

# Interim Report

Shaheen Amin, 01866464

Dr. Edward Stott, Supervisor

December 2023

## Abstract & Plagiarism Declaration

To address the issue of electronic waste and cluttered workspaces in the Level 1 Labs, this project aims to develop an automated system for identifying and sorting electronic components. Utilising computer vision techniques, the system will classify components like resistors, capacitors, ICs, and MOSFETs, and sort them into designated bins. The project will be executed in three stages: initial development of a computer vision system for component identification, integration of a semi-automated sorting mechanism, and potentially, full automation of the sorting process or additional features like IC testing. Supervised by Dr. Edward Stott, this project aims to practically solve the problems faced by the Level 1 Labs.

I affirm that I have provided explicit references for all the material in my Interim Report that is not authored by me but is represented as my work. I have used GPTv4, as an aid in the preparation of my report. It was used for LaTeX formatting, however all technical content is original.

## Contents

<b>1 Project Specification</b> .....	
<b>2 Background</b> .....	
2.1 Existing Computer Vision Systems ..	
2.2 Real-time Computer Vision Architectures .....	
2.3 Training Methods .....	
2.4 Mechanical Design .....	
2.5 Key Takeaways .....	
<b>3 Implementation</b> .....	
3.1 Hardware, Mechanics and Electronics .....	
3.1.1 Hardware.....	5
3.1.2 Mechanics.....	6
3.1.3 Electronics .....	7
3.1.4 Problems and Solutions .....	8
3.2 Software .....	11
3.3 Computer Vision.....	12
3.3.1 Data Collection .....	12
3.3.2 Vision System Pipeline .....	13
<b>4 Project Plan</b> .....	13
4.1 Stage 1.....	14
4.2 Stage 2.....	15
4.3 Stage 3.....	15
4.4 Stage 4.....	15
<b>5 Evaluation Plan</b> .....	16
<b>6 Ethical, Legal, and Safety Plan</b> .....	16
6.1 Electrical Safety .....	16
6.2 Mechanical Safety .....	17
<b>7 References</b> .....	17
<b>8 Appendix</b> .....	20

## 1 Project Specification

- 1 The motivation for this project is three-fold;
- 2 to solve the problem of electronic waste and cluttered workspaces in the Level 1 Labs; to alleviate the time-consuming process of sorting electronic components from the Labs' technicians;
- 3 and to bring the Lab closer to meeting the requirements for the LEAF (Laboratory Efficiency Assessment Framework) certification<sup>28</sup> that the Lab is currently working towards.

The LEAF certification is a framework that aims to improve the efficiency of laboratories by reducing waste, energy consumption, and costs. This project aligns with the LEAF certification's goals by reducing the amount of electronic waste produced by the Lab.

The project aims to achieve these goals by employing state-of-the-art computer vision techniques to classify various electronic components, and then sort them into designated bins. The project's deliverables are as follows:

- A computer vision system for component identification
  - An interface from which to observe and control the system's state
  - A semi-automated sorting mechanism
  - Potentially, full automation of the sorting process or additional features like IC testing
- After an initial consultation with Dr. Stott and visiting the Level 1 Labs to discuss the project, samples of the different electrical components commonly used were received, as follows:

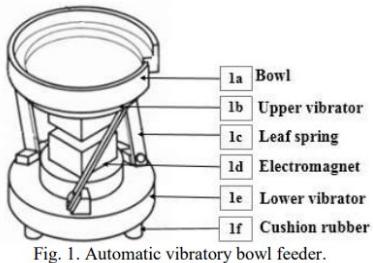


Figure 1: VBF<sup>26</sup>

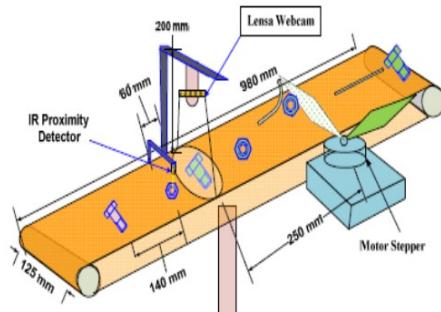


Figure 2: Conveyor Belt<sup>9</sup>

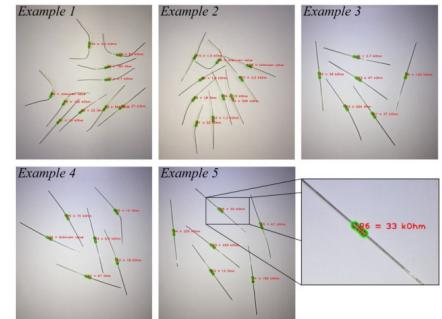


Figure 3: Resistor Test Set<sup>24</sup>

- Resistors
- MOSFETs
- Capacitors
- Transistors
- Ceramic Capacitors
- LEDs
- Inductors
- Wires
- Diodes
- ICs

For this initial stage of the project, the focus is on developing the foundation of the system that the project will be built upon so that the development of the various systems can be done independently, in parallel, and in such a way that each system is modular and easily extensible. For this reason, the project was initially split into three stages:

- Initial development of a computer vision system for component identification
- Integration of a semi-automated sorting mechanism
- Potentially, full automation of the sorting process or additional features like IC testing

After the initial stage, it was decided that the first stage would be split into two stages, with the first stage focusing on the development of the foundation of the system, and the second stage focusing on the development of the computer vision system, for a total of four stages. This was because the development of the foundation of the system took longer than expected, and the vision system needs to be trained on data that is representative of the conditions it will be used in, which requires the mechanical design to be finalised.

This approach was adopted to establish defined goals, ensuring that the project consistently progresses towards a deliverable product. Each stage extends the previous stage, so the project can be considered as a series of smaller projects.

## 2 Background

For the background of this project, it is necessary to consider literature related to the following:

- Existing computer vision systems for component identification
- Real-time computer vision architectures
- Mechanical design of existing sorting machines

Research into these topics will inform the decisions made in the design of the project and its various systems.

### 2.1 Existing Computer Vision Systems

A range of computer vision systems have been explored in the literature, ranging from PCA (Principle Component Analysis) Dhenge et al.<sup>9</sup> to more modern computer vision techniques like CNNs as used by Chand and Lal<sup>3</sup>, Xu et al.<sup>44</sup>. However, given the rate of advancement of computer vision techniques, the techniques used in the literature reflect the state of the art at the time of writing, and so are not necessarily the most viable current techniques for the project, but they inspire novel approaches to the problem of component identification.

PCA with ANN (Artificial Neural Networks) as used by Dhenge et al.<sup>9</sup> is a relatively simple but outdated statistical technique that was successful in identifying nuts and bolts, which are very distinct components. In this paper, PCA is used as a feature extractor and then fed directly into, though not explicitly mentioned, a FCL (Fully Connected Layer), as the classifier. Although a valid approach, CNNs (Convolutional Neural Networks) are known to be much more effective at feature extraction, and do not "have the problem of low recall and accuracy", as mentioned by Xu et al.<sup>44</sup>, that PCA may suffer from, so this approach is not viable for the project.

The authors Xu et al.<sup>44</sup> use the SqueezeNet CNN architecture to identify 22 different subcategories of electronic components, specifically resistors,

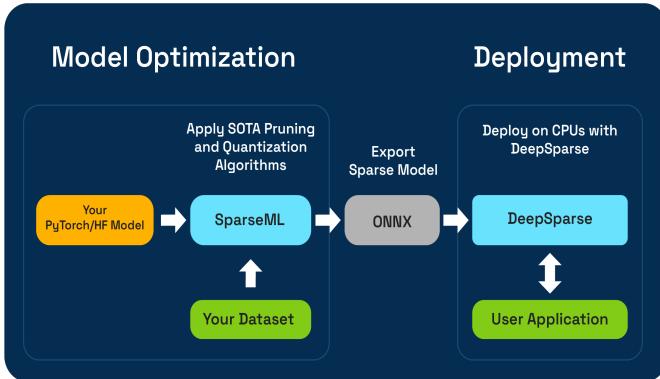


Figure 4: SparseML Pipeline<sup>23</sup>

capacitors, and inductors, achieving a True Positive Rate (TPR) of 99.999% with only a 2.67ms average inference time on a GTX 1050 2GB GPU (released in 2016) which is an impressive result. This work shows that a CNN is a viable approach to the problem of component identification, helping to inform the design of the project’s computer vision system.

The paper by Muminovic and Sokic<sup>24</sup> discusses the use of SVM to characterise resistors, by identifying the resistor’s centroid (centre of mass), determining the resistor’s orientation, and then analysing the bands to determine the resistor’s value. This is directly applicable to the project, as it is a viable approach to identifying resistors, which are very distinct from other components, given the presence of colour bands. The paper’s novel approach to identifying the resistor values allows it to achieve high accuracy (86%, however, the test images used are resistors that are positioned far away from the camera, as shown in Figure 3, and so the bands are very small, which will not be the case in the project’s vision system, so the accuracy will likely be higher) which makes it a very promising resource for the project.

## 2.2 Real-time Computer Vision Architectures

For the problem of component identification, the computer vision system must be able to identify components in real-time, as the components may eventually be moving on a conveyor belt, captured using a camera facing down at the components. This means that the computer vision system must be able to identify components in a very short amount of time, and so the system must be computationally efficient. The envisioned system

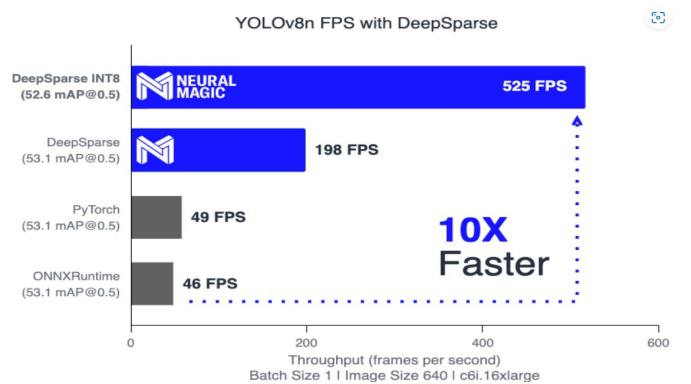


Figure 5: DeepSparse Performance<sup>22</sup>

is designed to be self-contained, so it must also be able to run solely on the Raspberry Pi 4, which has a 1.8GHz quad-core 64-bit ARM Cortex-A72 CPU, and up to 8GB of RAM<sup>11</sup>, while still being accurate and able to control the other systems in the project.

This requires the Computer Vision system to be both computationally efficient and accurate, which is a difficult balance to strike.

For the explicit purpose of identifying electronic components, the paper by Chand and Lal<sup>3</sup> compares a range of different object detection architectures, including YOLOv3, YOLOv4 and Faster SqueezeNet, with YOLOv4 achieving a mAP (mean Average Precision) of 98.6%.

YOLO (You Only Look Once) is a real-time object detection architecture, that is used very commonly in computer vision applications and is very effective at identifying objects in real-time Terven and Cordova-Esparza<sup>40</sup>. It makes use of a single CNN that takes in an image and outputs a list of bounding boxes and class probabilities for each bounding box. This is in contrast to other object detection architectures, such as R-CNN, which uses a CNN to propose regions of interest and then uses a second CNN to classify the regions of interest.

Other papers like Guo et al.<sup>16</sup> also comment on YOLOv4’s effectiveness at identifying electronic components, achieving 93.94% mAP on a dataset of 20 different components, and the paper also comments on YOLOv4’s ability to run in real-time, achieving 67 FPS, albeit on a powerful NVIDIA TITAN Xp GPU. For the Raspberry Pi 3B, the paper by Sismananda et al.<sup>39</sup> has shown to run YOLOv3 at a very low 1 FPS, with an IoU (Intersection over Union) accuracy of 86.7%, which is very low.

However, efforts made by Neural Magic<sup>22</sup> in the optimisation of YOLOv5 and YOLOv8 show performance improvements of up to 10x on

CPUs, through their open-source optimisation toolkit SparseML<sup>23</sup> and CPU inferencing runtime, DeepSparse<sup>21</sup>; a very promising result.

## 2.3 Training Methods

When training a neural network, it is important to have training data. For a computer vision system, this means having a dataset of images that are representative of the conditions that the system will be used in and are well-labeled with bounding boxes and class labels. It is important to have a large dataset to ensure that the model is robust and generalises well, which is time-consuming to create.

To solve this problem, a paper by Yang et al.<sup>45</sup> proposes many different training techniques, the most promising of which is semi-supervised learning. In this approach, the model is trained on a small labeled dataset, and then used to label a large unlabelled dataset, which is reviewed and corrected by a human.

This process is repeated until the model is sufficiently accurate, and then the model is trained on the large labeled dataset. This approach is very promising, as it allows for the training of a robust model with a large dataset, without the time-consuming process of manually labeling the entire dataset.

## 2.4 Mechanical Design

### Transport Mechanisms

The paper by Dhenge et al.<sup>9</sup> depicts a conveyor belt system that transports nuts and bolts to a computer vision system for identification, which aligns with the goals of the project. The hardware prototype is shown in Figure 2, and a down-facing webcam is used to capture images of the components as they pass by. The webcam uses Principle Component Analysis (PCA) to identify the components, which will be discussed in the next section.

The components then separate into two chutes, one for nuts and one for bolts, using a stepper motor, which may be useful for the future design of the sorting system. Additionally, the approach taken by the paper Quillay et al.<sup>34</sup> to sort Philippine table eggs also features a similar conveyor belt system, a down-facing camera and an arm to sort the eggs into different categories.

The approaches taken here are similar to industrial sorting systems seen in videos while

researching for this project; this approach seems to answer three major design questions for the project; how to transport the components to the computer vision system; how to transport the components from the computer vision System to the sorting system; and the placement of the camera.

Other approaches include vibratory feeders<sup>20</sup>, which use precise vibrations to orientate and transport components, and pneumatic systems from Abe et al.<sup>1</sup> (also suggested by Dr. Stott when dicussing transporting mechanisms in project meetings), use air pressure in tubes to transport components. These approaches require more complex hardware, and lack the easily achievable precision of the conveyor belt system, for example in the case of the pneumatic system, a complex network of tubes and valves and an air compressor is required, which is not practical for the project. Robotic arms are commonly used in the industry for sorting, however, this is not viable as this would either require an expensive robotic arm, or a complex system of motors and actuators to move the components.

From the approaches above, it seems that a conveyor belt with a down-facing camera is the most viable approach to transporting the components. Initially, the aim was for the system to be semi-autonomous, with a design allowing the camera to face upwards. This configuration would enable the user to place the component on an acrylic plate directly above the camera for immediate identification, enabling easy user interaction as the view of the component is not obstructed. Having the camera face-down would block the user's view of the component, which would be an inconvenience, and as such, the current design of the system features an up-facing camera. However, this will be changed to reflect the research outlined above, in the next stage of the project.

### Bowl Feeders

While researching for this project, and especially in videos<sup>43</sup>, it seems most industrial sorting machines use a Vibratory Bowl Feeder (VBF) to help feed components into the sorting system; as shown in Figure 1, the VBF consists of a bowl that vibrates coupled with a spring and electromagnet. The paper by Nam et al.<sup>26</sup> explores the optimal design of a VBF for USB keycaps, by attempting to identify the ideal parameters for the structure of the bowl, sorting track, mounting adapter, and

suspension system. The paper also uses modal analysis to determine the natural frequencies of the system and uses this to avoid resonant conditions that might cause inefficient or erratic operation.

This paper is useful as it provides a comprehensive overview of the design of a VBF, and provides a good starting point for the design of the VBF. In the future, the project may make use of one for fully autonomous sorting. The paper also provides a good overview of the design considerations for the VBF, and so can be used as a reference during the design process.

Additionally, Reinhart and Loy<sup>36</sup> delve into a mathematical model of a VBF, optimising more on the overall performance of the VBF rather than efficiency, and Silversides et al.<sup>38</sup> provides a good overview of the forces involved in the operation of a VBF, strengthening the basis for its design and viability.

The paper by Zhengyang Zhang<sup>46</sup> outlines a sorting system for vials and does not make use of a VBF, instead opting for a turntable design that mechanically orientates the vials. It primarily operates by using a design that is specific to the geometry of the vials, and so does not apply to this project, however, it does provide a good insight into the design of a sorting system.

An alternative to the VBF could be a robotic arm; fine control over movement would be necessary as the components are small and may be tangled, however, as mentioned before would require its own set of sensors and camera systems. As the components can be tangled, there are not many viable alternatives to the VBF or a robotic arm, so between these two solutions, it seems that the VBF is the most viable approach if the system is to be fully autonomous, as the vibrations of the VBF would help to untangle the components.

## 2.5 Key Takeaways

After reviewing the literature, the following key takeaways were made:

The most viable approach to the problem of component identification is to use a CNN, specifically the YOLOv8 architecture, as they are very effective at classification tasks, which translates to being able to identify electronic components. When the ability to optimise the model using SparseML and DeepSparse, the model may be able to run on the Raspberry Pi 4 in real-time, which is a key requirement of

the project. Additionally, the YOLO architecture may be covered in the modules Deep Learning COMP70010 and Machine Learning COMP70014, enriching the understanding of this architecture.

A down-facing camera is the most viable approach for the future of the project, as it allows for a conveyor belt system to solve the problem of component transportation, and so the design of the computer vision system will be based on the research outlined above in the next stage of the project.

A VBF is the most viable approach to the feeding mechanism of the sorting system, and so the design of the VBF (if employed in the project) will be based on the research outlined above.

As such for this project, the YOLOv8 architecture will be employed, making use of the SparseML and DeepSparse optimisation tools with semi-supervised learning to train the model.

## 3 Implementation

Three systems comprise the project:

- The Hardware, Mechanics and Electronics
- The software that coordinates the various components
- The Computer Vision system

It is important to note that for the initial version of the system, prior to conducting a comprehensive literature review, the system's design incorporated a face-up camera intended to identify components placed on an acrylic plate directly above it. This design was chosen as it was the simplest design that would allow for component identification and could be easily implemented using available resources. Additionally, the camera used in the system came with a short ribbon cable, which limited the distance between the camera and the Pi, which would have made the face-down design more difficult to reasonably implement, especially with the LCD.

### 3.1 Hardware, Mechanics and Electronics

The hardware, mechanics and electronics (HME) of the system are the physical components that make up the system. It is necessary to first consider the HME of the system as they are the foundation upon which the rest of the system is built; for example, when designing the computer vision system, it should utilise training data that

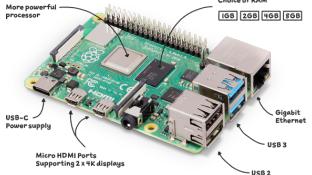


Figure 6: Raspberry Pi 4 Model B 2GB<sup>11</sup>



Figure 7: DFRobot 7" Touchscreen Display<sup>8</sup>



Figure 8: Okdo Adjustable Focus OV5647 Camera<sup>29</sup>



Figure 9: Power consumption of the system

was collected using the HME that the system will be deployed on, ensuring that the system is robust to the conditions it will be used in. As such, the main focus of this stage has been the HME.

To keep track of the HME, a spreadsheet was created that contains all the components of the system (otherwise known as the Bill of Materials or BOM), as well as the cost of each component. A sample of this spreadsheet can be found in Appendix 27.

### 3.1.1 Hardware

For the initial stage of the project, the main hardware components of the HME are:

- Raspberry Pi 4 Model B 2GB<sup>11</sup>
- 7" Touchscreen Display<sup>8</sup>
- 24V dc, 6.25A, 150W Power Supply
- Okdo Adjustable Focus OV5647 Camera<sup>29</sup>
- LED Light Ring

#### Raspberry Pi 4 Model B 2GB

The Raspberry Pi 4 Model B was chosen as the main component of the system and is the central hub that all other components are connected to. This model was chosen as it is regarded as a reliable SoC and is widely used in the industry, and was used in the module ELEC96018 Embedded Systems. It has a large amount of software and driver support, ensuring confidence in finding solutions to any potential issues that may arise. Additionally, it has GPIO pins that can be used to control other components which will be necessary in future stages. It also has a dedicated CSI camera port which allows for a camera to be connected directly to the SoC, which is necessary for the computer vision system.

It also has WiFi support, allowing for SSH and remote development software, as well as an HDMI port for the display. Originally, the 4GB model was ordered however an issue with the EE Stores resulted in receiving the 2GB model instead. This

was not an issue as the 2GB model is sufficient for the initial stage of the project, however, if it is found to be a bottleneck in future stages, a more powerful model may be used. The decision to not return the 2GB model is because the EEStore was not available over the Christmas break when the bulk of the work was done for this report.

An alternative to the Pi family of SoCs could be an Arduino, but while they are capable of controlling the components of the system, and potentially drive an LCD, their microcontrollers are not powerful enough to run the computer vision system and stream video from the camera, so they cannot be used as a replacement for the Pi. However, in the future stages of the project, the Pi may delegate control of the components to an Arduino to free up processing power for the computer vision system, if it is found to be a bottleneck.

A viable alternative to the Pi could be a Nvidia Jetson Nano<sup>27</sup>, as it is designed for computer vision applications and has a dedicated GPU. The dedicated GPU would reduce the likelihood of the computer vision system being a bottleneck, and it has a CSI port for the camera; in terms of CPU performance, the Nano is weaker than the Pi, having a quad-core ARM Cortex-A57, whereas the Pi has a quad-core ARM Cortex-A72<sup>11</sup>. However, the worse CPU performance is offset by the dedicated GPU, and the Nano has 4GB of RAM, making the Nano a very attractive alternative to the Pi. The Nano's drawback is that it is significantly more expensive than the Pi (up to 3x from retailers), and given the ability to optimise the computer vision system as discussed in Section 2 (Background), the Pi is still a viable choice for the computer vision system.

To address the lack of a dedicated GPU for inference, an article from Medium<sup>2</sup> benchmarked the Pi's performance on various computer vision tasks with various accelerators like the Intel

Neural Compute Stick 2 and the Google Coral USB Accelerator against other SoCs like the Jetson Nano and the Coral Dev Board. It was found that the Coral Dev Board was the best performing, however the Pi using optimisation libraries like TensorFlow Lite showed promising results, which helps to strengthen the argument for using the Pi for the computer vision system, especially with the SparseML and DeepSparse libraries that were discussed in Section 2 (Background).

### 7" Touchscreen Display

The DFRobot 7" Touchscreen Display was chosen as it is a relatively cost-effective display that is compatible with the Raspberry Pi. It has touchscreen support and has a Raspberry Pi 4 mount on its back, as well as HDMI adapter boards to connect to the Pi. This means that a physical mount for the Pi does not need to be designed, and only a mount for the display is required. The display is also powered by the Pi, so no additional power supply is required.

### 24V dc, 6.25A, 150W Power Supply

Two parameters were taken into consideration when choosing a power supply: the power output and the voltage.

- **Power Rating**

The power supply must be able to drive all components in the system with overhead in case of spikes in power consumption. In this first stage, the system only consists of the Raspberry Pi, the display, LED lights and the camera, making for a power consumption of under 20W, recorded using a smart plug with a power meter as shown in Figure 9. However, in the future, the system will likely need to drive additional motors and sensors, so the total power consumption will be higher. Even with accounting for this, the power consumption will be under 100W, so a 150W power supply will be sufficient, allowing a 50W overhead. In hindsight, this may be considered excessive.

- **Voltage Rating**

The voltage of the power supply must be greater than or equal to the highest voltage of any component in the system; this is necessary as there are only step-down converters available on hand, so choosing a higher voltage allows the stepping down of the higher voltage to the required voltage for each component. 24V was chosen as it is a common voltage for stepper motors, which may be used in future stages of

the project. It is also used for some LED strips, which are used in this stage. The Pi 4 uses 5V and the LED Ring uses 12V, which the 24V power supply can be stepped down to.

### Okdo Adjustable Focus OV5647 Camera

For the Computer Vision system, a camera is required to capture images of the components. The Okdo Adjustable Focus OV5647 Camera was chosen as it is CSI compatible, meaning it can be connected directly to the Raspberry Pi, and has a manual focus ring, allowing it to be used as a macro camera to capture images of small components placed directly above it. The manual focus ring allows the camera to be specifically tuned to the design of the system, allowing for the best possible image quality. It also has a 5MP sensor<sup>30</sup>, which is sufficient for the computer vision system, as high-resolution images would be preprocessed and reduced, only adding to the amount of processing required.

### LED Light Ring

An LED Ring is required to illuminate the components so that the camera can capture images of them. Initially, a 12V LED Ring was chosen as it would allow for a uniform light source and could be mounted directly below the camera; however, for reasons explained in the next section, it was replaced with a 5V WS2812B LED strip as it solves the issues faced with the LED Ring.

#### 3.1.2 Mechanics

To house the components, a 3D-printed housing was designed. It features a modular design, allowing for an iterative design process where different components of the system can be worked on independently. It also allows for easy replacement of components should the design prove to be inadequate. The housing was designed in FreeCAD<sup>12</sup>, a free and open-source parametric CAD software.

The parametric design allows for easy modification of the design should changes need to be made - with good design practices, large systematic changes can be made by simply modifying a few numerical parameters. The design was then printed using a 3D printer, a heavily modified Voxelab Aquila C2.

All parts are printed in PLA (Polylactic Acid), a biodegradable thermoplastic that is commonly used in 3D printing. The design at this stage does not require any special properties from the

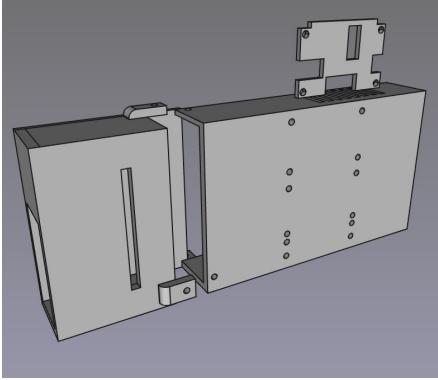


Figure 10: PSU Housing

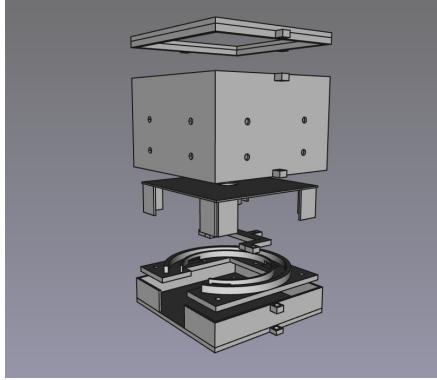


Figure 11: Camera Housing

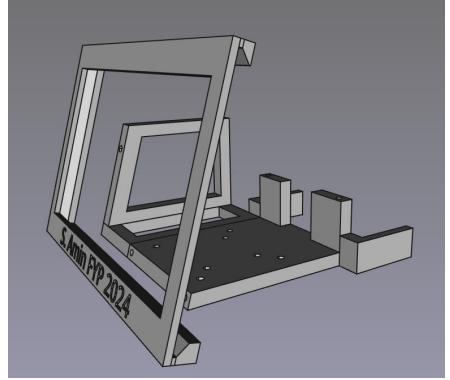


Figure 12: LCD Display Housing

material, so PLA is sufficient.

For this stage of the project, I must design 3 major components for the housing:

### PSU Housing

The PSU housing contains the power supply and ensures that all high-voltage components are safely enclosed. It also houses the power switch, the power socket and the terminals. As step-down converters are required for the Pi and LED Ring, the PSU housing also contains mounting points for the step-down converters, as well as a mounting plate to ensure that the step-down converters are mounted flat to the PSU housing.

### Camera Housing

The camera housing contains the camera and the LED Ring. It also contains the mounting points for the camera and LED Ring. The camera housing will also mount an acrylic plate above the camera, so components can be placed on the plate and be imaged by the camera.

From bottom to top in Figure 11, are the bottom casing, LED Ring mount, camera mount, light diffuser, middle casing and top casing.

### LCD Display Housing

The LCD housing contains the LCD and the Raspberry Pi. As the 7" DFRobot LCD has a Raspberry Pi 4 mount on its back, an explicit mount for the Pi is not required. The mount will position the LCD at an angle, so it can be viewed from above. The LCD simply slides into the mount, allowing for easy removal and also contains mounting points to secure the LCD to the camera housing.

The LCD housing is split into two parts; the LCD cover, where the LCD slides into, and a base plate, which mounts to the camera housing. The

LCD cover is then mounted onto the base plate, securing the LCD in place.

The components are secured using brass M3 heat-set inserts and M3 screws. The heat-set inserts are inserted into the 3D-printed parts using a soldering iron, and the components are then screwed into the inserts. This allows for easy removal of components should I need to make changes to the design, following my modular design principle.

The design of the system at this stage is shown in the Appendix in Figure 28.

### 3.1.3 Electronics

As the project is a physical system, several electronic components are required to make the system work.

#### Power Supply

As mentioned in Sections 3.1.1 and 3.1.2, the system is powered by a 24V, 6.25A, 150W power supply, and must be stepped down to 5V for the Pi and 12V for the LED Ring. This is achieved using an XL4015 Step Down Converter<sup>19</sup>, a variable step-down converter that can output up to 75W. The high power rating of the step-down converter allows for a large overhead, which is useful for future stages of the project where additional components may be added to the system.

#### Raspberry Pi 4

To power the Raspberry Pi, a standard USB-A to USB-C cable was chosen and was connected to a female USB-A breakout board connected to the 5V step-down converter. This allows for easy removal of the Pi should changes be made to the system, and prevents the unnecessary cutting of cables.

As the Pi is mounted to the LCD, a USB-A to Micro USB cable is used to connect the Pi to the display, which is then connected to the Pi's HDMI

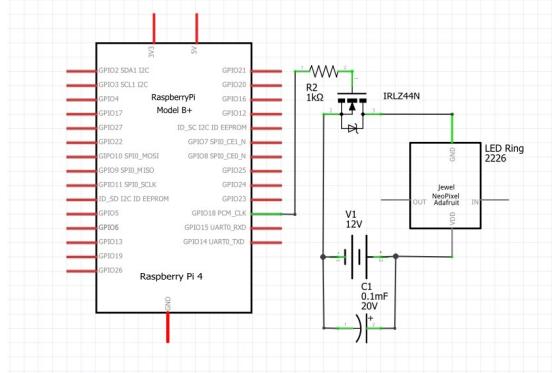


Figure 13: Wiring Schematic for MOSFET, made with Fritzing<sup>13</sup>

port, using a supplied HDMI adapter board.

### LED Lighting

As mentioned in Section 3.1.1, initially, a 12V LED Light Ring was used and was changed to a WS2812B 5V LED strip, for issues described later. To enable the ability to dim the LED Ring, a MOSFET was used to control the LED Ring. As shown in Figure 13, the LED Ring was connected to a 12V supply (achieved using the 12V step-down converter) and parallel to a  $100\mu\text{F}$  capacitor, which is used to smooth out the voltage spikes.

The LED Ring was then connected to an N-channel MOSFET IRFZ44N<sup>17</sup>, which is controlled by a PWM-enabled GPIO pin on the Pi. This GPIO pin is connected to a  $1\text{k}\Omega$  resistor, which prevents potential overcurrent from damaging the Pi. The MOSFET is then connected to ground, completing the circuit.

The IRFZ44N is a logic level MOSFET, meaning it can be fully turned on with a 3.3V gate voltage, which is the voltage of the Pi's GPIO pins, and the high switching speed (up to 1 mHz) of the MOSFET allows for the LED Ring to be PWM controlled, allowing for the brightness of the LED Ring to be controlled.

#### 3.1.4 Problems and Solutions

3D printing is typically an iterative process, where the design is printed, tested, and then modified based on the results. This further reinforces the need for a modular and parametric design, as it allows for easy modification of each component. After a few iterations with errors due to tolerances and slightly incorrect measurements, (not relevant to the technical content of this report) a design was produced that physically fit together and mounted the components as intended, however when combined with the software and



Figure 14: Main UI, captured using RealVNC<sup>35</sup>

electronics, a few problems were encountered that were overcome by making some design decisions that were discussed with the project's supervisor, Dr. Stott.

### Mechanical Design Problems

The first design problem was the initial design of the LCD mount. As mentioned in Section 3.1.2, the LCD housing is split into two parts; the LCD cover, where the LCD slides into, and a base plate, which mounts to the camera housing. The initial design of the LCD cover was only mounted to the base plate using two M3 screws at the bottom (Figure 17), which made it unstable and prone to wobbling. Over time, this would put stress on the plastic, causing it to break. To solve this, a third mounting point was added to the LCD cover in the form of a brace that reaches halfway up the LCD cover, as shown in Figure 18, increasing the stability of the LCD, and preventing it from wobbling.

The second design problem was the design of the camera housing. As mentioned in Section 3.1.2, the camera housing contains the camera and the LED strip. The camera itself is not a macro lens, but rather a standard lens with an adjustable focus. The workaround is to set the focus as close as possible, and then determine the optimal distance between the camera and the object to be imaged. This is done by placing an object on the acrylic plate. Initially, the height of the camera housing was too short as shown in Figure 19, resulting in a blurry image, which was remedied by simply changing the parametric parameters that define the height of the camera housing, as shown in Figure 20.

The above design problems were solved by simply modifying the design, however, the



Figure 15: Glare from LED Ring

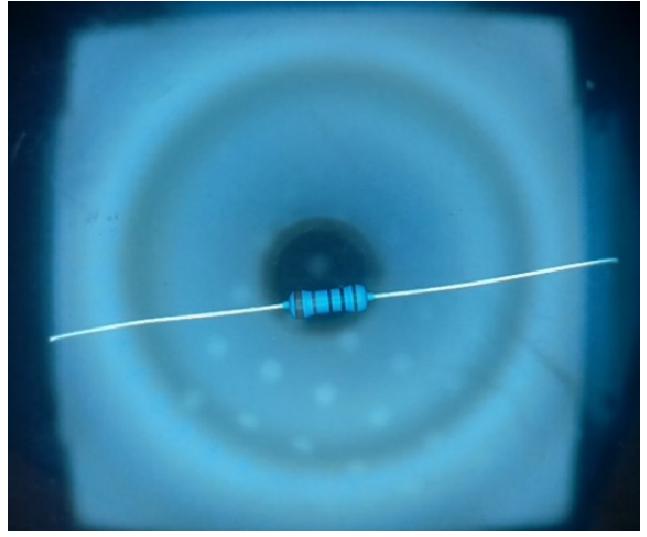


Figure 16: Diffused Light

third design problem was more complex. As mentioned in Section 3.1.2, the approach to imaging components is to place them on the acrylic plate, and then image them using the camera. However, when doing so, the light from the LED Ring would reflect off the acrylic plate as shown by Figure 15, resulting in an incredible glare that would obscure the image of the component. There was an attempt to 3D print a light diffuser to reduce the glare Figure 16, however, this was not effective and instead caused the entire image to be affected by the reflection of the LED Ring. This was a major design problem as, in the non-diffused image, any components in the glare of the ring are obscured, and in the diffused image, the entire image has incorrect colours and is partially obscured. To solve this, three solutions were considered:

- **Polarising Filter**

Polarising filters can be used to reduce glare by filtering out light that is polarised in a certain direction.

- **Different Light Source**

The LED Ring was chosen as it provides a uniform light source as it would surround the camera, however, it is possible to use a different light source that does not cause glare, for example, an LED strip that is mounted above the camera near the acrylic plate.

- **Down-facing Camera**

Instead of imaging the components from below, the camera can be mounted such that it is facing down, and images the components from above, as explored in Section 2 (Background). The issue is that the light of the LED Ring bounces

off the bottom layer of the acrylic plate, so any components on top are obscured.

However, in the end, none of these could be considered because of the electronics design problem discussed in the following section.

### **Electronics Design Problems**

Attempting to dim the LED Ring resulted in very sporadic flickering, and it did not react with changes to the frequency of the PWM signal despite the MOSFET control circuit shown in Figure 13. Originally, an issue with the MOSFET was considered as initially a MOSFET with a higher gate threshold voltage (IRF520N) was used, and so it was replaced with a MOSFET with a lower gate threshold voltage. However, this did not solve the issue, and a capacitor was added in parallel to the LED Ring to buffer the voltage, but to no avail. The LED Ring seemed to be a simple ring of LEDs, with no additional circuitry, so it was not clear it was not dimmable. Supervisor Dr. Stott was also unsure as to why it was not dimmable and proposed a different circuit with an inductor and diode but this requires a much higher switching frequency than the Pi's PWM signal can provide, and so it was not viable.

For these reasons, and after reviewing the literature as discussed in Section 2, it was decided that the camera mount was to be redesigned to be down-facing (as a conveyor belt will also be used in the future) and to replace the LED Ring with a WS2812B LED strip, as these are designed to be dimmable. They are also addressable, however, this is not a feature that is required at this stage of the project.

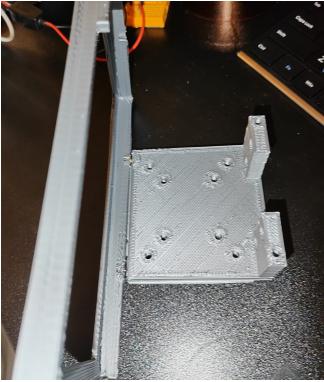


Figure 17: Unbraced Screen

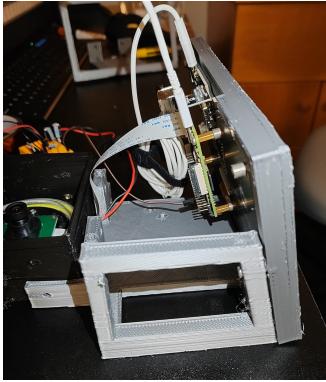


Figure 18: Braced Screen

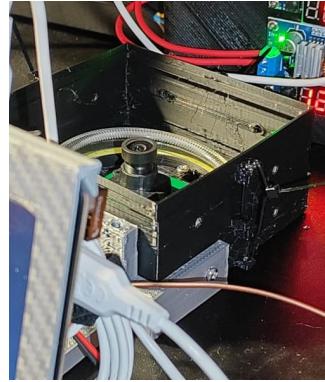


Figure 19: Short Camera Housing

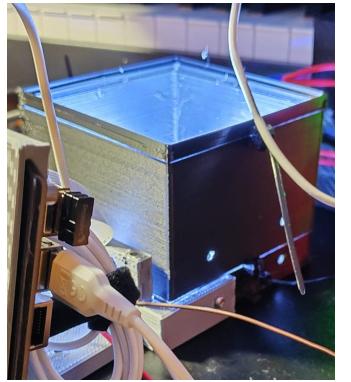


Figure 20: Optimal Camera Housing

### 3.2 Software

The entire system is controlled by a Raspberry Pi 4B running a very lightweight Linux distribution called DietPi<sup>10</sup>. The goal for the entire system is to be run locally on the Pi, so the Pi will be responsible for running the computer vision system, controlling the hardware, and running the user interface. In order to achieve this, the following software components are required:

- The User Interface
- The Computer Vision system
- The Hardware Control system

The system's software is written in Python<sup>33</sup>, which is well suited for the project given the availability of libraries for computer vision and hardware control. Again, the concept of modularity underpins the design of the software, and so each component is designed to be independent of the others and can be developed in parallel.

All written Python code adheres to a style defined in a .pylintrc file, which is a configuration file for the pylint<sup>32</sup> linter. This ensures that the code is consistent and readable. The code is also documented using docstrings for readability and maintainability. Each software component is designed to be self-contained so they can be run independently for testing purposes.

All software constants are defined in a constants.py file, which is imported by all other Python files, allowing for easy modification of constants that define the behaviour of the system which is useful for testing and debugging.

Additionally, a lot of thought went into streamlining the development process to ensure that the system is easy to develop and maintain. For example, Visual Studio Code's<sup>4</sup> Remote SSH

extension is used to develop the system remotely, as it allows developing the system on a much more powerful laptop, while using the familiar VSCode environment with any extensions that help streamline the development process. This is crucial as the Pi is not very powerful, so compiling and running the system on the Pi may be slow and cumbersome.

The Pi uses Git<sup>14</sup> for version control, and the repository is hosted on GitHub<sup>15</sup>, with a dedicated branch for the Pi that is regularly updated with the main branch. The vision system and the laptop also maintain their own Git branches which are regularly updated with the main branch, ensuring that all systems are running the same code. This is crucial as it enables development on a more powerful laptop, and then synchronises the changes to the Pi, without having to manually copy any files over.

During development, the Pi connects to the laptop's WiFi hotspot and is configured to be discoverable with the Pi's hostname, facilitating easy access to the Pi through SSH and a VNC Viewer without needing the Pi's IP address, which is dynamic. The Pi also makes use of SSH keys, allowing connection to the Pi without needing to enter a password every time, ensuring quick and easy access.

#### User Interface

As the system is intended to be self-contained, it is important to have a user interface that displays all relevant information to the user and allows them to interact with the system. The UI is displayed on the 7" touchscreen display, and is written using Pygame<sup>31</sup>;

This design decision was made after experimentation with other UI libraries like Tkinter<sup>41</sup>, however it was found that Pygame was

the most viable for several reasons; it contains a module for camera streaming, which is used to display the camera feed on the UI, and can also be used to feed the camera frames to the Computer Vision system; there is a very large amount of control provided by Pygame over the UI, especially in terms of performance. This is crucial as the UI is only there to provide information to the user, and so it must not interfere with the computer vision system or the hardware control system. With an additional library named pygame\_gui<sup>25</sup>, it is possible to create a UI with GUI elements like buttons, text boxes, and drop-down menus, making for a fully comprehensive UI library.

In Figure 14, the current version of the UI is shown. On the left, the camera feed and camera frame rate are displayed, with the system CPU and RAM usage displayed, for monitoring purposes. There is a slider to adjust the power of the connected LED Ring light, as well as two buttons related to the Computer Vision system.

The right side is blank for now but will be used to display information about the computer vision system and the hardware control system.

### 3.3 Computer Vision

The computer vision system should be trained on data that is representative of the conditions it will be used in, and because of the previously mentioned issues with the mechanical design of the system, the computer vision system was not developed at this stage. The data collection process is manual and time-consuming, so it is far more efficient to ensure that the mechanical design is finalised and that the foundation of the project is robust and reliable. As will be discussed in Section 4 (Project Plan), the mechanical design will be overhauled in the next stage, so if data was collected for the computer vision system in this stage, it may have been rendered useless in the next stage.

However, this does not mean that no work was done on the computer vision system. As per Section 2 Background, it was determined that YOLOv8 is the most suitable model for the computer vision system. As such, several tools were developed to aid in the development of the computer vision system.

Additionally, careful consideration of the Computer Vision system pipeline was done in this stage.

#### 3.3.1 Data Collection

For the task of component identification, the computer vision system must be trained on data that is representative of the conditions it will be used in. As such, the data collection process is manual and time-consuming. For this reason, a customtkinter<sup>37</sup> script was developed to aid in the data collection process. The script allows easy labelling of images with bounding boxes and class labels; and will save the labels in a format that is compatible with the YOLOv8 model. The script also streamlines the labelling process using shortcut keys, and sorts each image into a folder based on the class label, making it easy to manage the data. Given that this is a custom script, features can be easily added or removed - it is likely that features such as component orientation will be added, so a custom script is very useful.

This script works in conjunction with the UI, seen in Figure 14, running on the Raspberry Pi (made possible using RealVNC<sup>35</sup>, a remote desktop application), allowing for very easy labelling of images with bounding boxes and class labels.

The script has support for multiple components, including:

- Resistors
- Capacitors
- Ceramic Capacitors
- Inductors
- Diodes
- MOSFETs
- Transistors
- LEDs
- Wires
- ICs

A screenshot of the script at work can be seen in Figure 23.

A Jupyter Notebook<sup>18</sup> was also developed to aid in developing the computer vision system. This is common practice as it allows code to be run and the data associated with it to be inspected, which is very helpful for debugging purposes. It has the capability of training every model that will be used in the computer vision system and contains features like data augmentation, checkpointing, and model evaluation.

The data augmentation feature is very useful as it retains the original data, should more labels need to be added or modified in any way. The checkpointing feature allows facilitates saving the model at any point during training, which serves as a backup but also allows resuming training from a certain point and is useful to protect against overfitting. The model evaluation feature

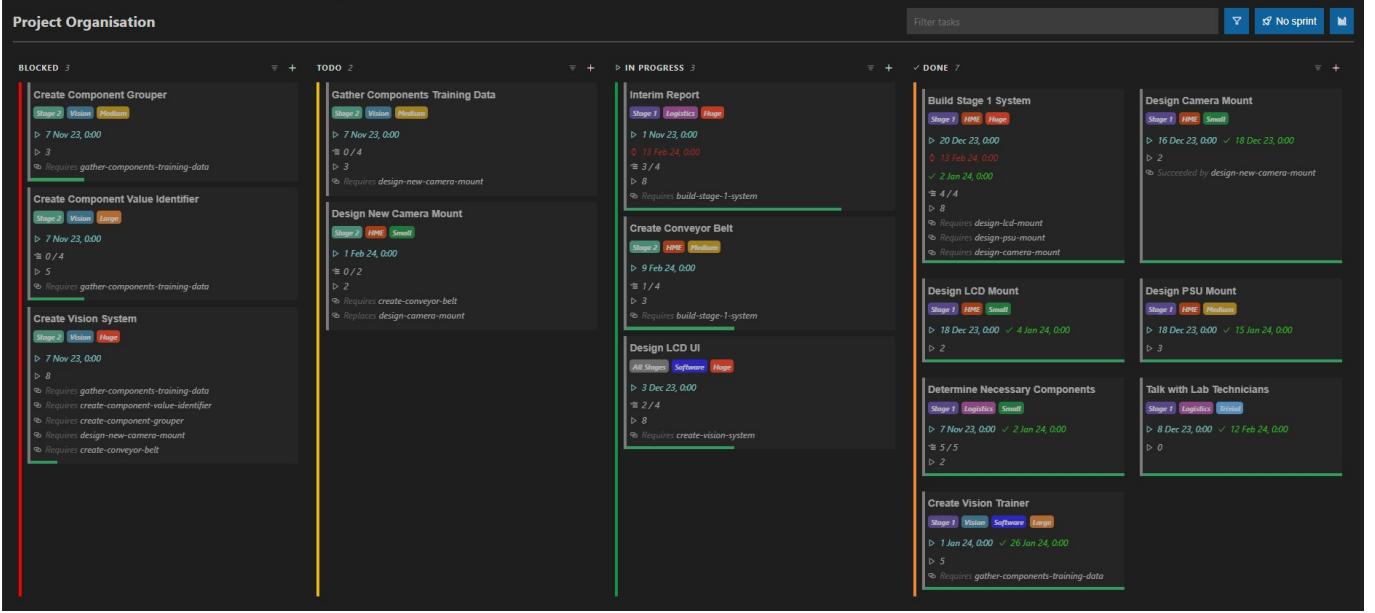


Figure 21: Project Plan

allows for the evaluation of the model on a test set and provides useful metrics like precision, recall, and mAP. This is useful for evaluating the model's performance, and also for comparing the performance of different models.

### 3.3.2 Vision System Pipeline

The computer vision system pipeline is the sequence of steps that the computer vision system takes to identify components.

The system pipeline is as follows:

- **Preprocessing**

The image is preprocessed to improve the quality of the image and to make it easier for the model to identify components, ensuring uniformity and consistency in the data.

- **Component Detection**

The image is fed to the model, which outputs a class, confidence score, bounding box, and orientation for each component.

- **Component Value Identification**

The value of the component is identified from the sub-image within the bounding box and is used to further classify the component.

While a single model can be used to identify components, it is necessary to use two models to identify the value of the component. This is because the value of the component may differ depending on the type of component. For example, the value of a resistor is read from the colour bands, whereas the value of a capacitor is read from the text printed on the component.

As such, each component may require a different model to identify its value.

This is not an issue as it conforms to the modular design of the system, and allows for easy extensibility. For example, if a new component is added to the system, it is easy to add a new/reuse an existing model to identify the value of the component; however the component detection model will need to be retrained as the dimensionality of the output layer will change due to the addition of a new class. The Jupyter Notebook developed for the Computer Vision system makes this process very trivial, as it would simply require adding the new component to the list of components, and the Notebook will handle the rest.

## 4 Project Plan

For the planning of the project, the decision was made to leverage the experience gained from a 6-month placement by adopting more traditional software development methodologies. This approach included the utilisation of a Kanban board to monitor the project's progress effectively.

In this Kanban board, labels were employed to categorise each task into different categories, such as which component of the system the task is related to, the priority of the task, and the stage of the project the task is related to. Each task had a description detailing what needed to be done, and some others had dependencies and checklists of sub-tasks that needed to be completed. Dates were also assigned to each task to ensure that the project

The screenshot shows a task management interface for a 'Create Conveyor Belt' task. The task details include:

- NAME:** Create Conveyor Belt
- DESCRIPTION:** Design the conveyor belt; use aluminium extrusion to form the frame and design 3D printed mounts for the motor and rollers. Integrate and wire the conveyor belt into the system.
- SUB-TASKS:**
  - Design motor and idler mounts (checked)
  - Design and fit rollers
  - Wire electronics
  - Integrate into system
- COLUMN:** In Progress
- ASSIGNED TO:** Assigned to
- STARTED DATE:** 09/02/2024
- DUE DATE:** dd/mm/yyyy
- COMPLETED DATE:** dd/mm/yyyy
- PROGRESS:** 0.4
- TAGS:**
  - Stage 2
  - Mechanics
  - Medium

Figure 22: Example of Kanban Task

was on track. An example of a task on the Kanban board is shown in Figure 22, detailing the task of developing the conveyor belt. A description of the task is given, as well as difficulty, workload, and the stage of the project it is related to. There are also subtasks and dependencies listed, as well as the date the task was created and progress of the task.

The main labels used in the Kanban board are as follows:

#### • Stage

This label is used to categorise the task into the stage of the project it is related to. The stages are as follows:

- Stage 1
- Stage 2
- Stage 3
- Stage 4

#### • Component

This label is used to categorise the task into the component of the system it is related to. The components are as follows:

- HME (Hardware, Mechanics, Electronics)
- Vision
- Logistics
- Software

#### • Workload

This label is used to categorise the task based on the expected workload. The workloads are as follows:

- Trivial (0)
- Tiny (1)
- Small (2)
- Medium (3)
- Large (5)
- Huge (8)

For the remainder of the project, this Kanban board will be used to track the progress of the project.

As mentioned previously, to keep track of the components of the system, a spreadsheet was maintained which is shown in Appendix 27.

## 4.1 Stage 1

The original plan for Stage 1 was to develop a system that could identify and classify components. However, the development of the foundation and hardware, as well as software to control the hardware, took longer than expected. This was because the vision system needs to be trained on data that is representative of the conditions it will be used in, which requires the mechanical design to be finalised, which can be seen from the dependencies listed on the Kanban board in Figure 21. As such, Stage 1 was split into two stages, with the first stage focusing on the development of the foundation of the system, and the second stage focusing on the development of the computer vision system.

While non-trivial, the strong foundation of the project instills confidence that Stage 2 can be accomplished within the next 5-6 weeks. The current Stage 1 system is shown in Figure 24 and the system diagram is shown in Figure 26. This diagram clearly shows the interdependency



Figure 23: Dataset Labeller Tool

of all the components of the system, and how the development of the hardware is a prerequisite for the development of the software. This further reinforces the need to underpin the project with a strong, modular and parallel development approach.

## 4.2 Stage 2

For the Stage 2 system, the system will be able to identify and classify components. The main tasks for this stage are as follows:

- **Conveyor Belt System**

A conveyor belt system will be developed to move the components from the input to the vision system.

- **Component Classification**

The system will need to be able to classify the components. This will be done using a YOLOv8 model, which will be trained on a dataset of components.

- **Component Value Identification**

After the component has been classified, the system will need to identify the value of the component. This will be done by using a separate model to identify the value of the component. In particular, the model for the resistors will be the highest priority given their complexity, followed by text-based values for capacitors and inductors, and finally ICs and MOSFETs.

- **User Interface**

The UI of the system will need to be updated to reflect the system's new functionality. This will be done using the Pygame library<sup>31</sup>.

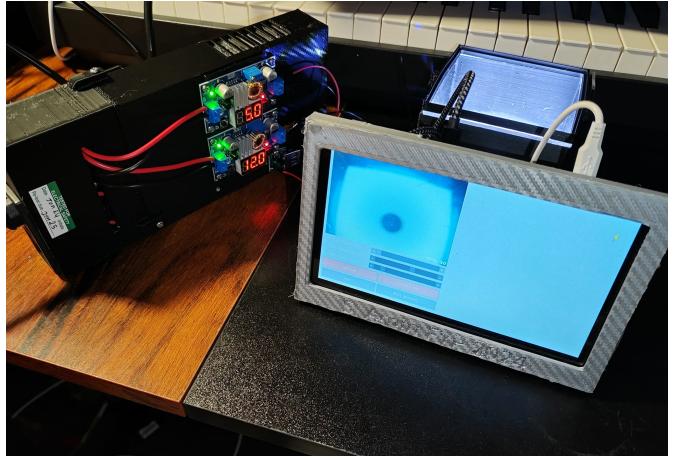


Figure 24: Current System

## 4.3 Stage 3

For the Stage 3 system, the system will become semi-autonomous, with the user having to manually place the components into the system, and the system automatically sorting the components into the correct bins after classifying and identifying the components.

For the Stage 3 system, the following tasks will need to be completed:

- **Sorting Mechanism**

The system will need to be able to sort the components into the correct bins. This will be done using a servo motor to move the components into the correct bin. This not only requires the software to be developed but also the physical mechanism to be designed and built, including the bins themselves. This is a non-trivial task, and so will be the main focus of this stage.

- **User Interface**

The UI of the system will need to be updated to reflect the system's new functionality. This will be done using the Pygame library<sup>31</sup>.

## 4.4 Stage 4

The final stage of the system will be to make the system fully autonomous, with the system being able to identify and sort components without any user input. The main task for this stage will be to develop a device that can automatically feed the components into the system, and as discussed in Section 2 (Background), this will likely be done with a vibratory bowl feeder (VBF).

Ensuring system reliability will also be a key task for this stage, as at this point the system will contain many moving parts, and will need to be

able to operate reliably for extended periods. This stage will mostly involve fine-tuning the system and logistics such as finalising the final report and presentation.

In terms of time management, the decision to enroll in only two modules this term was made after opting to complete four modules in the first term. Substantial work has already been completed during the Christmas break, so there is confidence that sufficient time on the project can be spent during this Spring term.

## 5 Evaluation Plan

To evaluate the performance of the system, several metrics could be used.

For the evaluation of the entire system, one could measure the system's performance in terms of the number of components sorted per minute and compare this to the number of components that a human can sort per minute. While a crude metric, this would provide a good indication of the system's performance in terms of speed. Additionally, the percentage of correctly sorted components could be measured, which would indicate the system's accuracy.

For the evaluation of the computer vision system, one could use a multitude of metrics, for example, the accuracy, precision, recall, and F1 score of the system. These metrics would be calculated by comparing the predictions of the system to the ground truth labels of each component. An important metric would also be throughput, which would be the number of components that the system can process per minute. This metric would be important as it would allow us to determine whether the system is fast enough to be used in a real-world setting, in other words, it would be the limiting factor for the speed of the system.

Alternatively, as the system aims to be a replacement for the current manual sorting process, the system could be evaluated in terms of the time saved by using the system compared to the time taken to manually sort the components. This would be a more practical metric, as it would allow us to determine whether the system is a viable replacement for the current manual sorting process. Additionally, as the Lab is working towards the LEAF certification, the system could be evaluated in terms of the amount of waste it saves compared to just throwing the components away. In the same vein, the system could be

evaluated in terms of the value of the components it saves and the cost of labour it saves.

For the evaluation of the sorting mechanism, the main metric would be the accuracy of the system, which would be the number of components that are sorted into the correct bin.

## 6 Ethical, Legal, and Safety Plan

While the system has no legal or ethical considerations at this stage, it is important to consider the safety of the system, as it will be used in the Level 1 Labs. As the system is mechanical and electrical, many safety considerations need to be taken into account.

### 6.1 Electrical Safety

As mentioned in Section 3.1.1 (Hardware), the system is powered by a 24V, 6.25A, 150W power supply, and must be stepped down to 5V for the Pi and 12V for the LED strip. This is achieved using an XL4015 Step Down Converter<sup>19</sup>, a variable step down converter that can output up to 75W. Due to the potential of high-current carrying wires, the PSU has been enclosed in a 3D printed case, which prevents accidental contact with the PSU's terminals.

A standard IEC C14 power connector and an RS Pro Snap-In Fused Rocker Switch<sup>6</sup> are used to connect the PSU to the mains power. As the power supply is a 6.25A power supply, a 6A fuse was chosen for the switch, which is the closest available fuse rating to the power supply's current rating. This ensures that the fuse will blow before the power supply is damaged, should there be a short circuit in the system.

For wiring the power supply and connecting the step-down converters, 18AWG wire was chosen which is rated for 17A<sup>5</sup>, more than enough for the system's power consumption. The wire is securely crimped to the power supply using a crimping tool.

In addition, my power supply has been PAT tested by technicians in the Level 1 Labs and has been deemed safe for use. PAT (Portable Appliance Testing) is a process by which electrical appliances are routinely checked for safety<sup>427</sup>. The following table describes the PAT testing results for my power supply as well as a description of the tests performed.

As shown in Table 1 and Figure 25, the power supply has passed all PAT tests and is safe for use.



Figure 25: PAT Testing Machine

Test	Result	Req	Description
Earth Continuity	0.06Ω	≤ 0.1Ω	Verifies the earth wire is connected.
Earth Leakage	0.2mA	≤ 0.75mA	Ensure the touchable metal parts cannot cause harm.
Insulation Resistance	320MΩ	≥ 1MΩ	Ensure wires are sufficiently insulated.

Table 1: PAT Testing Results

## 6.2 Mechanical Safety

Currently, the system does not contain any moving parts, and as such, there are no mechanical safety considerations that need to be taken into account for this stage. However, as mentioned in Section 4 (Project Plan) Stage 2 requires the use of a conveyor belt system, and Stage 4 requires the use of a vibratory bowl feeder. In this stage, mechanical safety will be a key consideration and will be discussed in the relevant sections in the final report.

## 7 References

- [1] Maria Concepcion Abe, Gabriel Angelo Gelladuga, Chirstine Joy Mendoza, Jisseth Mae Natavio, Jeanella Shaine Zabala, and Edgar Clyde R. Lopez. Pneumatic conveying technology: Recent advances and future outlook. *Engineering Proceedings*, 56(1), 2023. ISSN 2673-4591. doi: 10.3390/ASEC2023-16267. URL <https://www.mdpi.com/2673-4591/56/1/205>.
- [2] Alasdair Allan. Benchmarking the intel neural compute stick on the new raspberry pi 4, model b. <https://aallan.medium.com/benchmarking-the-intel-neural-compute-stick-on-the-new-raspberry>
- [3] Praneel Chand and Sunil Lal. Vision-based detection and classification of used electronic parts. *Sensors*, 22(23), 2022. ISSN 1424-8220. doi: 10.3390/s22239079. URL <https://www.mdpi.com/1424-8220/22/23/9079>.
- [4] Visual Studio Code. Visual studio code. <https://code.visualstudio.com/>, n.d. Accessed: 06-02-2024.
- [5] RS Components. Rs pro 18 awg hook up wire, pvc insulation. <https://docs.rs-online.com/770d/A700000007035534.pdf>, n.d. Accessed: 22-01-2024.
- [6] RS Components. Rs pro c14 snap-in iec connector. <https://docs.rs-online.com/c6ad/0900766b815867b2.pdf>, n.d. Accessed: 22-01-2024.
- [7] PAT Testing Course. Pat testing course. <https://www.pat-testing-course.com/>, n.d. Accessed: 28-01-2024.
- [8] DFRobot. 7 inch hdmi display with usb touchscreen. <https://www.farnell.com/datasheets/3162025.pdf>, n.d. Accessed: 19-01-2024.
- [9] Amol I. Dhenge, Nāgpur, and A. S. Khobragade. Mechanical nut-bolt sorting using principle component analysis and artificial neural network. In *n.a*, 2013. URL <https://api.semanticscholar.org/CorpusID:201742258>.
- [10] DietPi. Dietpi. <https://dietpi.com/>, n.d. Accessed: 28-01-2024.

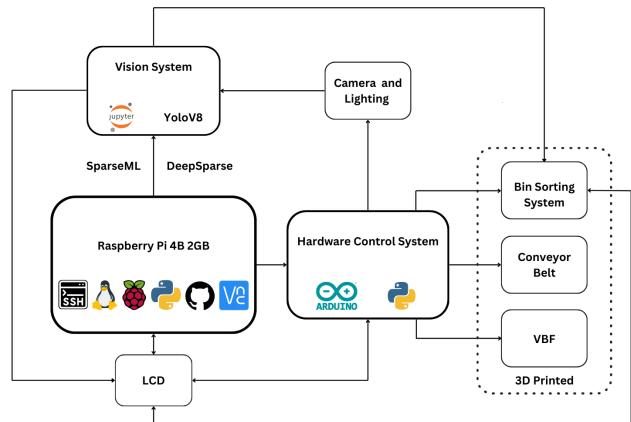


Figure 26: Stage 1 System Diagram

-pi-4-model-b-e419393f2f97, 08-2019.  
Accessed: 06-02-2024.

- [3] Praneel Chand and Sunil Lal. Vision-based detection and classification of used electronic parts. *Sensors*, 22(23), 2022. ISSN 1424-8220. doi: 10.3390/s22239079. URL <https://www.mdpi.com/1424-8220/22/23/9079>.
- [4] Visual Studio Code. Visual studio code. <https://code.visualstudio.com/>, n.d. Accessed: 22-01-2024.
- [5] RS Components. Rs pro 18 awg hook up wire, pvc insulation. <https://docs.rs-online.com/770d/A700000007035534.pdf>, n.d. Accessed: 22-01-2024.
- [6] RS Components. Rs pro c14 snap-in iec connector. <https://docs.rs-online.com/c6ad/0900766b815867b2.pdf>, n.d. Accessed: 22-01-2024.
- [7] PAT Testing Course. Pat testing course. <https://www.pat-testing-course.com/>, n.d. Accessed: 28-01-2024.
- [8] DFRobot. 7 inch hdmi display with usb touchscreen. <https://www.farnell.com/datasheets/3162025.pdf>, n.d. Accessed: 19-01-2024.
- [9] Amol I. Dhenge, Nāgpur, and A. S. Khobragade. Mechanical nut-bolt sorting using principle component analysis and artificial neural network. In *n.a*, 2013. URL <https://api.semanticscholar.org/CorpusID:201742258>.
- [10] DietPi. Dietpi. <https://dietpi.com/>, n.d. Accessed: 28-01-2024.

- [11] Raspberry Pi Foundation. Raspberry pi 4 model b. <https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/>, n.d. Accessed: 19-01-2024.
- [12] FreeCAD. Freecad. <https://www.freecadweb.org/>, n.d. Accessed: 19-01-2024.
- [13] Fritzing. Fritzing. <https://fritzing.org/>, n.d. Accessed: 22-01-2024.
- [14] Git. Git. <https://git-scm.com/>, n.d. Accessed: 22-01-2024.
- [15] GitHub. Github. <https://github.com/>, n.d. Accessed: 22-01-2024.
- [16] Ce Guo, Xiao ling Lv, Yan Zhang, and Ming lu Zhang. Improved yolov4-tiny network for real-time electronic component detection. *Scientific Reports*, 11(1):22744, 2021. doi: 10.1038/s41598-021-02225-y. URL <https://doi.org/10.1038/s41598-021-02225-y>.
- [17] Infineon. Irlz44n datasheet. [https://www.infineon.com/dgdl/Infineon-IRLZ44N-DataSheet-v01\\_01-EN.pdf?fileId=546d462533600a40153567217c32725](https://www.infineon.com/dgdl/Infineon-IRLZ44N-DataSheet-v01_01-EN.pdf?fileId=546d462533600a40153567217c32725), n.d. Accessed: 22-01-2024.
- [18] Jupyter. Jupyter. <https://jupyter.org/>, n.d. Accessed: 25-01-2024.
- [19] Katranji. Xl4015 datasheet. <https://www.katranji.com/tocimages/files/457120-274359.pdf>, n.d. Accessed: 22-01-2024.
- [20] Sigitas Kiliukevičius and Algimantas Fedaravičius. Vibrational transportation on a platform subjected to sinusoidal displacement cycles employing dry friction control. *Sensors*, 21(21), 2021. ISSN 1424-8220. doi: 10.3390/s21217280. URL <https://www.mdpi.com/1424-8220/21/21/7280>.
- [21] Neural Magic. Deepsparse. <https://github.com/neuralmagic/deepsparse>, n.d.
- [22] Neural Magic. Yolov8 detection 10x faster with deepsparse—over 500 fps on a cpu. <https://neuralmagic.com/blog/yolov8-detection-10x-faster-with-deepsparse-500-fps-on-a-cpu/>, n.d.
- [23] Neural Magic. Sparseml. <https://github.com/neuralmagic/sparseml>, n.d.
- [24] Mia Muminovic and Emir Sokic. Automatic segmentation and classification of resistors in digital images. In *2019 XXVII International Conference on Information, Communication and Automation Technologies (ICAT)*, pages 1–6, 2019. doi: 10.1109/ICAT47117.2019.8939034.
- [25] MyreMylar. Pygame gui. [https://github.com/MyreMylar/pygame\\_gui/tree/0cbf7056518377b455d51a8d20167f4029756ad9](https://github.com/MyreMylar/pygame_gui/tree/0cbf7056518377b455d51a8d20167f4029756ad9), n.d. Accessed: 22-01-2024.
- [26] Le Nam, Nguyen Mui, and Dang Tu. A method to design vibratory bowl feeder by using fem modal analysis. *Vietnam Journal of Science and Technology*, 57:102, 02 2019. doi: 10.15625/2525-2518/57/1/12859.
- [27] NVIDIA. Jetson nano. <https://developer.nvidia.com/embedded/jetson-nano-developer-kit>, n.d. Accessed: 06-02-2024.
- [28] University of Bristol. Leaf green lab certification. <https://www.imperial.ac.uk/sustainable-imperial/resource-management/energy-use/laboratory-efficiency-assessment-framework-leaf/#:~:text=In%202021%2D22%2C%20we%20awarded,leaving%20procedures%20were%20in%20place>, n.d. Accessed: 06-02-2024.
- [29] Okdo. Okdo, camera module, csi-2 with 2592 x 1944 resolution. <https://uk.rs-online.com/web/p/raspberry-pi-cameras/2020456/>, n.d. Accessed: 19-01-2024.
- [30] Okdo. Okdo, camera module, csi-2 with 2592 x 1944 resolution specification. <https://docs.rs-online.com/b064/A70000006917308.pdf>, n.d. Accessed: 19-01-2024.
- [31] Pygame. Pygame documentation. <https://www.pygame.org/docs/>, n.d. Accessed: 22-01-2024.
- [32] Pylint. Pylint. <https://github.com/pylint-dev/pylint>, n.d. Accessed: 19-01-2024.
- [33] Python. Python. <https://www.python.org/>, n.d. Accessed: 19-01-2024.
- [34] Erwin Quilloy, C. Delfin, and M. Pepito. Single-line automated sorter using mechatronics and machine vision system for philippine table eggs. *African Journal of*

- Agricultural Research*, 13:918–926, 04 2018. doi: 10.5897/AJAR2018.13113.
- [35] RealVNC. Realvnc. <https://www.realvnc.com/en/>, n.d. Accessed: 22-01-2024.
- [36] Gunther Reinhart and Michael Loy. Design of a modular feeder for optimal operating performance. *CIRP Journal of Manufacturing Science and Technology*, 3(3):191–195, 2010. ISSN 1755-5817. doi: <https://doi.org/10.1016/j.cirpj.2010.09.003>. URL <https://www.sciencedirect.com/science/article/pii/S1755581710000908>.
- [37] Tom Schimansky. Custom tkinter widgets. <https://github.com/TomSchimansky/CustoMtkinter>, n.d. Accessed: 25-01-2024.
- [38] Richard Silversides, Jian S Dai, and Lakmal Seneviratne. Force Analysis of a Vibratory Bowl Feeder for Automatic Assembly. *Journal of Mechanical Design*, 127(4):637–645, 08 2004. ISSN 1050-0472. doi: 10.1115/1.1897407. URL <https://doi.org/10.1115/1.1897407>.
- [39] Pertiwang Sismananda, Maman Abdurohman, and Aji Gautama Putrada. Performance comparison of yolo-lite and yolov3 using raspberry pi and motioneyeos. In *2020 8th International Conference on Information and Communication Technology (ICoICT)*, pages 1–7, 2020. doi: 10.1109/ICoICT49345.2020.9166199.
- [40] Juan Terven and Diana-Margarita Cordova-Esparza. A comprehensive review of yolo: From yolov1 to yolov8 and beyond. *n.j*, 04 2023.
- [41] Tkinter. Tkinter documentation. <https://docs.python.org/3/library/tkinter.html>, n.d. Accessed: 22-01-2024.
- [42] Gov UK. Pat testing: portable appliance testing. <https://www.hse.gov.uk/electricity/faq-portable-appliance-testing.htm>, n.d. Accessed: 27-01-2024.
- [43] Feeder University. Vibratory feeder basics. <https://www.youtube.com/watch?v=m27oD1wfQ0Y>, n.d. Accessed: 06-02-2024.
- [44] Yuanyuan Xu, Genke Yang, Jiliang Luo, Jianan He, and Chenxi Huang. An electronic component recognition algorithm based on deep learning with a faster squeezenet. *Mathematical Problems in Engineering*, 2020: 2940286, 2020. doi: 10.1155/2020/2940286. URL <https://doi.org/10.1155/2020/2940286>.
- [45] Xiangli Yang, Zixing Song, Irwin King, and Zenglin Xu. A survey on deep semi-supervised learning. *IEEE Transactions on Knowledge and Data Engineering*, 35(9): 8934–8954, September 2023. ISSN 2326-3865. doi: 10.1109/TKDE.2022.3220219. URL <http://dx.doi.org/10.1109/TKDE.2022.3220219>.
- [46] Department of Mechanical Engineering Zhengyang Zhang, MIT. *Design and Development of an Automated Sorting and Orienting Machine for Vials*. Massachusetts Institute of Technology, Department of Mechanical Engineering, 2019. URL <https://books.google.co.uk/books?id=Mi5WzQEACAAJ>.

## 8 Appendix

### Bill of Materials

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
Category	Group	Usage	Component	Link	Price	Component Desc	Dependencies	Notes	Bought Batch 1	Bought Batch 2	Bought Batch 3	Brought Myself		Totals	Batch 1:	240.51	
1	Processing	Processors	Provides the system with														
3			Jetson Nano	p/processor-dev	134	CSI camera interf											
4			raspberry Pi 4B 4Cpu4-moddp4gb/r	45		ports, GPIO, CSI i	RPI	a 2GB model inst	45.46							Myself:	8.2
5			Io ROCK 4 Model158?intcmp=UK-	45		ith all the same in											325.25
6	GPU Processor	Separate modules that p															
7			Neural Compute	p/processor-dev	59	erencing capabili	RPI										
8	Vision	CSI Camera	Camera used for the visi														
9			Camera Adjustab171ab5e10-0&pd	6		s camera, allows	CAM	www.instructables.c									
10			IV5847 AdjustableWebp/raspberry	9.83		AA	CAM		9.83								
11			7LQR Adjustable	45		AA	CAM										
12			bon Extension Comm/cable-for-p-c	1.20		300mm cable			1.19				2.20				
13			SD Card	32g10-sec/micro	8.30	2GB UHS-I U1 V1			8.30								
14			LED Ring	15f91f5-05pdp_r	3.40	LED Ring for illu							6.00				
15	Components	PSU	Power Supply Unit														
16			IV Switching Powerp/switching-pow	20		150W			19.38								
17			14 IE200 Connect.com/web/p/iec-c	3.71		2pc				7.42							
18			Socket to Type Gom/web/p/power-c	4.46		1.8m				4.46							
19			s Cartridge Fuse, m/web/p/cartridge	0.333		10pc				3.33							
20	Cables	Connect Components															
21			4 Way USB Hub b3-hb-4pm-v2/hub	8.99		3pc			8.99								
22			SB A to USB C 3/aid-usb2-0-a-c-m	6.42		3pc			6.42								
23			SB A to Micro B 3/usb2-0-a-male-m	5.94		3pc			5.94								
24			USB C to C 2-0-type-c-type-c-	3.61		1pc			3.61								
25			18 awg red .om/web/p/hook-u	10.27		25m			10.27								
26			18 awg black .om/web/p/hook-u	10.27		25m			10.27								
27			ribbon cables 28 autsy10ac85/ribbon	4.82		6m				4.82							
28			Wire stripper /d03004/stripper-	2.15		1pc			2.15								
29			icro HDMI to HDMIom/web/p/hdmi-c	5.56		2m			5.56								
30			Heatshrink link-sleeving-kit-	8.45		150pc			8.45								
31	Bands	Allow prototyping by plac															
32																	

Figure 27: Bill of Materials

## FreeCAD System Design

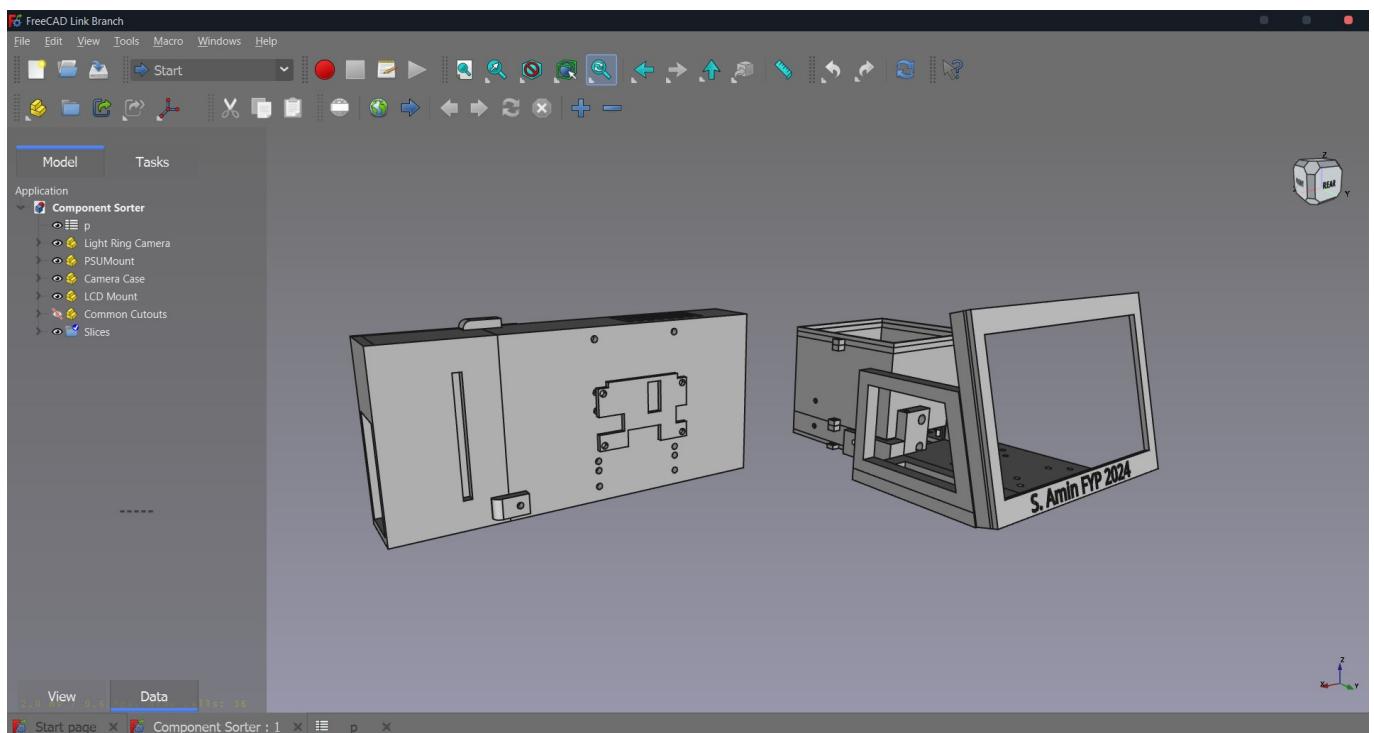


Figure 28: Stage 1 FreeCAD System Design