GLASGOW CALEDONIAN UNIVERSITY

MEng Group Research Project

MMH723842-24-AB-GLAS

Design and implementation of a PSD-Based Analogue 2D Sun Sensor

word count: xxx

by Zac McCaffery, Alexandru Belea,
Sebastian Alexander, William Kong, Nassor Salim,

Date: April 9, 2025

Contents

Abstract **5**

1 Acknowledgements **6**

2 Introduction **7**
2.1 Problem Statement 7
2.2 Aim of the Project 7
2.3 Objectives of the Project 7

3 LiteratureReview **10**
3.1 CubeSat Design 10
3.2 PSD Enabled Sun Sensor 11
3.3 Mechanical Design and Analysis 11
3.4 Photodiode Simulation and Signal Analysis 11
3.5 IoT Communication Enhancement with LEO Satellites 11

4 Background **12**

5 Methodology **13**
5.1 Python Simulation tbc 13
5.2 Prototype Development 13
5.2.1 Lifecycle 13
5.3 System Design Overview 17
5.3.1 Functional Requirements 18
5.3.2 Design Approach 18
5.3.3 System Architecture 18
5.4 Sensor Array Development 18
5.4.1 Functional Requirements 20
5.4.2 Design Approach 20
5.4.3 System Architecture 20
5.5 Signal Conditioning Circuitry 20
5.5.1 Functional Requirements 21

5.5.2 Design Approach . . . 21
5.5.3 System Architecture . . . 21
5.6 Enclosure Design And Fabrication . . . 21
5.6.1 Functional Requirements . . . 23
5.6.2 Design Approach . . . 23
5.6.3 System Architecture . . . 23
5.6.4 Arduino based DAQ . . . 23
5.6.5 Renewable Energy Demonstrator Testbench . . . 25

6 Results **26**
6.1 Sensor Characterization . . . 26
6.1.1 Functional Requirements . . . 27
6.1.2 Design Approach . . . 27
6.1.3 System Architecture . . . 27
6.2 Amplification Performance . . . 27
6.2.1 Functional Requirements . . . 29
6.2.2 Design Approach . . . 29
6.2.3 System Architecture . . . 29
6.3 Photodiode Angular Response . . . 29
6.3.1 Functional Requirements . . . 30
6.3.2 Design Approach . . . 30
6.3.3 System Architecture . . . 30
6.4 Enclosure Effectiveness . . . 30
6.4.1 Functional Requirements . . . 32
6.4.2 Design Approach . . . 32
6.4.3 System Architecture . . . 32
6.5 Data Acquisition System Evaluation . . . 32
6.5.1 Functional Requirements . . . 33
6.5.2 Design Approach . . . 33
6.5.3 System Architecture . . . 33
6.6 System Performance Analysis . . . 34
6.6.1 Operational Constraints Identified . . . 34
6.6.2 Environmental Factors Impact . . . 34
6.6.3 System Stability and Repeatability . . . 34
6.6.4 Recommendations for Improvement . . . 34
6.7 Comparative Analysis . . . 34
6.7.1 Breadboard vs. Stepboard Results . . . 35
6.7.2 Iteration Improvements Analysis . . . 35
6.7.3 Performance Against Design Requirements . . . 35

6.7.4 Design Evolution Assessment 35
6.8 System Limitations And Considerations 35
6.8.1 Functional Requirements 36
6.8.2 Design Approach 36
6.8.3 System Architecture 36

7 Conclusions **39**

8 FutureWork **40**

Bibliography **40**

List of Figures

5.1 System Design Overview Flowchart 18
5.2 System Architecture Diagram 19
5.3 System Design Overview Flowchart 19
5.4 System Architecture Diagram 20
5.5 System Design Overview Flowchart 21
5.6 System Architecture Diagram 22
5.7 System Design Overview Flowchart 22
5.8 System Architecture Diagram 23
5.9 Flowchart of C code on Arduino DAQ 24

6.1 System Design Overview Flowchart 27
6.2 System Architecture Diagram 28
6.3 System Design Overview Flowchart 28
6.4 System Architecture Diagram 29
6.5 System Design Overview Flowchart 30
6.6 System Architecture Diagram 31
6.7 System Design Overview Flowchart 31
6.8 System Architecture Diagram 32
6.9 System Design Overview Flowchart 33
6.10 System Architecture Diagram 34
6.11 Environmental Testing Results 34
6.14 System Design Overview Flowchart 35
6.12 Overall System Performance Analysis 37
6.13 Prototype Iteration Comparison 38
6.15 System Architecture Diagram 38

Abstract

add abstract here

1. Acknowledgements

2. Introduction

2.1 Problem Statement

With the ever-increasing commercialization of the space and satellite industry there is a growing need for a cost-effective method of attitude tracking for smaller satellite missions of such as CubeSat as these missions are purpose built for very specific objectives. Whilst the larger commercial satellite missions make use of expensive digital camera systems for tracking purposes, this is not feasible for much smaller CubeSat setups. CubeSats are defined from 1 unit to 12 – where 1U is a 10x10x10 cm satellite. Consequently, there is a demand for a cost-effective and easily implementable attitude tracking system that can provide accurate measurements for CubeSat missions, such as a Position Sensitive Detector (PSD) using photodiodes.

2.2 Aim of the Project

"To investigate and develop a cost-effective and reliable sun sensing solution suitable for Low Earth Orbit (LEO) nanosatellite attitude determination."

2.3 Objectives of the Project

To investigate the design of a sun sensing system for nanosatellites, used in orientation determination, through detection of its relative position to the sun using analogue sensors located on the satellite's body. Our goal is to create a system which balances cost-effectiveness and simplicity. To achieve this, we will create a software model of the analogue sensor(s) to simulate the system's ability to track the sun from various angles in orbit. After which, we aim to build a physical prototype and use a movable light source to simulate the sun's movement, allowing comparison between the real sensor's performance against our simulations. Although the physical prototype will be built using non-space-grade materials, one of the objectives is to look at and analyse materials required for building a space-grade PCB and sensor. For this step, the Mechanical side of the team will perform Printed circuit board (PCB) and aperture device finite analysis

using ANSYS to determine resilience to environmental factors such as stress and thermal simulation. Throughout the project, we will address challenges like interference from other light sources (such as the moon), reflections from the Earth's surface, and how factors like radiation and temperature changes in space might impact the sensor's accuracy. The application of signal processing will be explored to provide usable data, filter out noise, and improve the system's accuracy. This approach aims to develop a cost-effective and reliable, in-house sun sensing solution specifically for nanosatellites operating in Low Earth Orbit. Major Objective points:

- **Conduct literature review:**
  - Analyse existing research on sun sensing technologies, with a focus on PSD-based analogue sensors and their applications in nanosatellites.
  - Identify current challenges, best practices, and advancements in attitude determination in Low Earth Orbit. Use these insights to guide the design and optimisation of the proposed sun sensing system.
- **Develop software model:**
  - Simulate the performance of the PSD-based analogue sun sensor in tracking the sun's position from various angles in Low Earth Orbit.
- **Design and fabrication of physical prototype:**
  - Integrate analogue sun sensor components, test and validate its performance under controlled conditions.
- **Compare simulated and experimental results:**
  - Establish evaluation methodology between simulated and experimental test results to ensure that topology evaluation is applicable.
- **Optimise sensor topology:**
  - Research and evaluate various configurations of analogue sun sensing systems to maximise sun detection accuracy and minimise blind spots.
- **Investigate environmental factors:**
  - Evaluate the impact of relevant LEO specific environmental factors on the sensor's accuracy and reliability.
  - Evaluate the material requirements of the PCB and aperture device.
- **Implement signal processing algorithms:**

  - Investigate the filtering of noise to enhance the signal-to-noise ratio and otherwise ensure the acquisition of usable data for accurate sun position determination.
  - Implement data handling which optimises scanning rates and efficiently processes the analogue signal data for real-time attitude determination.

- **Document results and overall cost-effectiveness:**
  - Develop criteria for final evaluation of sun sensing systems, on which to base the final presentation of project findings.

3. LiteratureReview

3.1 CubeSat Design

Puig-Suari, Turner and Ahlgren published an IEEE paper in 2001 with the help of their students at California Polytechnic State University exploring a need for micro satellites for use by universities in an ever-expanding space programme. They provide as a solution a standard satellite form-factor that will bring down the cost of both manufacture and deployment of satellites by smaller entities: the CubeSat. The paper identifies a key component for the success of this form factor a need for a standard CubeSat deployer mechanism which can deploy several satellites safely and develop such a platform, called Poly Picosatellite Orbital Deployer or P-POD. They point out the need and provide microsatellite size and shape of the CubeSat form factor [1]. Sai balaji et al. performed a study using MATLAB simulation of several attitude control algorithms to look at the ability to control a CubeSat of size 1U. They also simulated sensors such as sun sensors, magnetometer, and gyroscope. They concluded that it is possible to operate the satellite using a magnetorquer type actuator and an array of mathematical models and algorithms: it would take 2000 seconds for a 1U satellite to stabilize at 505km, 98° degree attitude in orbit with the methods utilized by them [**?**]. Incentivised by the rapidly increasing use of LEO, Lopez-Calle and Franco perform a quantitative comparative study on the catastrophic failure of CubeSats and Nanosats from radiation exposure due to the harsh environment of space versus failure due to collisions in the increasingly busy Low Earth Orbit (LEO). The authors concluded that while sustained damage and damage protection from radiation exposure used to be and currently still is the most crucial factor in protecting LEO microsatellites, increasingly the risk of debris collisions is becoming more important and will become the most important in the following 50 to 70 years. The authors conclude that microsatellite designers need to move their focus more towards defence from debris impacts as these, even if not resulting in catastrophic failure of the satellite, they will impact the attitude of the satellite [**?**].

3.2 PSD Enabled Sun Sensor

3.3 Mechanical Design and Analysis

3.4 Photodiode Simulation and Signal Analysis

3.5 IoT Communication Enhancement with LEO Satellites

4. Background

5. Methodology

5.1 Python Simulation tbc

5.2 Prototype Development

5.2.1 Lifecycle

This section provides an overview of the Prototype Development Lifecycle.

Conceptualization and Requirements Definition

- The prototype must have four photodiodes in an xy pattern with respective circuitry required to output 0-5 Volts that will be read by an Arduino based DataAcquisitionSystem (DAQ). The circuit must be able to react to light intensity changes, however the change will be at low frequency (below 1Hz) as a satellite attitude is considered to change only gradually.
- While the prototype may not have a high accuracy, it is hoped that it will be enough to measure light position changes.
- The prototype within the scope of this paper will show the ability to detect the position of light at normal room conditions, therefore it does not need to withstand temperature changes or radiation that a final product would require if deployed in space.
- Interface requirements: the prototype electrical output needs to be compatible with the Arduino Analog to Digital Converter (ADC) input. Therefore, the signal shall not go below 0 Volts or exeed 5 volts.
- Size and weight are not of high importance, but the device must fit in the testing equipment, which is the Renewable Energy Demonstrator arch. Preferably a height not higher than 5cm.

Theoretical Design

- Research photodiode technology options and selection criteria
- Model sun sensor geometry and aperture design
- Determine optimal photodiode placement for coverage and accuracy
- Develop mathematical models for sun vector determination
- Simulate sensor performance under various lighting conditions

Preliminary Design

- Create detailed electrical schematics
- Design photodiode array configuration
- Engineer aperture design and mask pattern
- Develop analog front-end circuitry
- Design signal conditioning and processing circuits
- Outline firmware architecture and algorithms

Component Selection and Procurement

- Select appropriate photodiodes (spectral response, sensitivity)
- Choose microcontroller/processor
- Source analog-to-digital converters
- Identify appropriate materials for aperture and housing
- Procure test equipment for validation

Breadboard Testing

- Assemble basic circuit on breadboard
- Test photodiode response characteristics
- Verify analog front-end performance
- Validate signal processing approach
- Identify design weaknesses and optimization opportunities

First Prototype Development

- Design printed circuit board (PCB)
- Manufacture PCB
- Design and fabricate aperture mask
- Develop housing/enclosure
- Assemble prototype components
- Write initial firmware implementation

Initial Testing and Characterization

- Conduct functional testing
- Measure photodiode response curves
- Characterize sun angle determination accuracy
- Test temperature sensitivity
- Evaluate power consumption
- Assess signal-to-noise ratio

Design Refinement

- Analyze test results
- Modify aperture design if needed
- Optimize photodiode configuration
- Update signal processing algorithms
- Refine PCB layout
- Improve firmware algorithms

Second Prototype Development

- Implement design improvements
- Manufacture revised PCB
- Fabricate improved aperture
- Enhance housing design
- Update firmware with optimized algorithms
- Assemble refined prototype

Comprehensive Testing

- Laboratory performance testing (angular accuracy, resolution)
- Environmental testing (thermal cycling, vibration)
- Radiation testing (if applicable for space applications)
- Interface compatibility testing
- Long-term stability assessment

Validation and Calibration

- Develop calibration procedures
- Create calibration fixtures
- Perform sensor calibration
- Document calibration coefficients
- Validate sensor performance against requirements

Documentation and Production Readiness

- Create detailed technical specifications
- Document calibration procedures
- Prepare assembly instructions
- Write user manual/interface control document
- Develop acceptance test procedures

Pre-production Prototype

- Build small batch of pre-production units
- Conduct acceptance testing
- Verify production processes
- Validate consistency between units
- Finalize design for production

Technology Transfer to Production

- Document manufacturing processes
- Train production personnel
- Establish quality control procedures
- Define production testing requirements
- Prepare for volume manufacturing

5.3 System Design Overview

This section provides an overview of the System Design Overview.

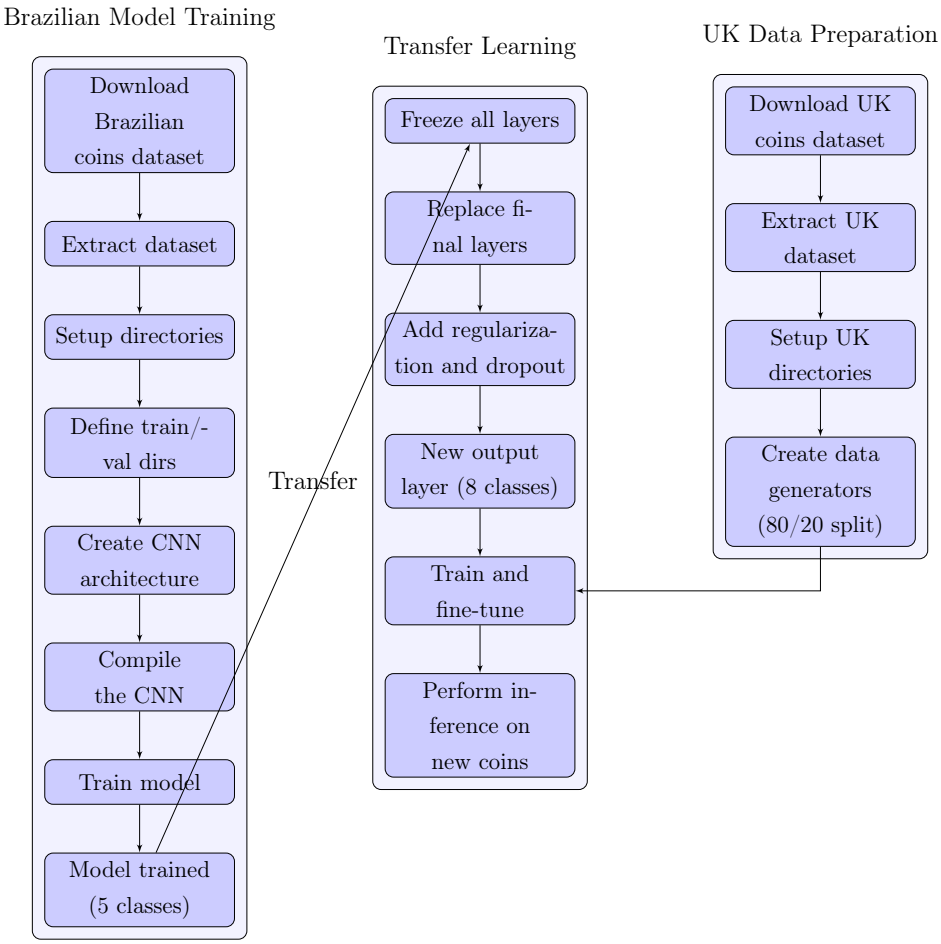

Figure 5.1: System Design Overview Flowchart

5.3.1 Functional Requirements

5.3.2 Design Approach

5.3.3 System Architecture

As shown in Figure 5.1 the system architecture consists of various components.

```
# Your code here
```

Listing 5.1: System Architecture Code Example

5.4 Sensor Array Development

This section provides an overview of the Sensor Array Development.

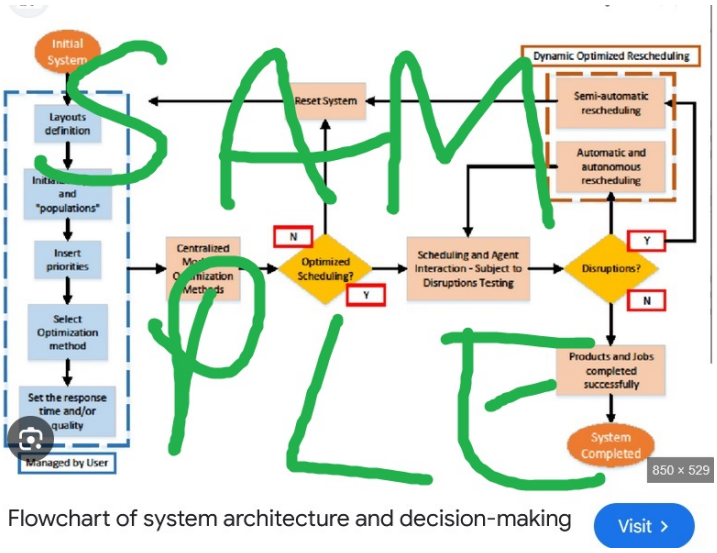

Figure 5.2: System Architecture Diagram

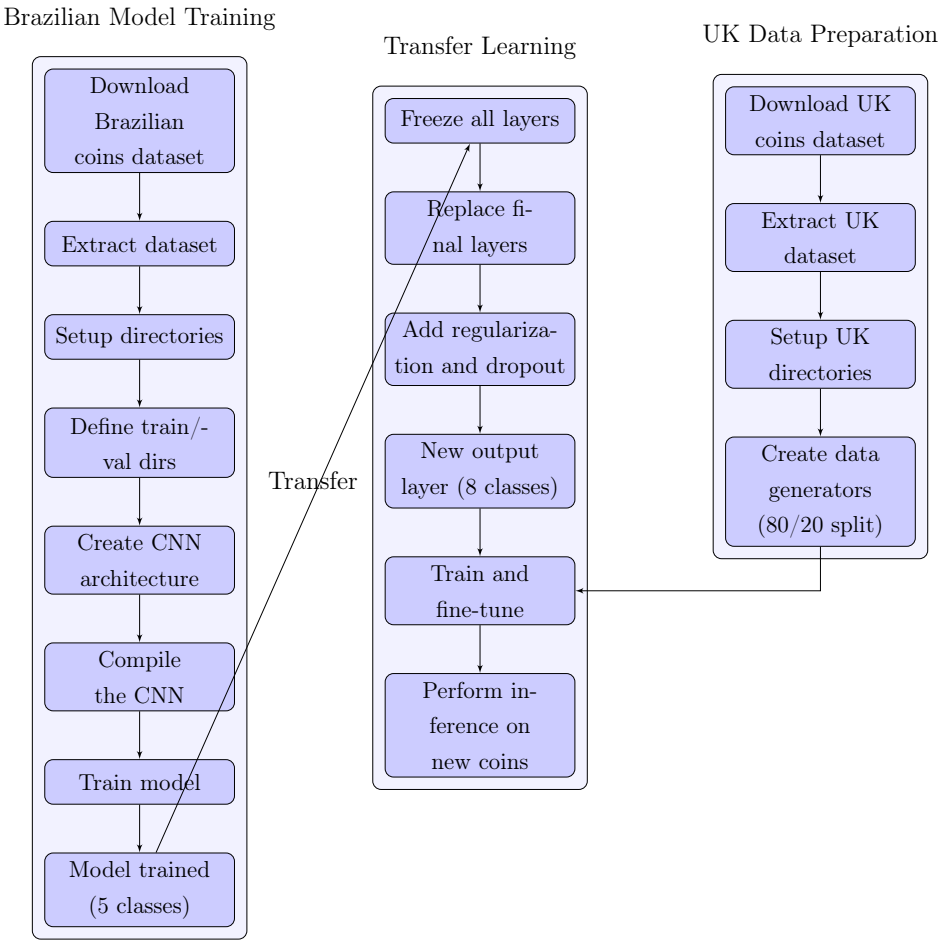

Figure 5.3: System Design Overview Flowchart

5.4.1 Functional Requirements

5.4.2 Design Approach

5.4.3 System Architecture

As shown in Figure 5.3 the system architecture consists of various components.

```
# Your code here
```

Listing 5.2: System Architecture Code Example

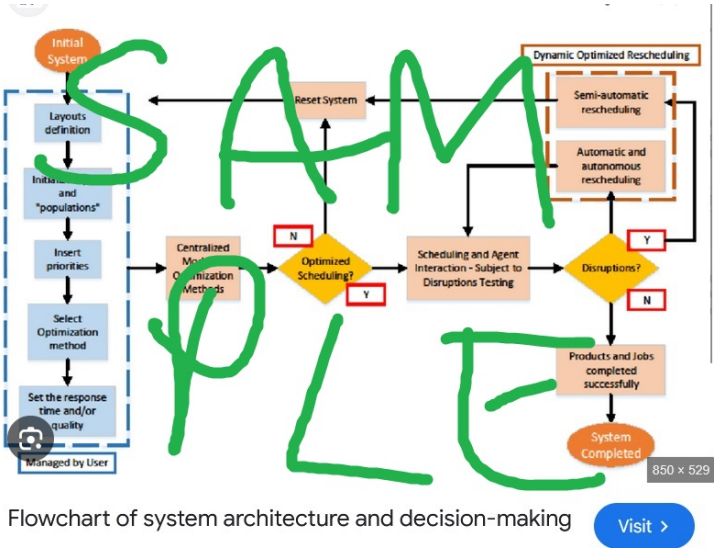

Figure 5.4: System Architecture Diagram

5.5 Signal Conditioning Circuitry

This section provides an overview of the Signal Conditioning Circuitry.

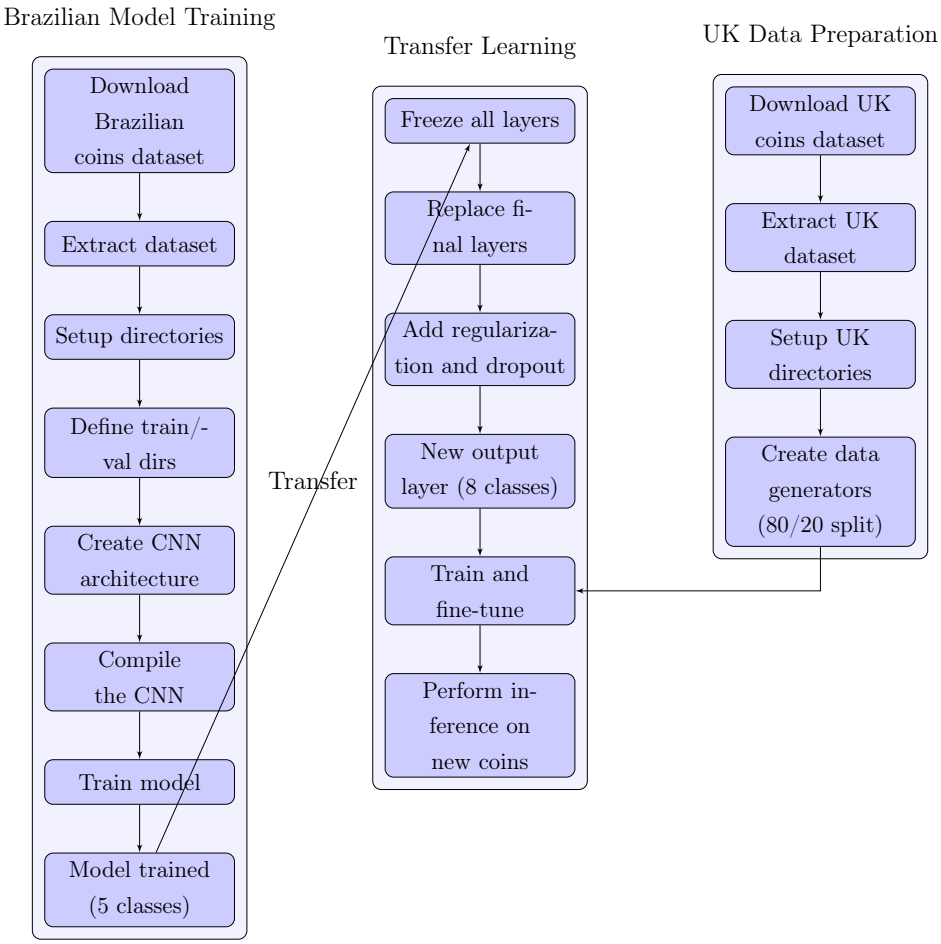

Figure 5.5: System Design Overview Flowchart

5.5.1 Functional Requirements

5.5.2 Design Approach

5.5.3 System Architecture

As shown in Figure 5.5 the system architecture consists of various components.

```
# Your code here
```

Listing 5.3: System Architecture Code Example

5.6 Enclosure Design And Fabrication

This section provides an overview of the Enclosure Design And Fabrication.

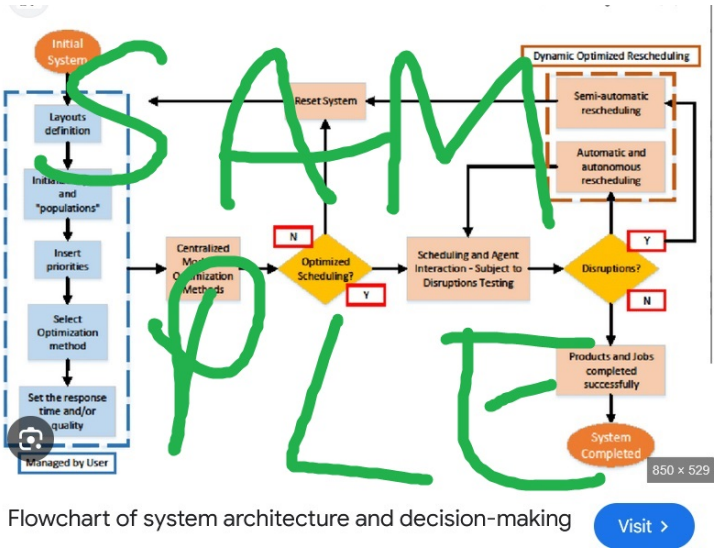

Figure 5.6: System Architecture Diagram

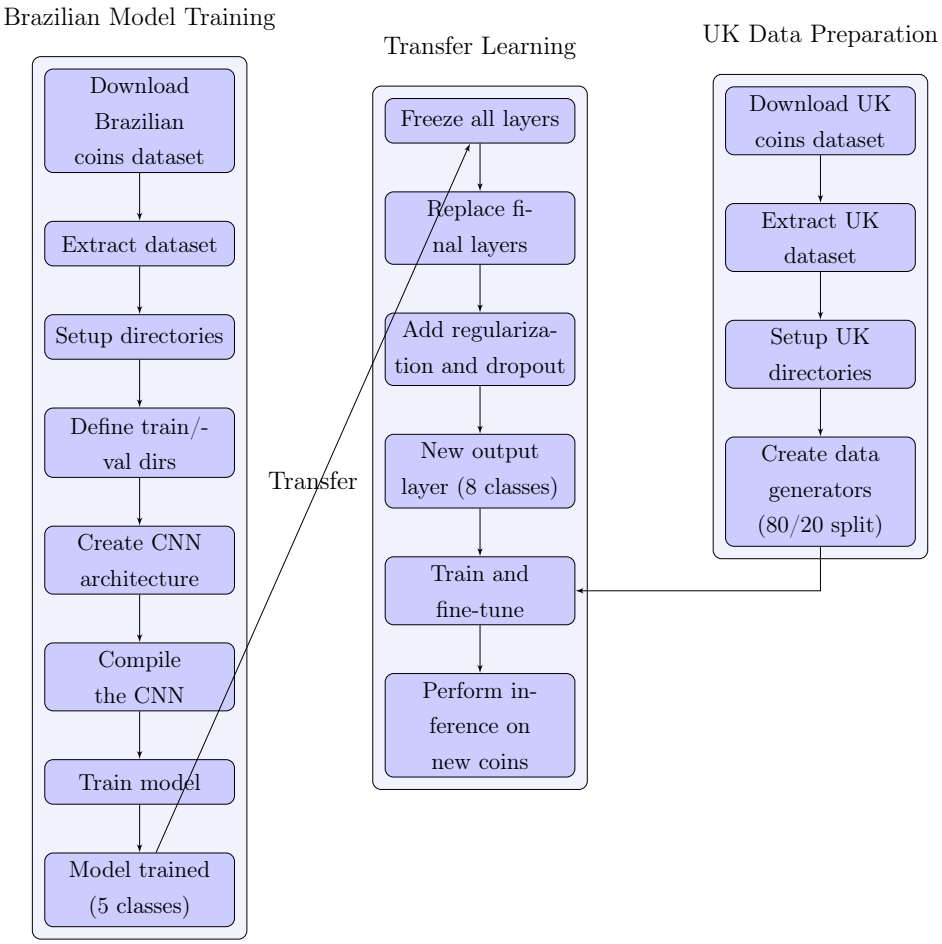

Figure 5.7: System Design Overview Flowchart

5.6.1 Functional Requirements

5.6.2 Design Approach

5.6.3 System Architecture

As shown in Figure 5.7 the system architecture consists of various components.

```
# Your code here
```

Listing 5.4: System Architecture Code Example

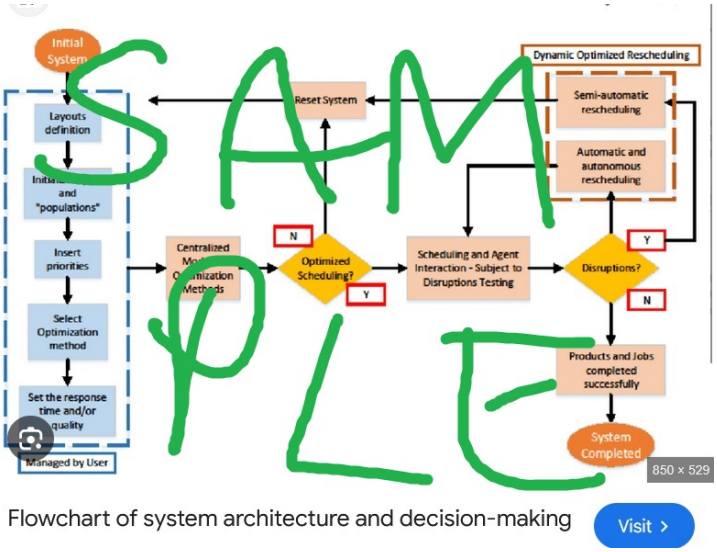

Figure 5.8: System Architecture Diagram

5.6.4 Arduino based DAQ

Arduino Code

Introduce Code

Sampling Rate Several factors restrict the sampling rate:

Sample Interval Setting The most direct limitation is the `sampleInterval` constant set to 2ms in the code, which means samples are taken no more frequently than every 2 milliseconds (500 Hz theoretical maximum).

ADC Prescaler Configuration The ADC prescaler is set to 16 (from the default of 128) with this line:

```
ADCSRA = (ADCSRA & 0xF8) | 0x04;
```

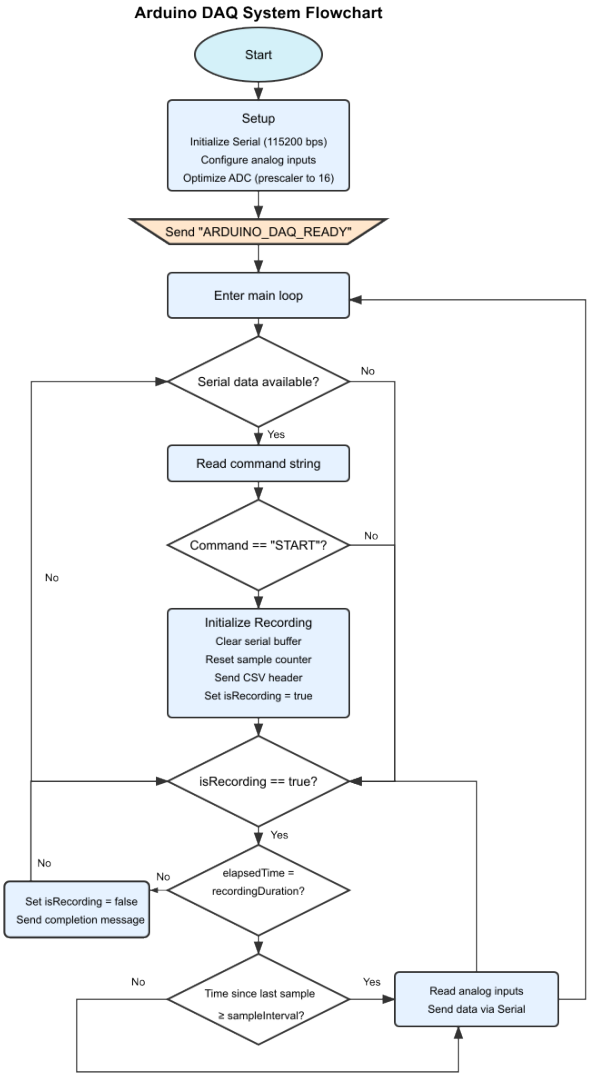

Figure 5.9: Flowchart of C code on Arduino DAQ

This increases the ADC clock to 16MHz/16 = 1MHz. With each conversion taking 13 ADC clock cycles, the theoretical maximum sampling rate is about 76.9kHz for a single channel.

Multiple Channel Reading Since the system reads from 4 analog inputs sequentially, the effective per-channel sampling rate is reduced by approximately a factor of 4.

Serial Transmission Overhead Each sample requires formatting and sending data over serial:

```
String dataString = String(sampleCount) + "," + String(elapsedTime);
// ... format and add voltage values ...
Serial.println(dataString);
```

This string creation and serial transmission takes considerable time.

Serial Baud Rate The code uses 115200 bps, which limits how quickly data can be transmitted. Each sample in this format might be around 30-40 bytes, which means ~3000-3800 samples/second theoretical maximum throughput.

String Operations The use of the Arduino `String` class is memory-intensive and can cause fragmentation over time, potentially causing slowdowns.

Loop Cycle Time Other operations in the main loop consume processing time.

The dominant limiting factor in this implementation is likely the combination of the explicit 2ms sample interval and the serial communication overhead. Higher sampling rates would require optimizing the data transmission format, possibly using binary rather than text formatting, and reducing the sample interval.

5.6.5 Renewable Energy Demonstrator Testbench

6. Results

6.1 Sensor Characterization

For the SensorCharacterization.tex file, you'd want to focus on the fundamental properties and performance of your photodiodes themselves, distinct from the other subsections. Here are some key elements that would belong specifically under SensorCharacterization:

Basic Photodiode Electrical Characteristics:

Dark current measurements Junction capacitance I-V characteristics in different lighting conditions Spectral response profiles (sensitivity vs. wavelength)

Individual Sensor Benchmarking:

Performance comparison between the 4 photodiodes (matching/differences) Responsivity measurements (A/W) Quantum efficiency calculations Detection threshold levels

Response Linearity:

Measurements showing linear range of the photodiodes Saturation point characterization Recovery time from saturation

Temperature Dependency:

Performance drift with temperature Baseline shift measurements Temperature compensation data

Aging/Stability Tests:

Long-term drift measurements Repeatability of measurements over time

This section should focus on the inherent properties of the photodiodes themselves - essentially providing the baseline characterization data that underpins all the other analysis. The other sections then build on this foundation by examining how these sensors perform when integrated into the complete system with amplification, angular positioning, enclosure effects, etc.

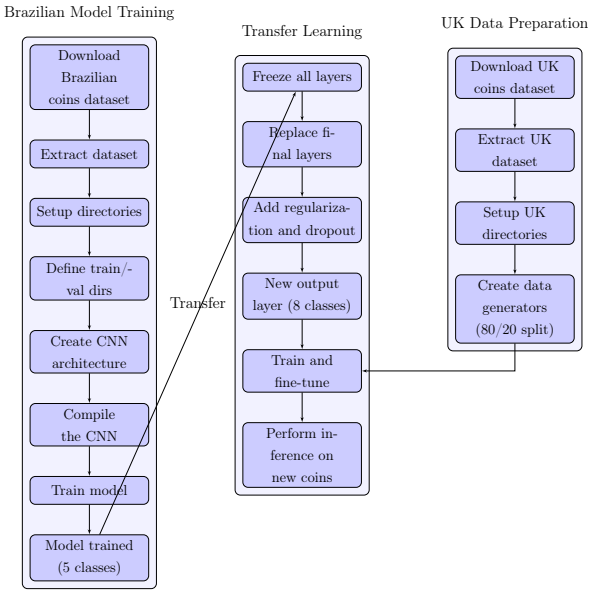

Figure 6.1: System Design Overview Flowchart

6.1.1 Functional Requirements

6.1.2 Design Approach

6.1.3 System Architecture

As shown in Figure 6.1 the system architecture consists of various components.

```
1 # Your code here
```

Listing 6.1: System Architecture Code Example

6.2 Amplification Performance

This section provides results of the amplifier performance.

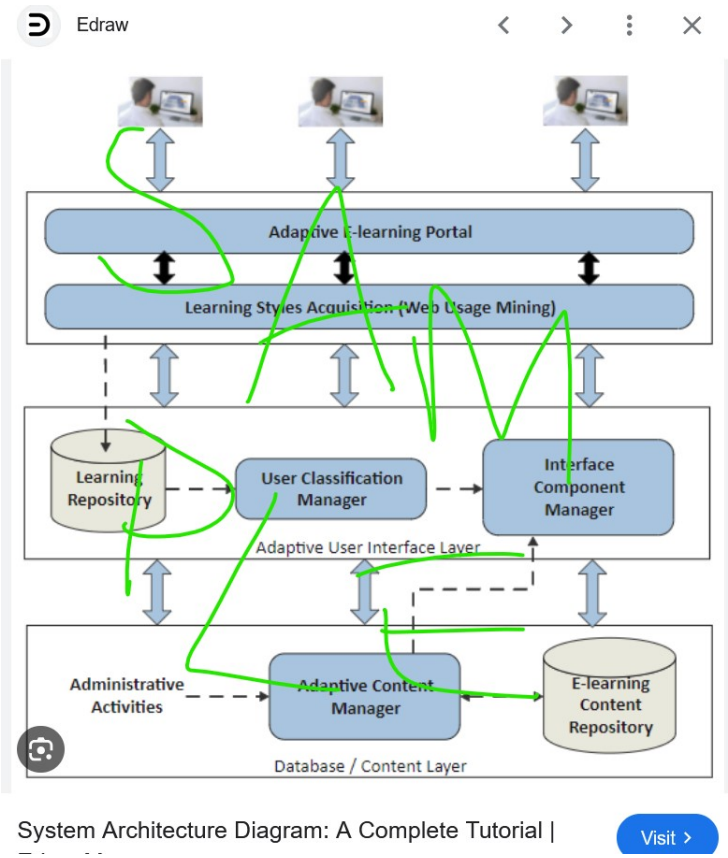

Figure 6.2: System Architecture Diagram

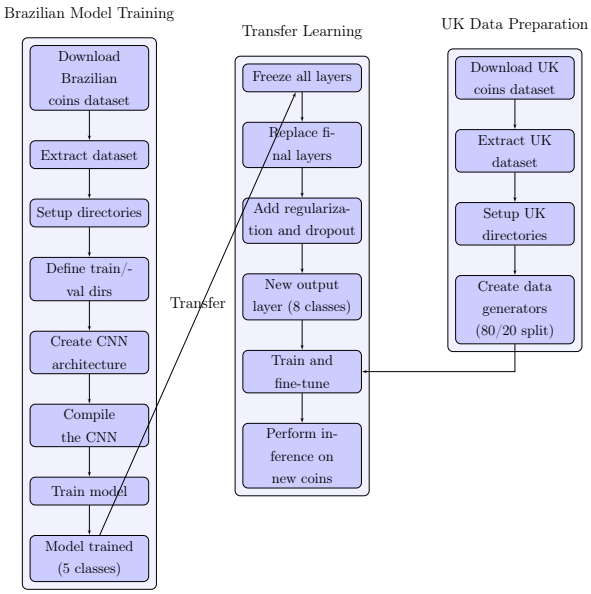

Figure 6.3: System Design Overview Flowchart

6.2.1 Functional Requirements

6.2.2 Design Approach

6.2.3 System Architecture

As shown in Figure 6.3 the system architecture consists of various components.

```
# Your code here
```

Listing 6.2: System Architecture Code Example

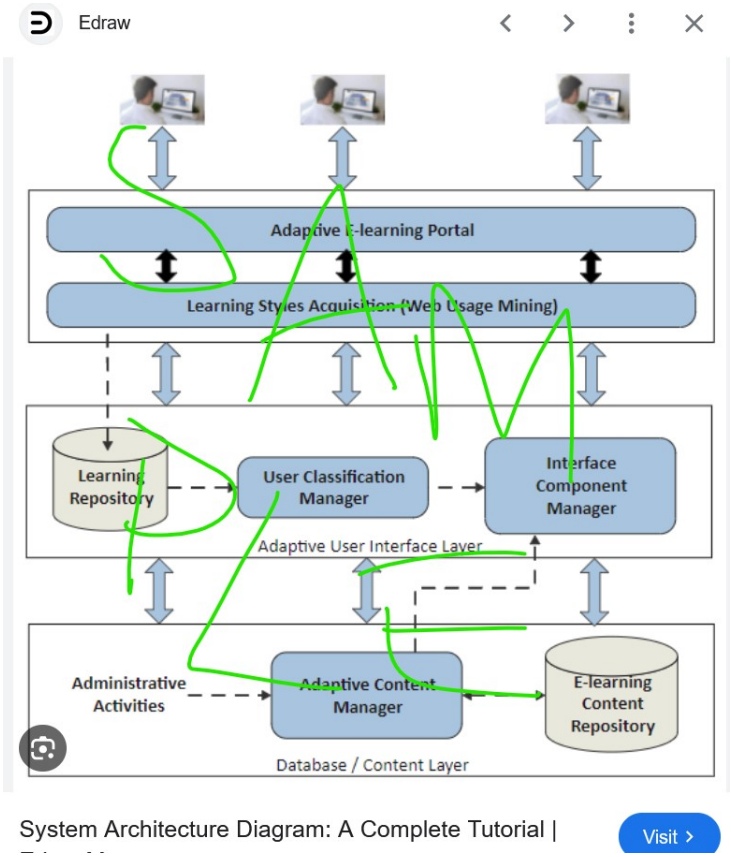

Figure 6.4: System Architecture Diagram

6.3 Photodiode Angular Response

This section discusses the results of the response of the solar sensor to angular changes of the light source.

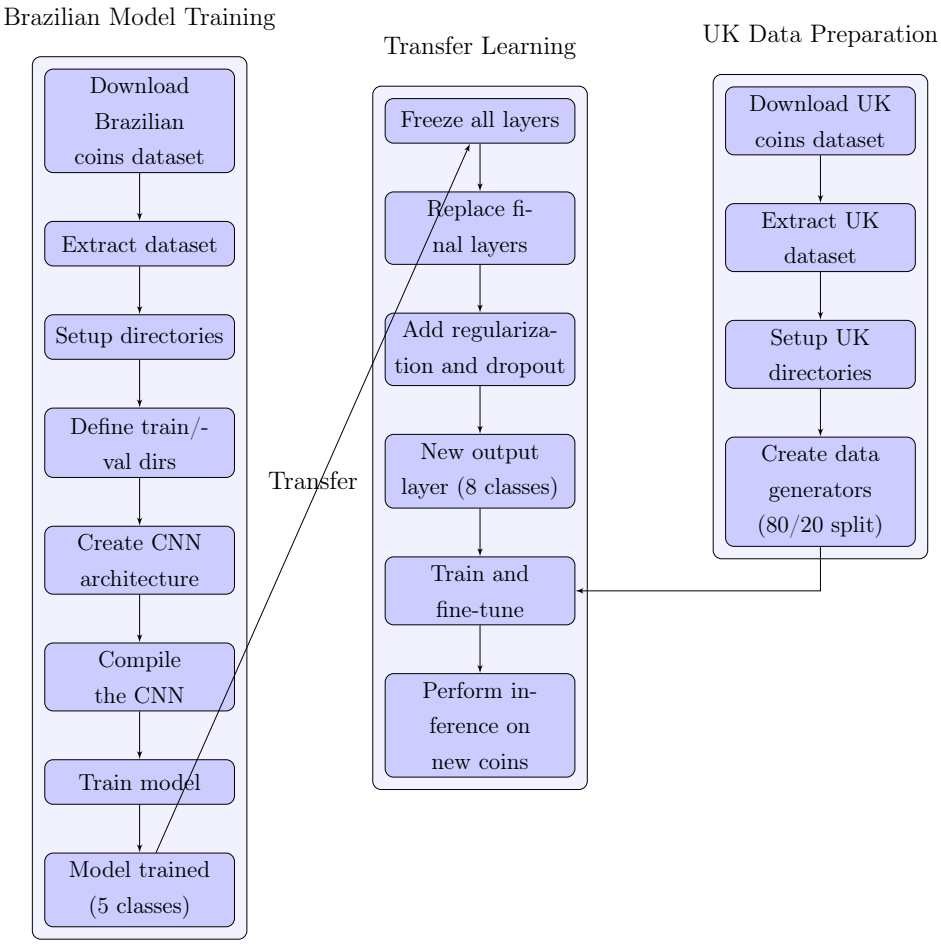

Figure 6.5: System Design Overview Flowchart

6.3.1 Functional Requirements

6.3.2 Design Approach

6.3.3 System Architecture

As shown in Figure 6.5 the system architecture consists of various components.

```
1 # Your code here
```

Listing 6.3: System Architecture Code Example

6.4 Enclosure Effectiveness

This section discusses the effectiveness of the Photodiode enlosure.

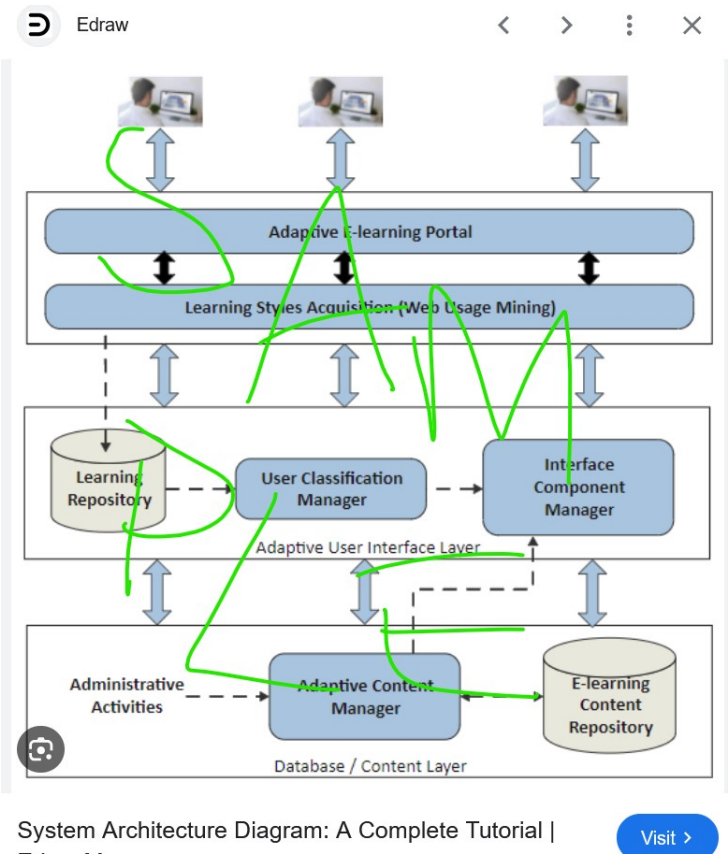

Figure 6.6: System Architecture Diagram

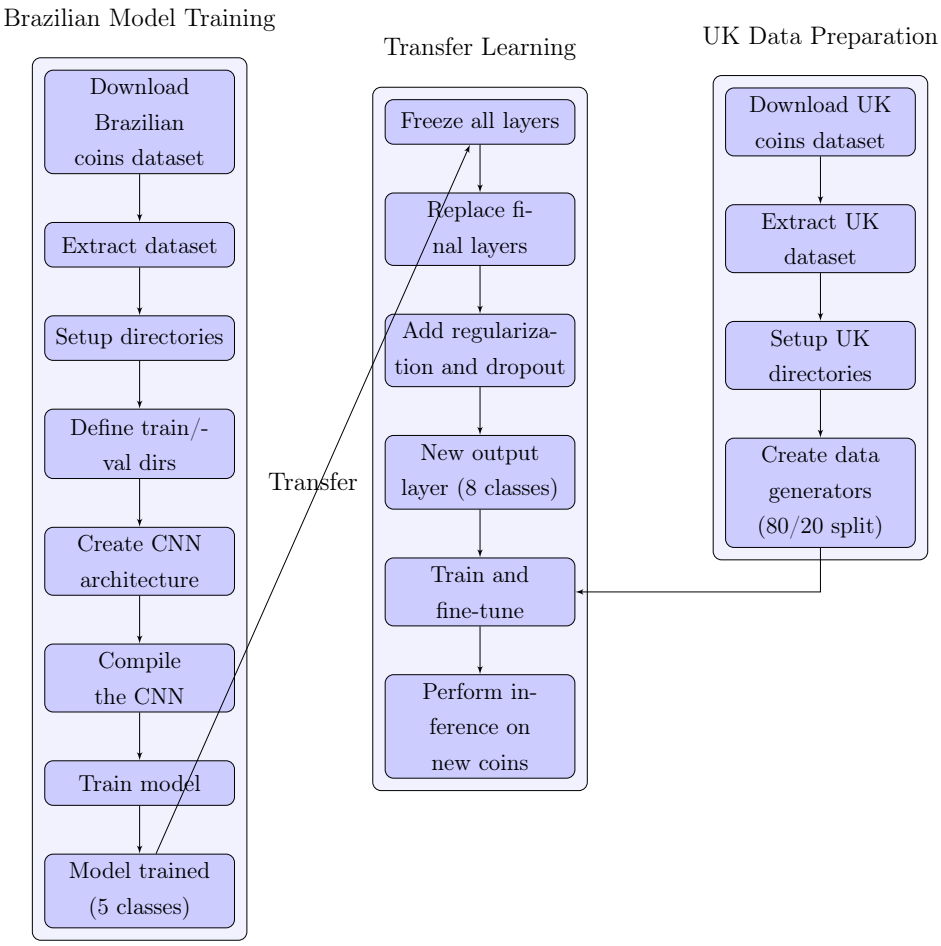

Figure 6.7: System Design Overview Flowchart

6.4.1 Functional Requirements

6.4.2 Design Approach

6.4.3 System Architecture

As shown in Figure 6.7 the system architecture consists of various components.

```
# Your code here
```

Listing 6.4: System Architecture Code Example

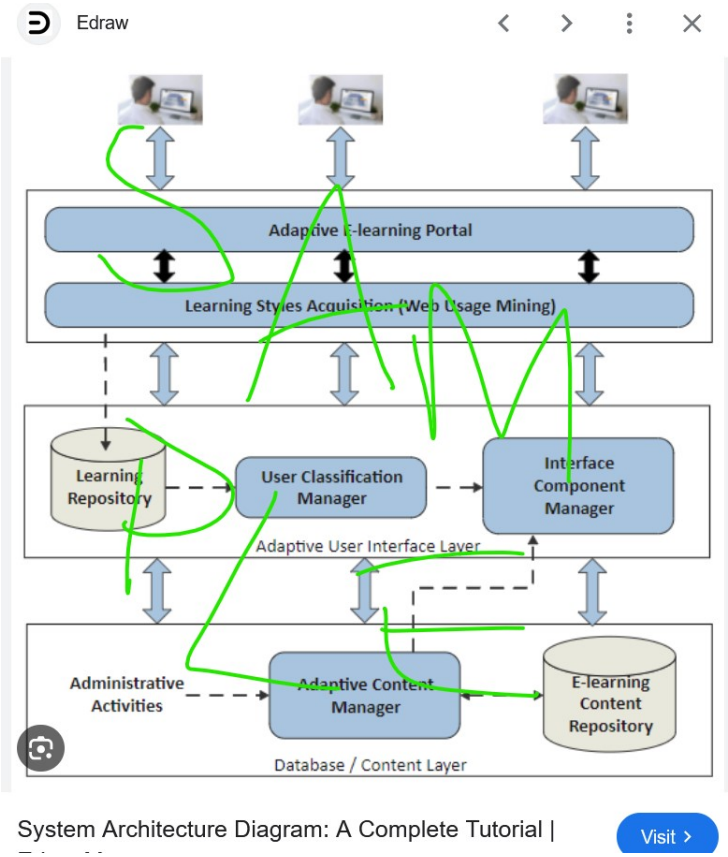

Figure 6.8: System Architecture Diagram

6.5 Data Acquisition System Evaluation

This section provides results related to the Arduino DAQ.

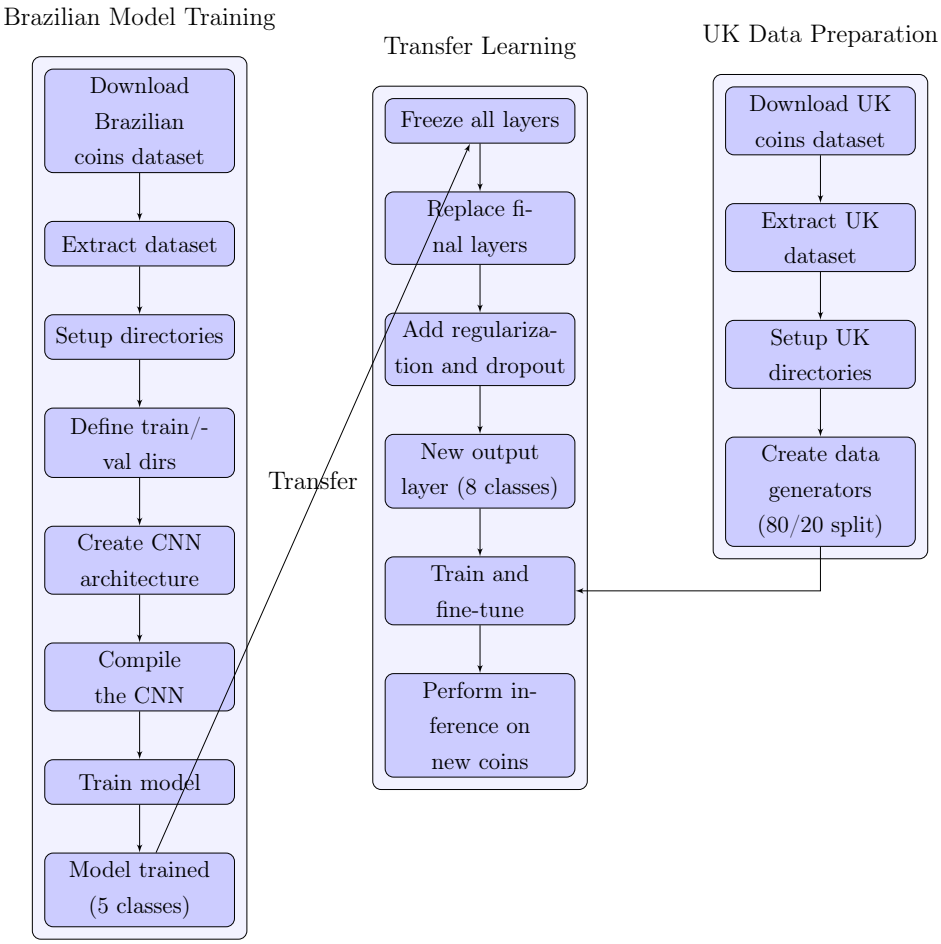

Figure 6.9: System Design Overview Flowchart

6.5.1 Functional Requirements

6.5.2 Design Approach

6.5.3 System Architecture

As shown in Figure 6.9 the system architecture consists of various components.

```
# Your code here
```

Listing 6.5: System Architecture Code Example

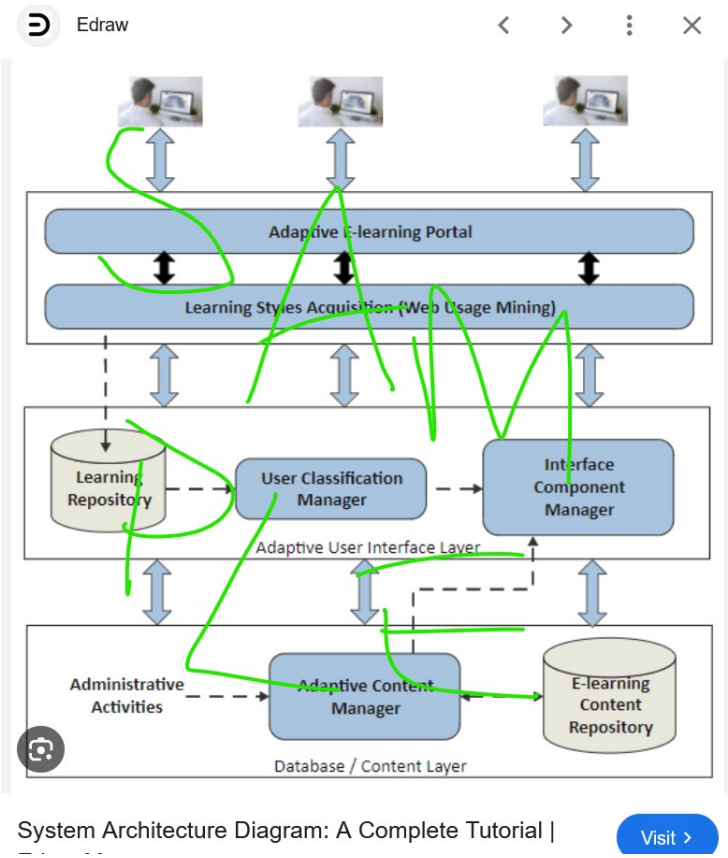

Figure 6.10: System Architecture Diagram

6.6 System Performance Analysis

6.6.1 Operational Constraints Identified

6.6.2 Environmental Factors Impact

```
// Environmental test results
// Temperature, ambient light, and vibration effects
```

Figure 6.11: Environmental Testing Results

6.6.3 System Stability and Repeatability

6.6.4 Recommendations for Improvement

6.7 Comparative Analysis

This section compares the simulation with the prototype results.

6.7.1 Breadboard vs. Stepboard Results

6.7.2 Iteration Improvements Analysis

6.7.3 Performance Against Design Requirements

The performance ...

6.7.4 Design Evolution Assessment

The what now?

6.8 System Limitations And Considerations

This section discusses the limitations and future work.

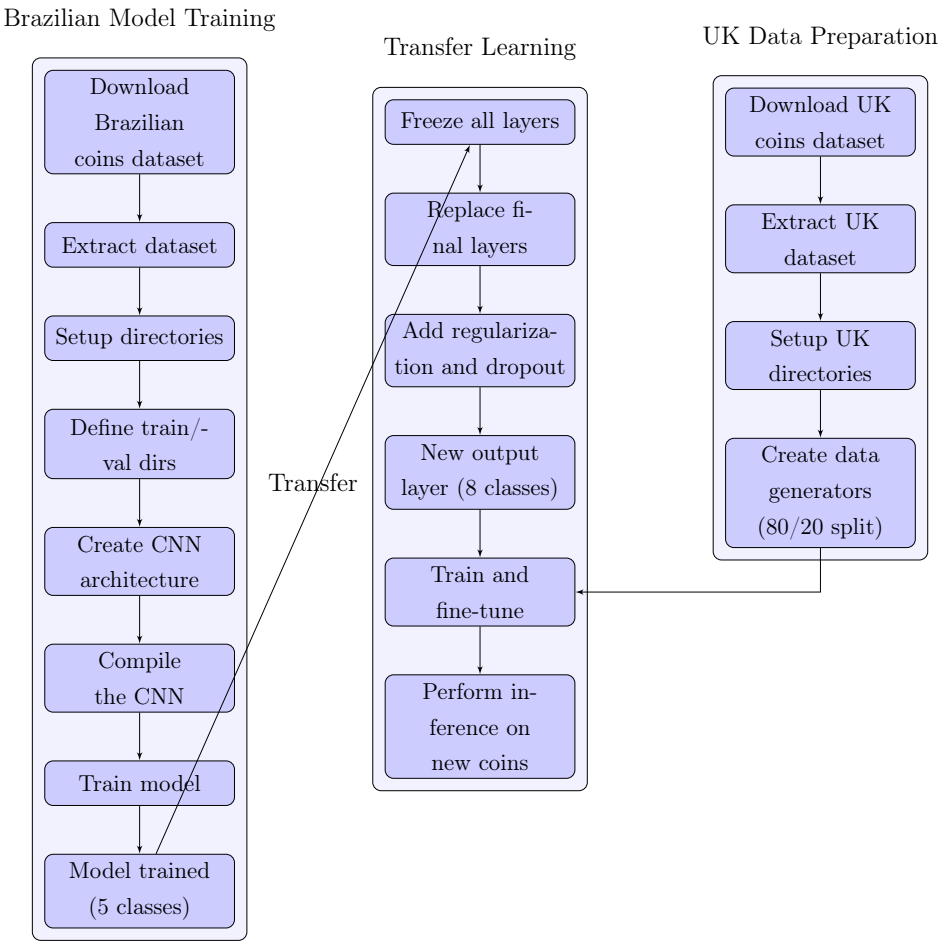

Figure 6.14: System Design Overview Flowchart

6.8.1 Functional Requirements

6.8.2 Design Approach

6.8.3 System Architecture

As shown in Figure 6.14 the system architecture consists of various components.

```
# Your code here
```

Listing 6.6: System Architecture Code Example

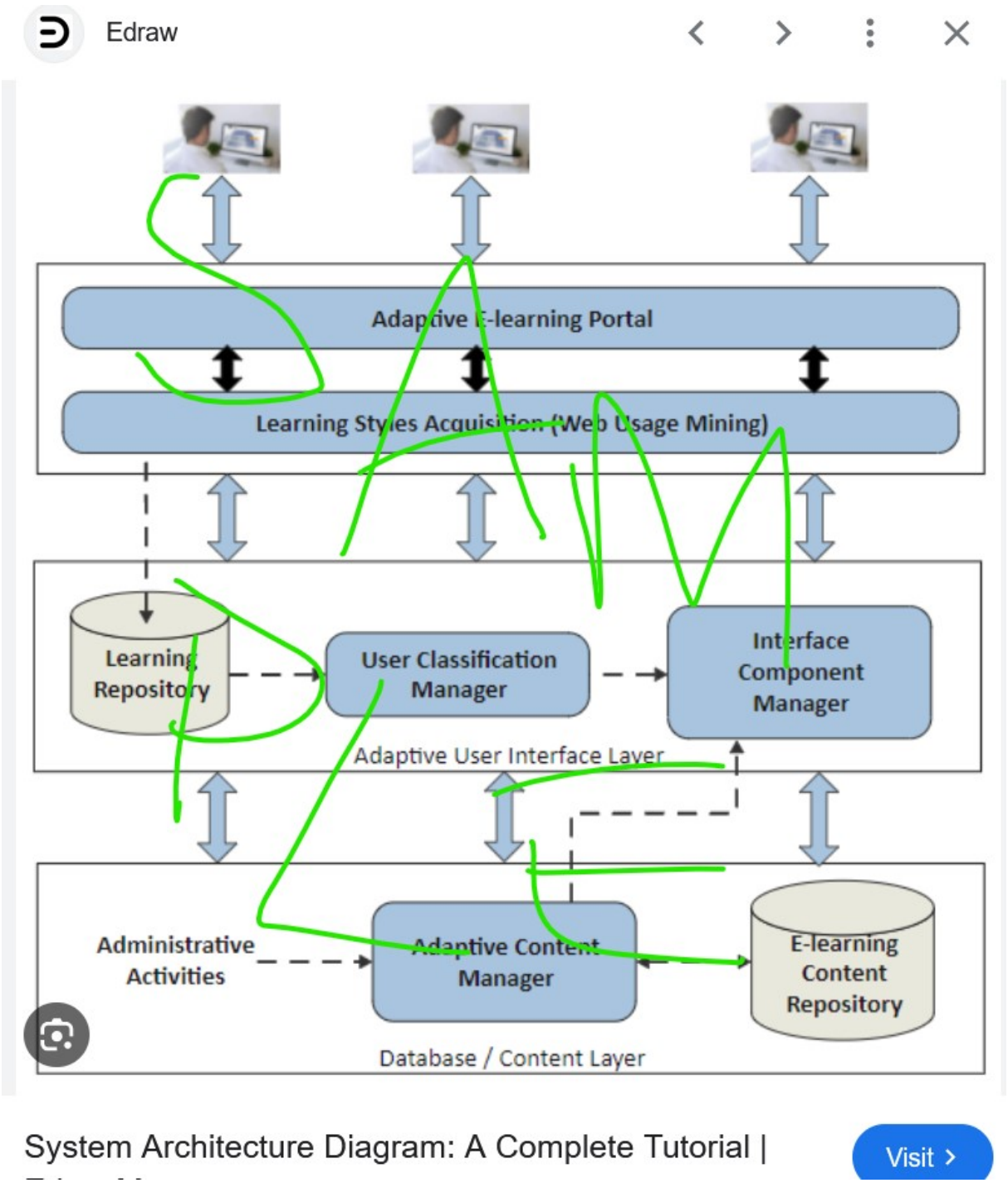

Figure 6.12: Overall System Performance Analysis

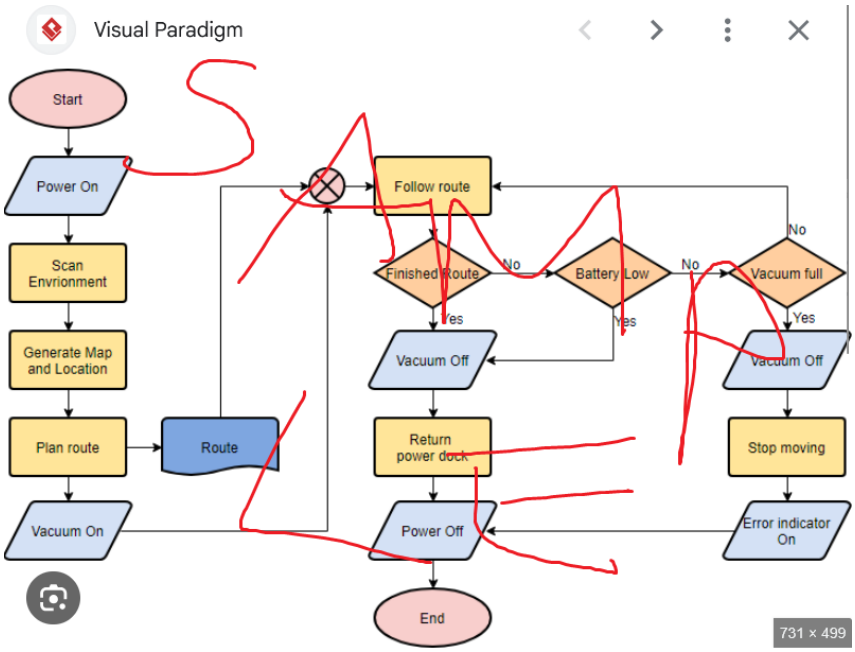

Figure 6.13: Prototype Iteration Comparison

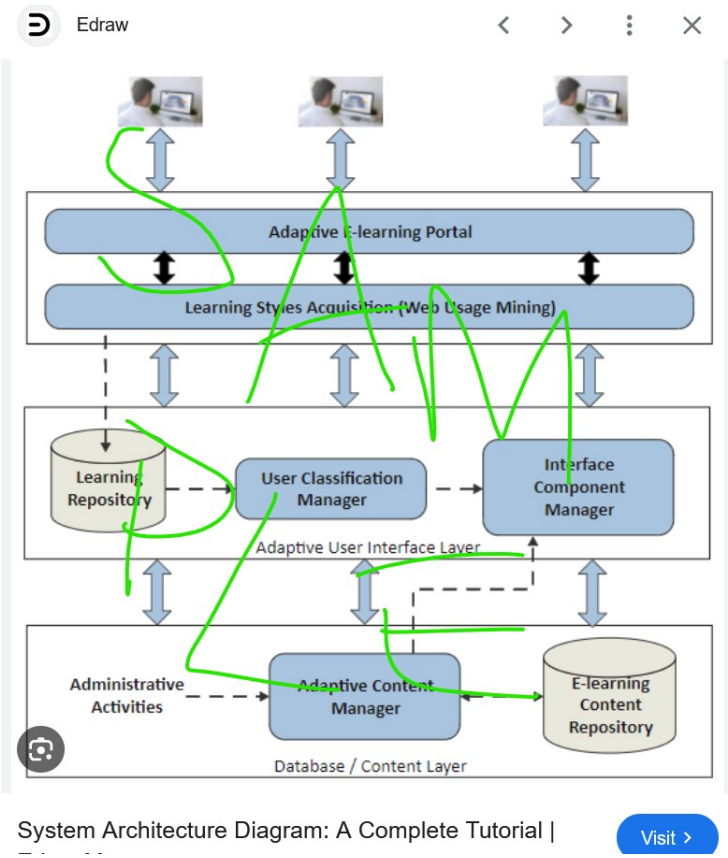

Figure 6.15: System Architecture Diagram

7. Conclusions

8. FutureWork

Bibliography

[1] J. Puig-Suari and C. Turner, "Development of the standard cubesat deployer and a cubesat class picosatellitel."