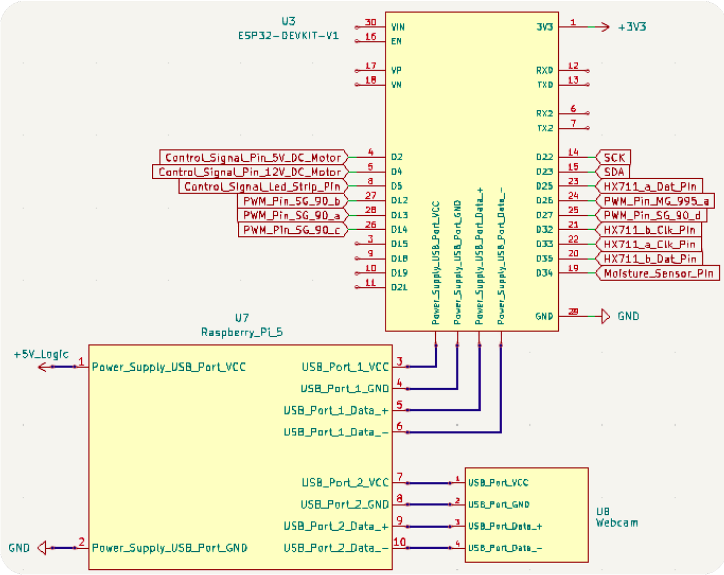

Figure 13: Digital devices.

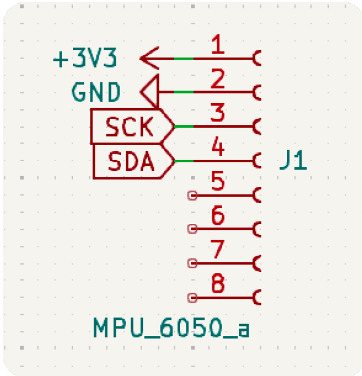

Figure 14: IMU.

The servo motors are powered by the lower-power step-down converter and are controlled by the ESP32 to operate the structure's mechanisms. The Figure 15 shows this connection. A DC motor power control circuit was developed using a MOSFET. A flyback diode is placed in parallel with the inductive load to prevent current spikes in the circuit. Additionally, pull-down resistors and current-limiting resistors are present at the transistor's gate. Similarly, the same control circuit is used for the LED strips in order to regulate their brightness when capturing an image. The Figures 16, 17 and 18 show these circuits.

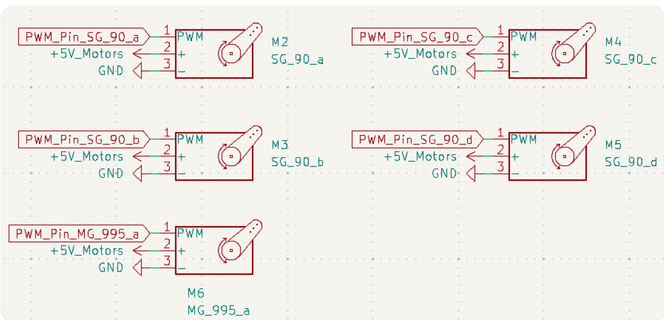

Figure 15: Servo motors.

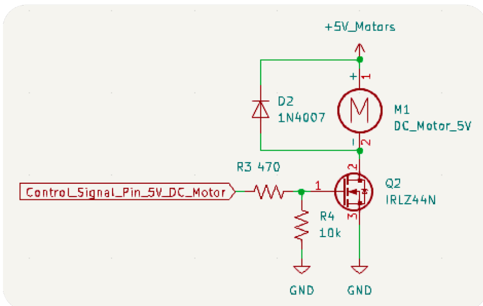

Figure 16: 5V DC motor control circuit.

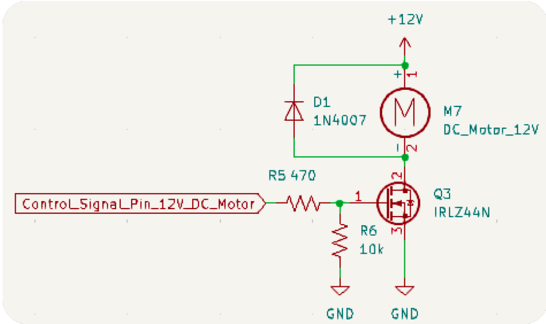

Figure 17: 12V DC motor control circuit.

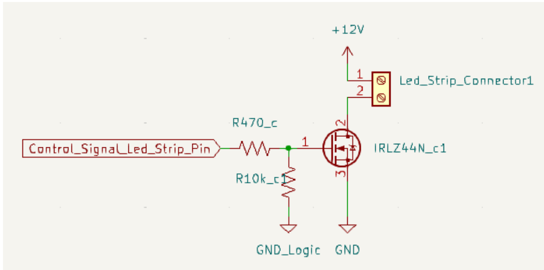

Figure 18: Led strip control circuit.

For the circuit assembly, a PCB was designed using the KiCad [8] software. The routing between terminals was done exclusively on the bottom layer, and notations were added on the top layer. The board was manufactured using ferric chloride and acetone. The final result is shown in the Figures 19 and 20.

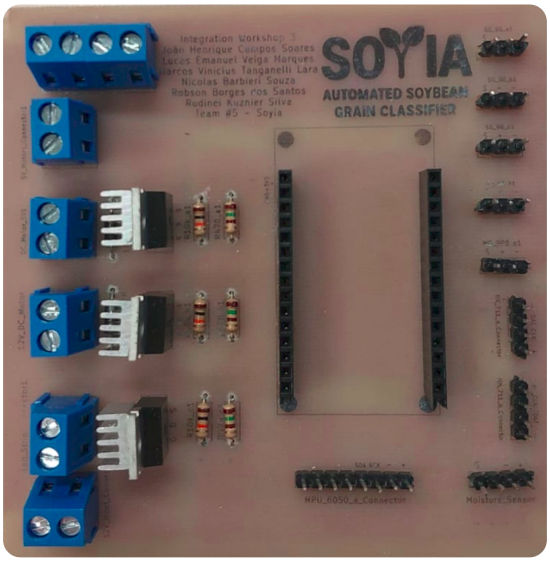

Figure 19: PCB (top view).

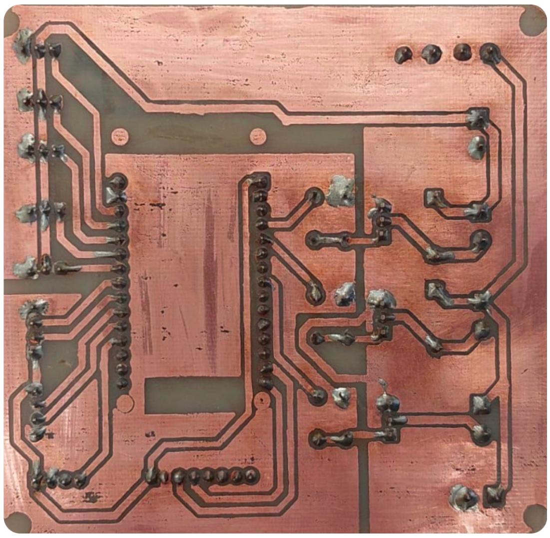

Figure 20: PCB (bottom view).

The deployment diagram is shown in the Figure 21. It illustrates how the devices are interconnected at a high level, as well as the communication protocols.

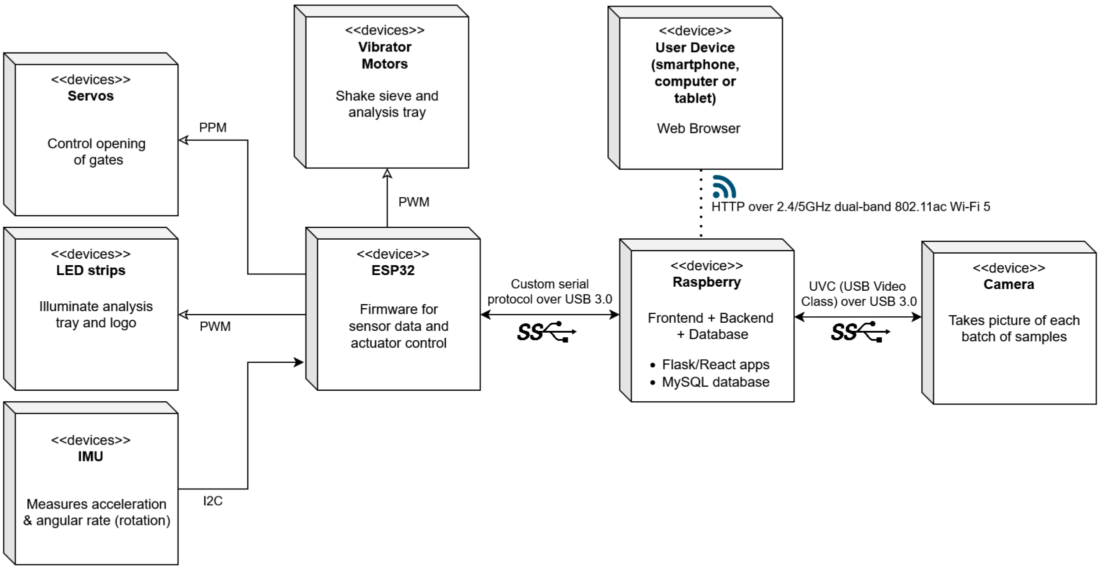

Figure 21: Deployment diagram.

3.3 Firmware development

The firmware designed for the ESP32 was developed in C++ [4] using the Arduino IDE [1]. The program was built with the help of FreeRTOS [7], a real-time operating system. As previously mentioned, this microcontroller is responsible for reading data from the IMU, controlling the motors and LED strips, and handling communication with the Raspberry Pi. Since only one of the ESP32's serial interfaces is used, a mutex was required to manage access to this resource. There are three threads running:

- **Serial thread**: responsible for detecting incoming data on the ESP32's serial interface.
- **IMU thread**: responsible for reading the IMU's gravitational vectors in real time and sending the data via serial.
- **Main thread**: responsible for initiating the analysis process when requested by a user.

Figure 22 shows the main thread flowchart.

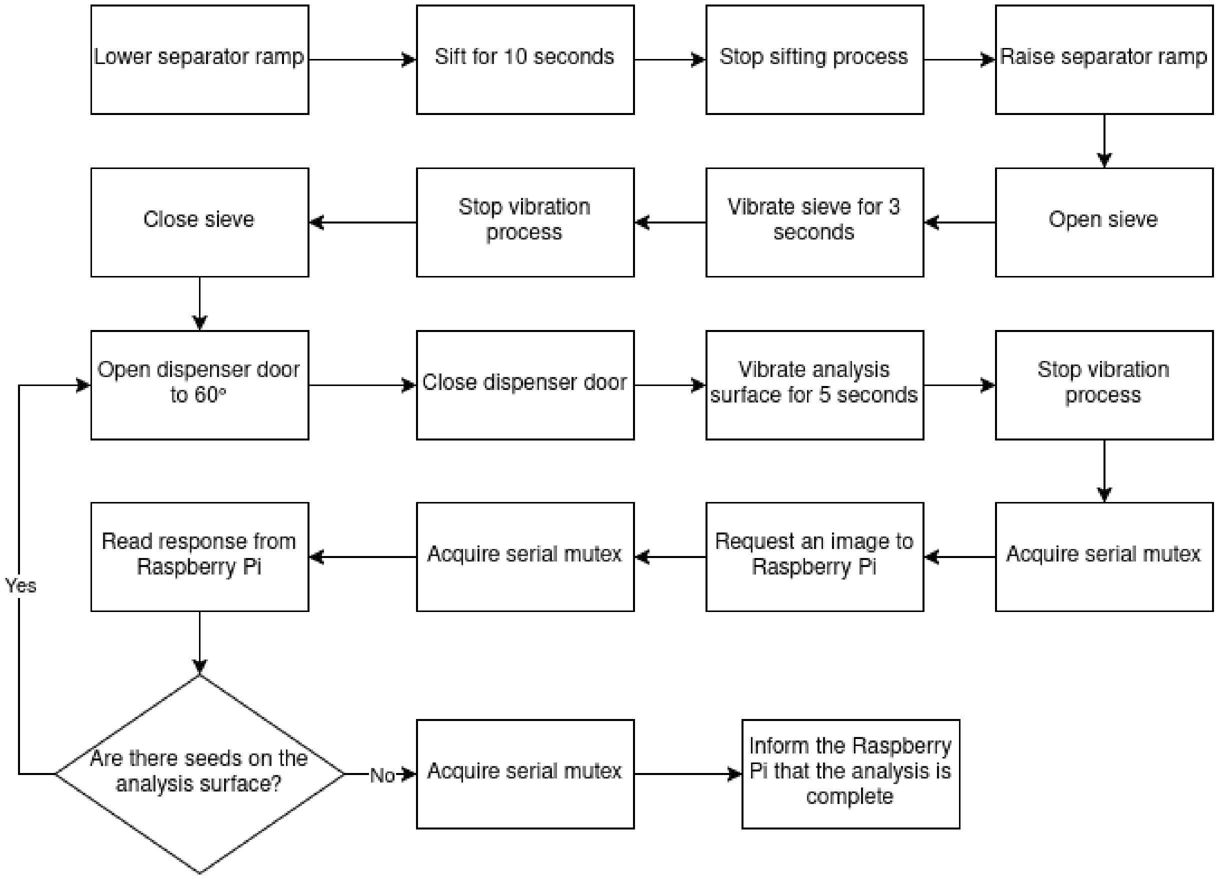

Figure 22: Main thread flowchart.

3.4 Website development

The website served as the human-machine interface developed for the system. Through it, the user can start a new analysis and view previous ones. To access it, the user must use a smartphone, computer, or tablet and connect to the soyia Wi-Fi network. After that, the interface can be accessed at the address soyia.local via a web browser.

The frontend was developed using React [13] and Bootstrap [3].

The Figure 23 shows the website's home page, where the user has the option to start a new analysis or view previous analyses. To view past analyses, simply click the "Reports" button and the data will be shown (Figure 24).

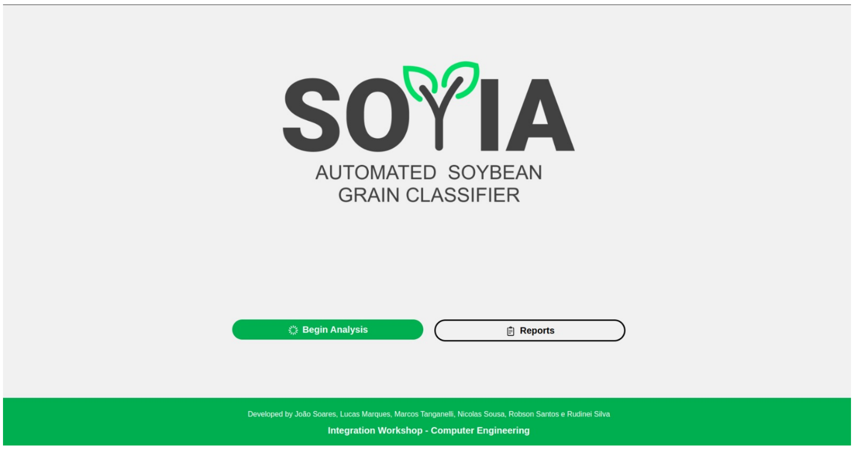

Figure 23: Home page.

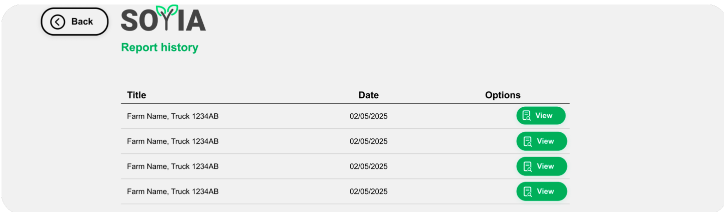

Figure 24: Previous reports.

By clicking the "Begin analysis" button, the user will be taken to a screen that indicates whether the structure is leveled (Figure 26) or not (Figure 25), providing instructions on how to start the process. It is necessary to enter a name for the sample to be analyzed.

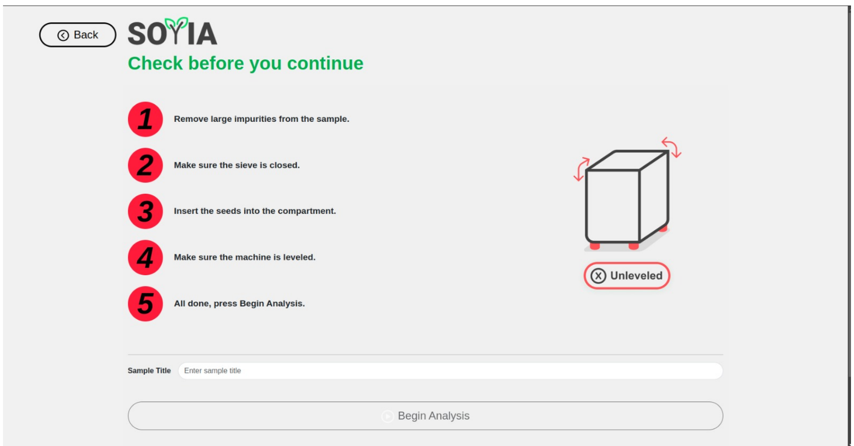

Figure 25: Structure unleveled.

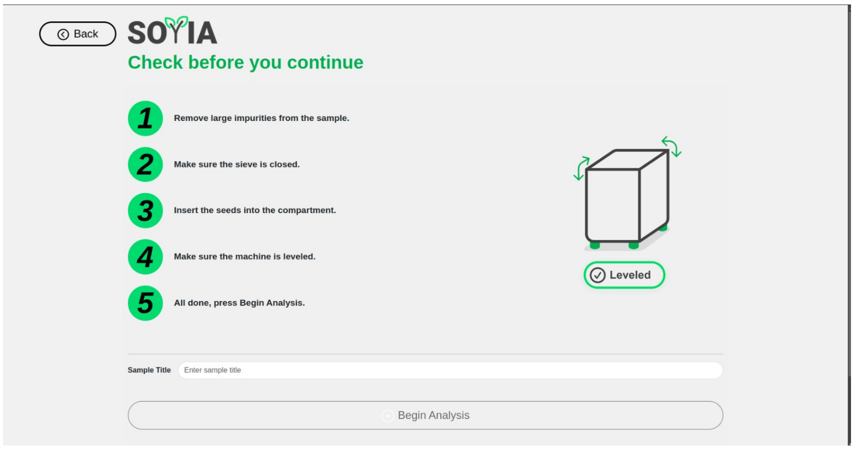

Figure 26: Structure leveled.

When the analysis starts, the website will indicate when it is complete, allowing the user to view the results. These steps are illustrated in the Figures 27 and 28.

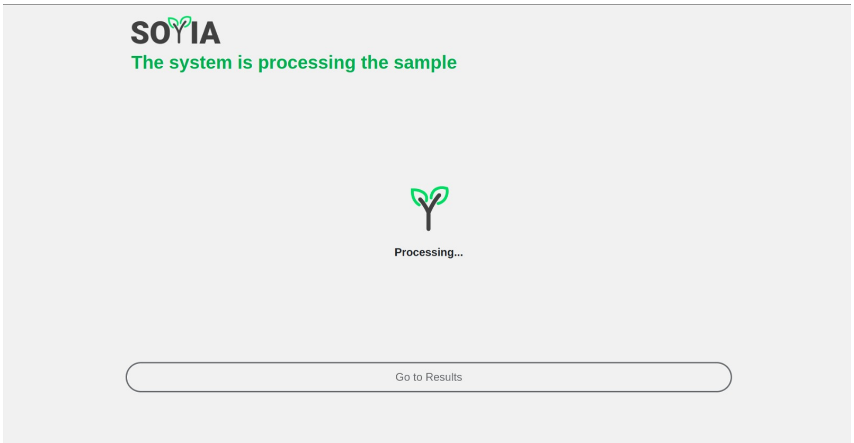

Figure 27: Analysis in process.

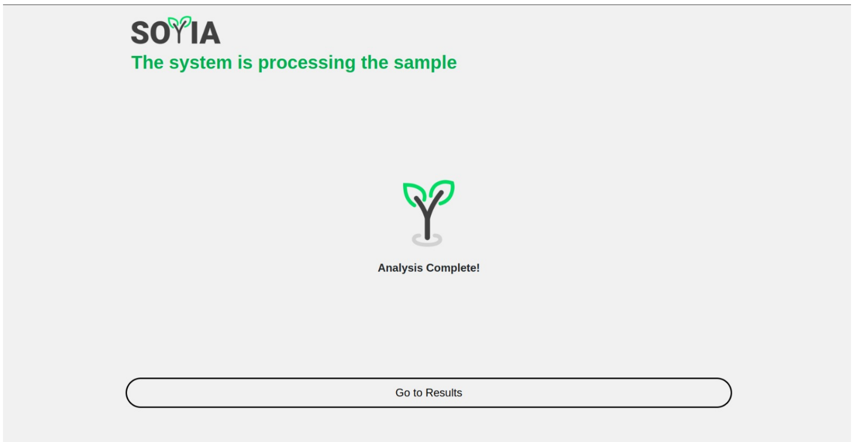

Figure 28: Analysis finished.

The results include the analyzed quantities of each class present in the sample (good, burnt, moldy, and crushed seeds), displaying charts for each (Figure 29). Additionally, the user can view the images taken during the process as well as the model's inference on them (Figures 30 and 31).

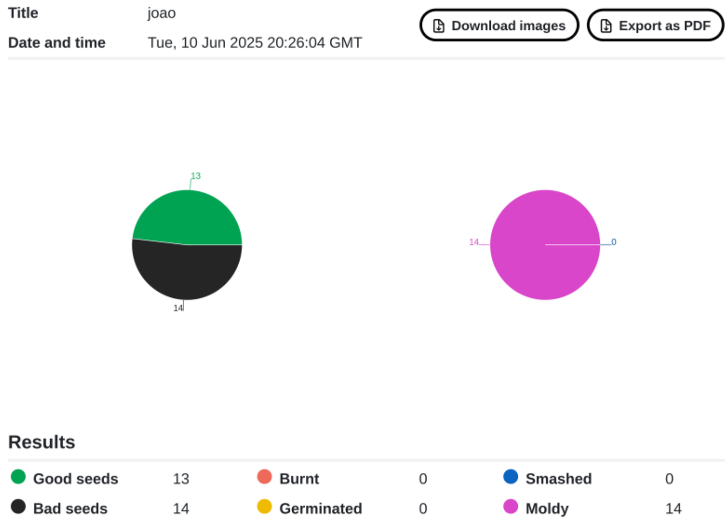

Figure 29: Results.

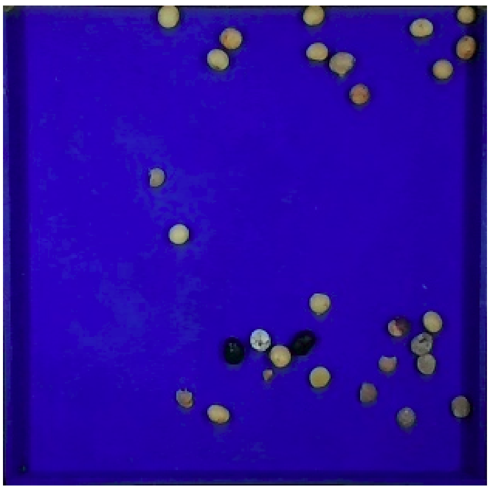

Figure 30: Image captured from the sample.

Figure 31: Model inference on the image.

The backend of the application was developed using Python, with a web server running on port 80 of the Raspberry Pi. Upon initialization, the Raspberry Pi establishes serial communication with the ESP32, with a dedicated thread responsible for receiving events from the microcontroller. There are two main commands sent from the ESP32 to the Raspberry Pi: start an analysis and IMU data.

- **Start an analysis**: A new thread is created in the backend to synchronize activities between the Raspberry Pi and the ESP32. Essentially, the Raspberry Pi waits to capture a snapshot of the analysis surface, cropping it so that only the blue part of the surface is included in the image. After that, the YOLO object performs inference on the image, counting each of the identified classes. At the end of the process, when there are no more seeds to be analyzed, the analysis data is saved in the application's MySQL database [11].

- **IMU data**: Periodically, the ESP32 sends data about the leveling of the surface to the Raspberry Pi. This data is displayed to the user before starting a new analysis.

3.5 Model inference development

One of the most important parts of the project was the development of the inference model for the sample images. The neural network used was YOLOv11 [16], as it performs well on devices like the Raspberry Pi 5. The dataset used for training was created by the authors and contains 129 images and over 4,000 seeds in total, covering all the mentioned classes: good seeds (Figure 32), smashed seeds (Figure 34), burnt seeds (Figure 35) and moldy seeds (Figure 33). For the semantic segmentation process, the LabelMe [9] software was used, and the labeling process is shown in the Figure 36. Data augmentation was applied to the dataset by adjusting the brightness of some images so the model could perform well under various conditions. The training process was carried out over 150 epochs, and the model's results are shown in the Figures 37, 38.

Figure 32: Good seeds.

Figure 33: Moldy seeds

Figure 34: Smashed seeds.

Figure 35: Burnt seeds.

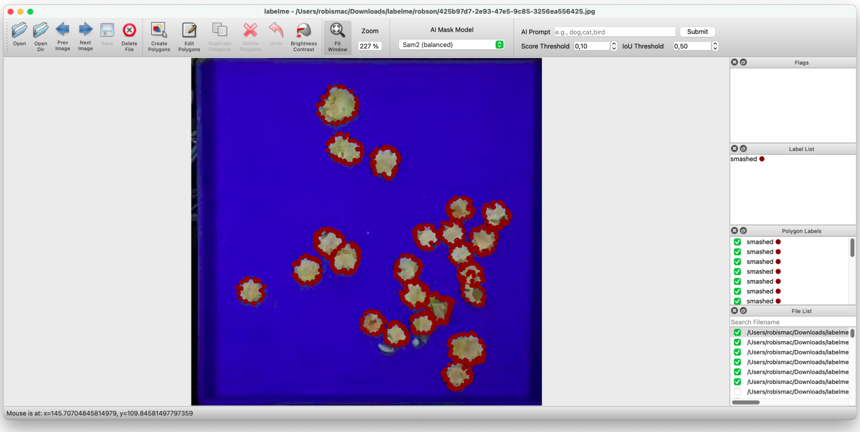

Figure 36: Labeling process using LabelMe.

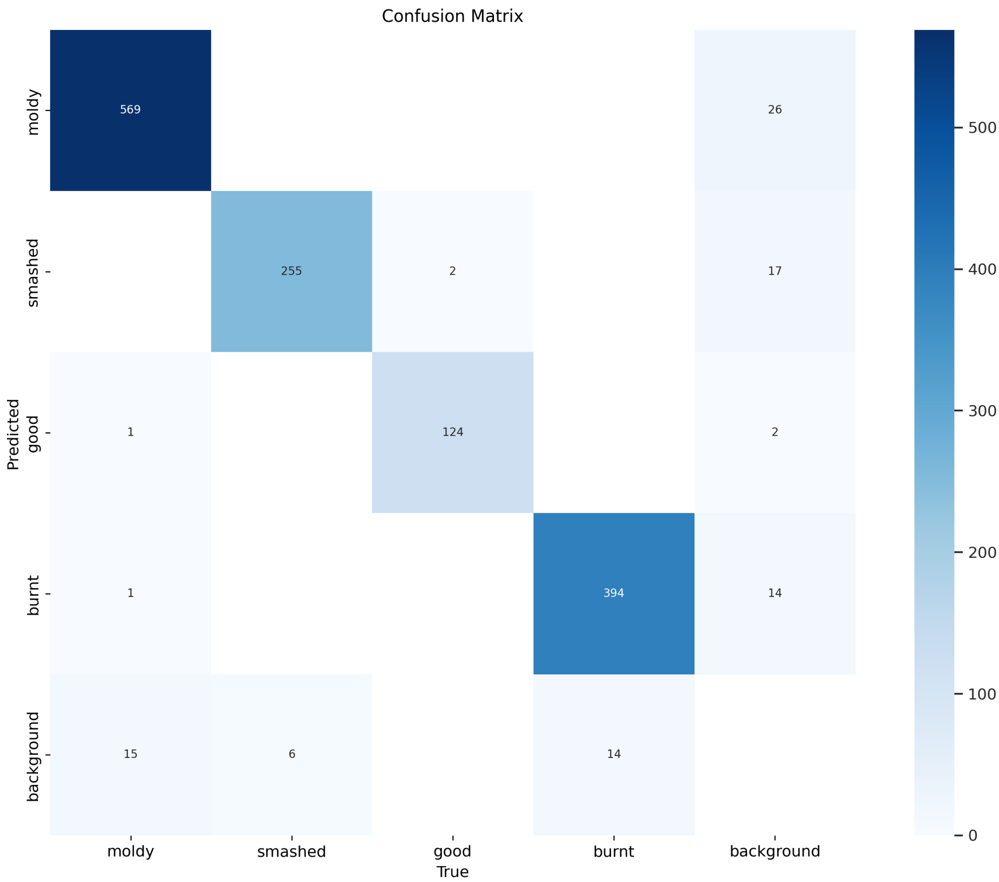

Figure 37: Confusion matrix.

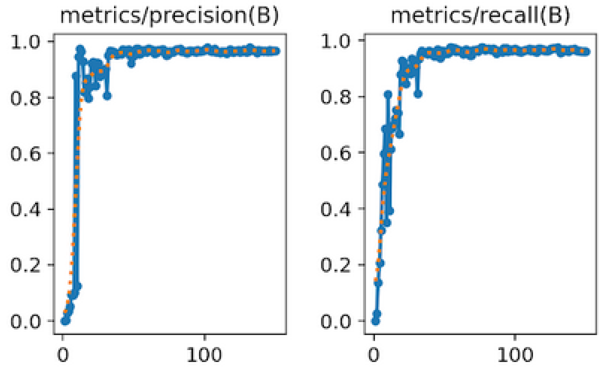

Figure 38: Precision and recall.

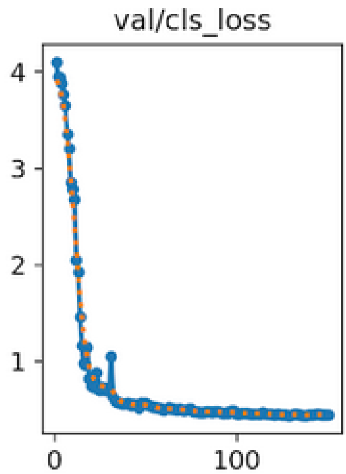

Figure 39: Loss.

It is noticeable that the model had a satisfactory learning curve, as the precision and recall values are close to 1, indicating that it makes few incorrect inferences (false positives or false negatives). Additionally, the loss value is close to 0, which indicates that the model correctly predicts the position of objects in the image as well as the bounding boxes. The confusion matrix confirms these results.

4. Results

4.1 Budget

The allocated budget was not fully used because most of the materials (especially the more expensive ones) were already owned or borrowed by the authors. The materials are shown in the Figure 40.

Name	Quantity	Value	Total
LED strip	1	R$10.83	10.83
Load cell sensor	2	R$9.00	18
Raspberry Pi 5 8GB RAM	1	R$563.00	563
PLA Filament 1kg	1	R$69.00	69
3mm MDF boards	16	R$8.69	139.04
Micro servo motor SG90 9g	6	R$11.30	67.8
RF370 Vibration Motor 12V	1	R$36.77	36.77
Webcam 2K, USB, WC577, Bright	1	R$160.00	160
Moisture Capacitive Sensor	1	R$38.00	38
Switching source 12V/30A	1	R$53.42	53.42
ESP32	1	R$37.00	37
MPU-6050 gyroscope	2	R$15.12	30.24
HX711 amplifier	1	R$4.65	4.65
Step down converter 5V/6A	1	R$6.51	6.51
H Bridge motor driver	1	R$14.15	14.15
Miscellaneous	1	R$100.00	100
Motor 5V OT-1762	1	R$2.69	2.69
			SUM 1351.1

Figure 40: Budget

4.2 Functional requirement completion

The team was unable to fulfill one of the functional requirements: the system should be able to identify germinated seeds. This was due to poor planting conditions for their production — in the end, all of them ended up molding. Therefore, it was not possible to obtain a significant number of germinated seed samples to build the entire dataset. Despite this, the remaining mandatory requirements were completed.

5. Conclusions

The project aimed to develop an automatic soybean grain classifier that complies with part of the process established by the Ministry of Agriculture in order to improve the current manual procedure. The project is in a functional state, meeting almost 100% of the established requirements. As future improvements, the team can apply different types of data augmentation and evaluate how the model performs. Moreover, according to the Ministry of Agriculture, there are more grain classes that can be analyzed within a sample, such as damaged, broken, shriveled grains, etc. An improvement to the system would be to collect various types of grains from these other classes and expand the dataset so the model becomes more efficient.

References

[1] *Arduino IDE*. 2025. URL: https://www.arduino.cc/en/software (visited on 06/25/2025).

[2] Autodesk. *Autodesk Fusion*. 2025. URL: https://www.autodesk.com/br/products/fusion-360/overview (visited on 06/25/2025).

[3] *Build fast, responsive sites with Bootstrap*. 2025. URL: https://getbootstrap.com/ (visited on 06/25/2025).

[4] *C++*. 2025. URL: https://cplusplus.com/ (visited on 06/25/2025).

[5] Empresa Brasileira de Pesquisa Agropecuária. *Soja em números (safra 2024/25)*. 2025. URL: https://www.embrapa.br/soja/cultivos/soja1/dados-economicos (visited on 06/25/2025).

[6] *ESP32: A feature-rich MCU with integrated Wi-Fi and Bluetooth connectivity for a wide-range of applications*. 2025. URL: https://www.espressif.com/en/products/socs/esp32 (visited on 06/25/2025).

[7] *FreeRTOS™: Real-time operating system for microcontrollers and small microprocessors*. 2025. URL: https://www.freertos.org/ (visited on 06/25/2025).

[8] *KiCad: A Cross Platform and Open Source Electronics Design Automation Suite*. 2025. URL: https://www.kicad.org/ (visited on 06/25/2025).

[9] *LabelMe: Build datasets for AI with Private and Flexible annotation*. 2025. URL: https://labelme.io/ (visited on 06/25/2025).

[10] *MPU-6050: Six-Axis (Gyro + Accelerometer) MEMS MotionTracking™ Device*. 2025. URL: https://invensense.tdk.com/products/motion-tracking/6-axis/mpu-6050/ (visited on 06/25/2025).

[11] *MySQL*. 2025. URL: https://www.mysql.com/ (visited on 06/25/2025).

[12] *Raspberry Pi*. 2025. URL: https://www.raspberrypi.com/ (visited on 06/25/2025).

[13] *React: The library for web and native user interfaces*. 2025. URL: https://react.dev/ (visited on 06/25/2025).

[14] Serviço Nacional de Aprendizagem Rural. *Classificação de grãos - soja*. 2023. URL: https://www.sistemafaep.org.br/wp-content/uploads/2024/01/PR.0230-Classificacao-Soja_web.pdf (visited on 06/25/2025).

[15] *Webcam 2K WC577*. 2025. URL: https://www.bright.com.br/produtos/informatica/webcam/371-webcam-2k (visited on 06/25/2025).

[16] *YOLO*. 2025. URL: https://docs.ultralytics.com/ (visited on 06/25/2025).