NESA:
39381729

Autonomous Vigilance

Raspberry Pi Zero-Powered,

User-Defined Surveillance System with Advanced Night Vision Capabilities

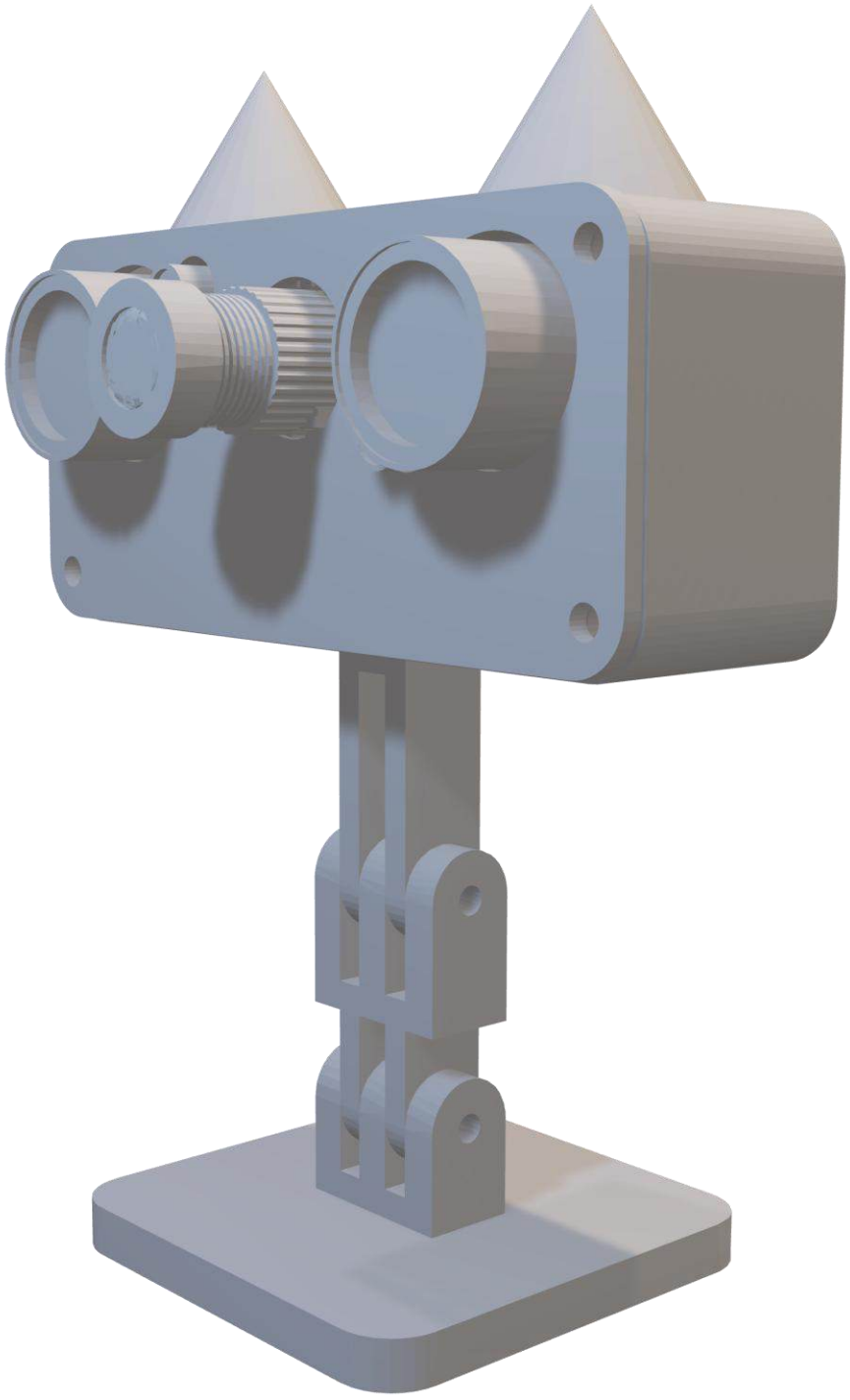


Table of Contents

Introduction

Project Overview:

Autonomous Vigilance is a customisable, cost-effective surveillance system built using a Raspberry Pi Zero and a night vision camera module. Designed for versatility, the system can be adapted for various applications.

Key features include:

- **Motion detection** for automated recording.
- **Infrared night vision** for low-light environments.
- **User customisation** for flexible development.
- **Remote access** via secure web interface.
- **Enhanced security measures**, including local storage and encryption, to protect stored footage.

This project combines Hardware and software development to create a privacy-focused, affordable alternative to commercial surveillance systems, addressing real-world issues such as high cost, cloud storage vulnerability, and customisation.

Statement of Intent:

As a passionate designer and technologist, I aim to explore the intersection of security, user empowerment, and computer vision through Autonomous Vigilance, a DIY surveillance system built on the Raspberry Pi platform. This project responds to the limitations of commercial surveillance systems, such as high cost, limited customisation, and privacy concerns due to cloud dependency, by offering an affordable, adaptable, and user-controlled alternative.

By leveraging the Raspberry Pi, I intend to develop an efficient and intelligent home security system that meets the needs of diverse end-users. This project enables me to apply and expand my knowledge in embedded systems, cybersecurity, and product design, while addressing real-world challenges in security and automation. It also demonstrates the potential of AI-powered surveillance to improve accessibility and affordability.

Beyond technical objectives, Autonomous Vigilance is a platform for continuous learning and innovation. It strengthens my understanding of computer vision, machine learning, and hardware-software integration, while contributing to advancements in user-centred, DIY security solutions. Additionally, this project serves as a portfolio piece, showcasing my ability to deliver practical, impactful technology that inspires others to engage with responsible design and innovation.

I am committed to realising this project, demonstrating how thoughtful design and technology can empower individuals with greater security and control, and contribute to the evolution of intelligent, cost-effective surveillance solutions.

Project Scope:

This project focuses on the design, development, and evaluation of a functional, motion-activated, night vision surveillance system with the following primary features:

- High-quality infrared night vision recording
- Motion-triggered alerts and automatic video storage
- Remote access through a secure website

Advanced features under exploration include facial recognition and AI-powered object detection. While these present technical challenges, a basic version is being developed to demonstrate the system's potential for future enhancement.

Identification and Exploration of the Need

Current commercially available surveillance systems present several challenges for end-users, including high initial and ongoing costs, privacy concerns due to cloud-based storage, and limited options for customisation. These issues can deter homeowners, renters, and small business owners from adopting effective security solutions tailored to their specific needs.

Problem Statement:

Through user research and analysis of existing products, the following key issues were identified:

- **High Costs:** Quality security systems are expensive, often requiring additional monthly fees for cloud storage.
- **Privacy Concerns**: Cloud-based storage exposes sensitive footage to potential hacking and unauthorised access.
- **Limited Customisation:** Pre-built systems lack flexibility, restricting users from adapting the system to unique requirements or environments

Real-World Problem Addressed:

The lack of affordable, privacy-focused, and customisable surveillance solutions leaves many users without effective security options. This project aims to address these shortcomings by developing a system that is:

- Affordable: The total system cost is under $200, making it accessible to a wider audience.
- Privacy-Focused: All video footage is stored locally and encrypted, ensuring user control over data.
- Customisable: Open-source software and modular hardware allow users to adapt the system to their needs

Target Audience:

- **DIY Enthusiasts and Hobbyists**
    - People who enjoy building and tinkering with electronics and software.
    - Individuals interested in learning about Raspberry Pi, Python Programming, and embedded systems.
    - Those who like to customise and personalise their tech.
- **Homeowners and Renters Concerned About Security**

  - Individuals looking for affordable and customizable home security solutions
  - People who prioritise privacy and want control over their surveillance footage.
  - Those who are wary of cloud-based security systems and their potential Vulnerabilities.

- **Tech-savvy Individuals**
  - People who are interested in exploring and implementing AI and machine learning in practical applications.
  - Those who value open-source technology and its flexibility.
  - People who want to learn more about cybersecurity.

- **Students and educators**
  - Students interested in technology projects.
  - Educators looking for hands-on projects to teach embedded systems, programming, and security concepts.
  - People who want to learn about CAD design and 3D printing.

My Solution:

To develop Autonomous Vigilance in alignment with real user needs, I conducted informal interviews and surveys with local homeowners, renters, and small business owners. This user-centred research revealed common frustrations with existing surveillance systems, including frequent false alarms, limited privacy due to cloud storage, difficulties in setup and alert management, and poor night-time visibility.

Requirement	Justification
Reliable Night Vision	Ensures clear footage during nighttime using IR-enabled cameras.
AI-Powered Detection	Distinguishes between people, pets, and vehicles to reduce false alarms.
Local Encrypted Storage	Ensures user control over footage without reliance on cloud servers.
Custom Detection Zones	Allows users to monitor only areas of interest, avoiding unnecessary alerts.
Remote Notifications	Sends alerts to users in real time, increasing reaction speed.
Scalability	Supports future expansion with more cameras or advanced compute modules.
User-Friendly Setup	Simplifies system configuration via intuitive web interfaces.

Reflecting on user feedback and research findings, it became clear that a successful solution must balance advanced technical features with ease of use and robust privacy protections. These insights directly informed the system requirements and design priorities for Autonomous Vigilance, ensuring the final product is both functional and accessible for a diverse range of end-users.

Researching

Areas of Investigation

To develop a robust system, I conducted research in the following areas:

- **Raspberry Pi Models**: Evaluated the Raspberry Pi Zero for its low cost and energy efficiency.
- **Night Vision Cameras**: Selected a camera module with excellent infrared sensitivity and image quality.
- **Motion Detection Technologies**: Compared different methods for detecting movement.
- **AI-Based Image Processing**: Explored facial recognition and object detection algorithms.
- **3D Printing & Enclosure Design**: Researched materials (PLA for indoor use, PETG for outdoor durability).
- **Security Protocols**: Investigated encryption methods to secure video storage and remote access.

Technology Research

The hardware components for Autonomous Vigilance were carefully selected to achieve a balance between optimal performance, affordability, and compatibility with the system's requirements. Below is a detailed breakdown of the research and selection process for each hardware component.

Component	Purpose	Key Considerations	Alternatives	Selection	Justification
Raspberry Pi Zero 2 W	Serves as the core computing unit,	Low power consumption, compact size, cost-	Raspberry Pi 3, 4, 5; Jetson	Raspberry Pi Zero 2 W	Selected for its efficient resource usage,

Component	Purpose	Key Considerations	Alternatives	Selection	Justification
	processing video feeds and running the surveillance system.	effectiveness, and Linux compatibility	Nano; Pi Zero W		strong compatibility with Frigate and Docker, and affordability, making it ideal for compact, cost-effective surveillance applications.
Raspberry Pi 5	Provides an alternative high-performance computing platform for more demanding surveillance tasks.	Enhanced processing power, increased RAM, and advanced connectivity options	Raspberry Pi 3, 4, Jetson Nano	Raspberry Pi 5	Chosen for scenarios requiring greater processing capability, ensuring scalability and future-proofing for advanced features or additional video streams.
Raspberry Pi Night Vision Camera	Captures high-quality infrared video footage for day and night monitoring.	Infrared sensitivity, resolution, and CSI interface compatibility	Arducam Night Vision, Waveshare IR Camera	Waveshare Raspberry Pi Night Vision Camera	Offers superior infrared sensitivity and seamless integration with the Raspberry Pi, ensuring reliable 24/7

Component	Purpose	Key Considerations	Alternatives	Selection	Justification
					monitoring in various lighting conditions.
MicroSD Card (32 GB+)	Provides local storage for the OS, surveillance software, and recorded video footage.	High endurance, fast read/write speeds, sufficient capacity, affordability.	SanDisk Ultra, Samsung EVO Select	SanDisk High Endurance 64GB	Selected for its durability and performance under continuous video recording, ensuring reliable local storage and data integrity for surveillance footage.
USB Power Supply	Delivers consistent power to the Raspberry Pi and peripherals.	Stable voltage, sufficient amperage, reliability, and portability	Official Raspberry Pi Power Supply, Power Bank	Official Raspberry Pi 5V 2.5A Power Supply	Ensures stable and reliable power delivery, preventing undervoltage issues that could disrupt system operation or video recording.
Power Bank	Provides portable power for flexible	Sufficient capacity,	Anker, Xiaomi, Official Raspberry	High-Capacity	Enables mobile operation, allowing the

Component	Purpose	Key Considerations	Alternatives	Selection	Justification
	deployment in various locations.	portability, and reliability	Pi Power Bank	Power Bank	system to function in environments without fixed power sources, thus increasing deployment flexibility.
3D-Printed Case	Protects and houses internal components while allowing custom design features.	Adequate ventilation, material strength, ease of assembly, and customisation	PLA/ABS printed case, acrylic enclosure	Custom 3D-Printed PLA Case	Allows for tailored integration of components, lightweight and durable protection, and effective thermal regulation, all while supporting project-specific design requirements.
3D Printer	Fabricates custom enclosures and mounting brackets for the system.	Printing resolution, material compatibility, build volume, and reliability	Ender-3 V2, Prusa i3 MK3S, Ender-3 V3 SE	Ender-3 V3 SE	Chosen for its precision, reliability, and affordability, ensuring high-quality prints for custom parts and enclosures

Component	Purpose	Key Considerations	Alternatives	Selection	Justification
					essential to the project's physical design.

Software Development

The software components were carefully selected to ensure seamless integration, efficient performance, and robust security. Below is a detailed breakdown of the research and selection process for each software component well-suited for integration into Autonomous Vigilance.

Component	Purpose	Key Considerations	Alternatives	Selection	Justification
Kerberos.io	Provides real-time object streaming via a web browser on the local network	RTP streaming compatibility, privacy-focused local storage, and ease of setup	MotionEye, ZoneMinder	Kerberos.io	Chosen for its lightweight design, open-source nature, and seamless integration with Raspberry Pi. Its browser-based interface ensures accessibility without proprietary software, aligning with the project's focus on user control.

Component	Purpose	Key Considerations	Alternatives	Selection	Justification
MotionEye	Serves as a backup for real-time object streaming and motion detection	Raspberry Pi compatibility, motion-triggered alerts, minimal configuration	MotionEyeOS, Shinobi	MotionEye	Selected for redundancy and reliability. Its motion detection features and straightforward web interface provide a fallback option if primary tools encounter issues.
Frigate	Manages AI-powered object detection and video analysis	ARM compatibility, TensorFlow Lite integration, Docker support	Blue Iris, ZoneMinder	Frigate	Optimised for Raspberry Pi's limited resources, Frigate enables efficient real-time object detection using AI models, ensuring accurate surveillance without high computational overhead.
Docker	Containerises services for isolated, reproducible	ARM architecture support, low resource consumption,	Podman, LXC	Docker	Simplifies deployment of Frigate and other services while maintaining

Component	Purpose	Key Considerations	Alternatives	Selection	Justification
	deployments	and scalability			dependency isolation. Docker's widespread adoption ensures long-term maintainability and ease of updates.
Home Assistant	Centralised monitoring and control via a customizable dashboard (optional integration)	MQTT protocol support, device interoperability, and remote access	OpenHAB, Domoticz	Home Assistant	Integrates with Frigate for unified smart home monitoring. Its modular design allows tailored automation and notifications, enhancing usability for advanced scenarios.
Raspberry Pi OS Lite	Serves as the minimal base operating system for the surveillance system	Headless operation, ARM optimisation, low resource footprint	Ubuntu Server, DietPi	Raspberry Pi OS Lite	Provides a stable, lightweight foundation optimised for Raspberry Pi, ensuring maximum performance for 24/7 operation without

Component	Purpose	Key Considerations	Alternatives	Selection	Justification
					unnecessary bloat.
Visual Studio Code	Facilitates code editing, debugging, and web interface development	Cross-platform support, extension ecosystem, and remote SSH capabilities	PyCharm, Sublime Text	Visual Studio Code	Enables efficient development of both backend logic and web interfaces. Extensions for JavaScript, HTML, and CSS streamline frontend development, critical for building the project's user-facing dashboard.
VS Code Extensions	Enhances workflow for web development, container management, and remote access	Compatibility with web frameworks, Docker integration, and productivity tools	N/A	Live Server, Docker, Remote SSH	Extensions like Live Server enable real-time web interface testing, while Docker and Remote SSH support seamless interaction with Raspberry Pi and containers, accelerating

Component	Purpose	Key Considerations	Alternatives	Selection	Justification
					development cycles.
VNC Viewer	Enables remote desktop access to the Raspberry Pi for GUI-based management	Cross-platform compatibility, security, and minimal latency	TeamViewer, AnyDesk	VNC Viewer	Simplifies troubleshooting and configuration by providing direct access to the Raspberry Pi's desktop environment, reducing reliance on physical peripherals during setup.
Guit hub					

Programming Languages Used

The selection of programming languages for Autonomous Vigilance is guided by the project's need for efficiency, flexibility, and user accessibility. Each language is chosen to fulfil a specific role within the system, ensuring seamless integration between hardware, software, and user interface components. By leveraging a combination of modern, widely supported languages, the project achieves robust performance in motion detection, secure data handling, and intuitive remote access. The table below outlines the primary programming languages used, their purposes, and the justifications for their selection.

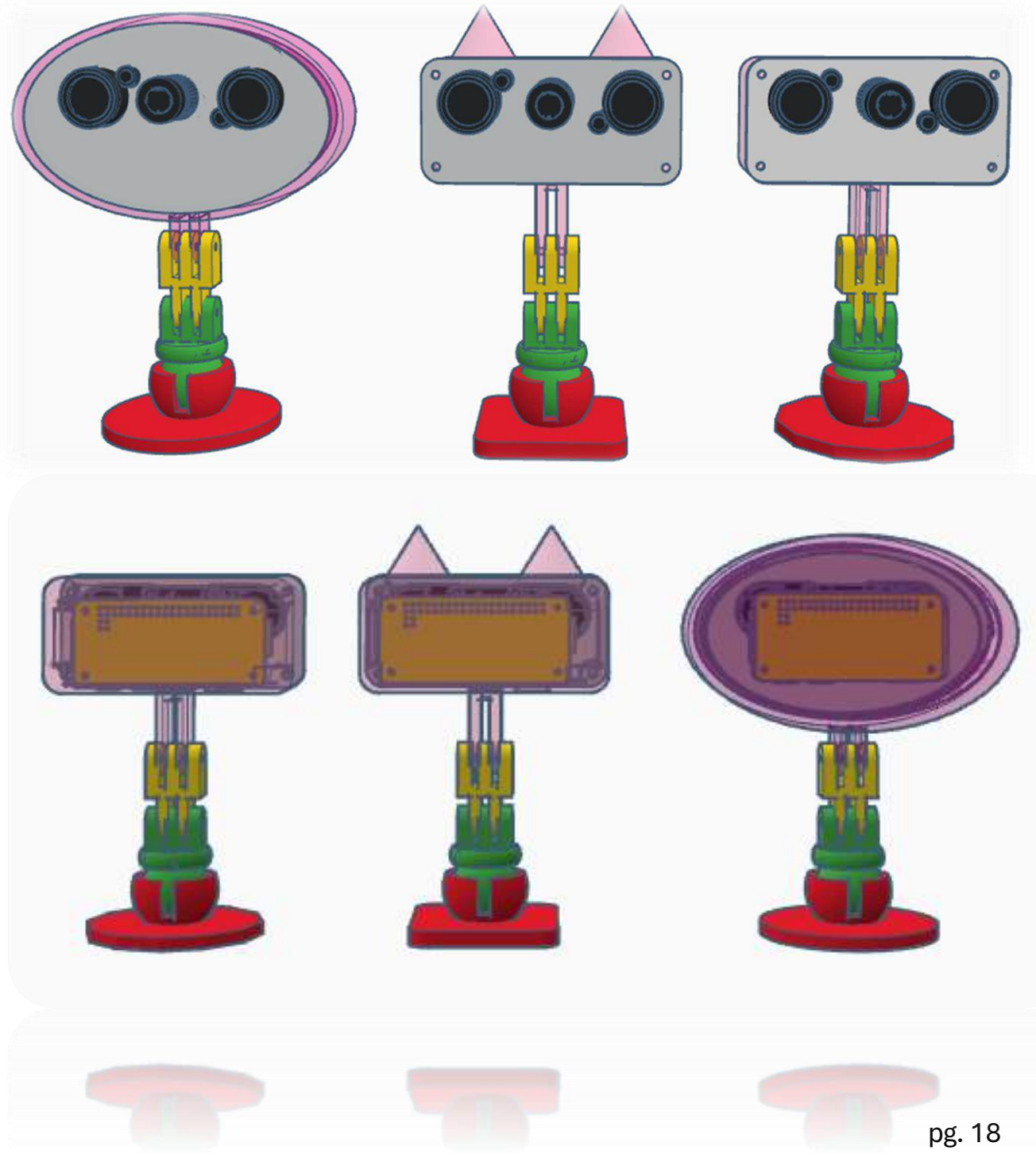
Language	**Purpose/Role in Project**	**Key Features/Benefits**	**Justification for Selection**
Python	Backend development, motion detection, AI integration	Readable syntax, vast libraries (OpenCV, TensorFlow), strong Raspberry Pi support	Ideal for rapid prototyping, AI, and computer vision tasks; widely used in embedded systems
JavaScript	Web interface interactivity and real-time updates	Enables dynamic dashboards, works with HTML/CSS, cross-platform	Essential for building responsive, user-friendly web dashboards accessible from any device
HTML & CSS	Web interface structure and styling	Universal web standards, easy customisation	Ensures the interface is accessible, attractive, and easy to navigate
Bash/Shell	Automation of setup, deployment, and maintenance tasks	Native to Linux, automates repetitive tasks	Streamlines system setup and management on Raspberry Pi, reducing errors and saving time.

Language	**Purpose/Role in Project**	**Key Features/Benefits**	**Justification for Selection**
YAML/JSON	Configuration files for Docker, Frigate, Home Assistant	Human-readable, widely supported for configs	Simplifies customization and integration of services, supporting scalability and user control

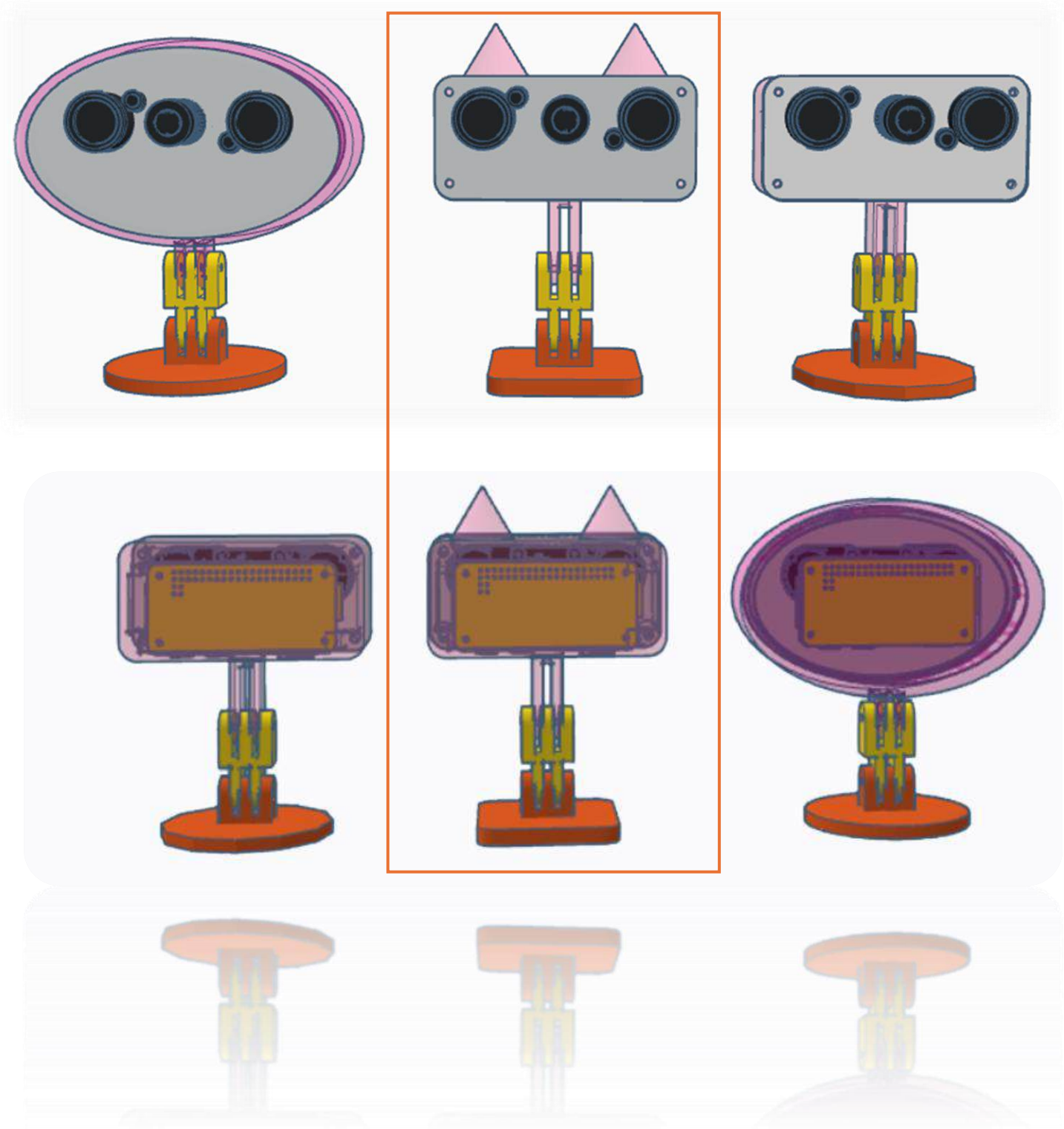
Designs & Brainstorming

Model Blueprint

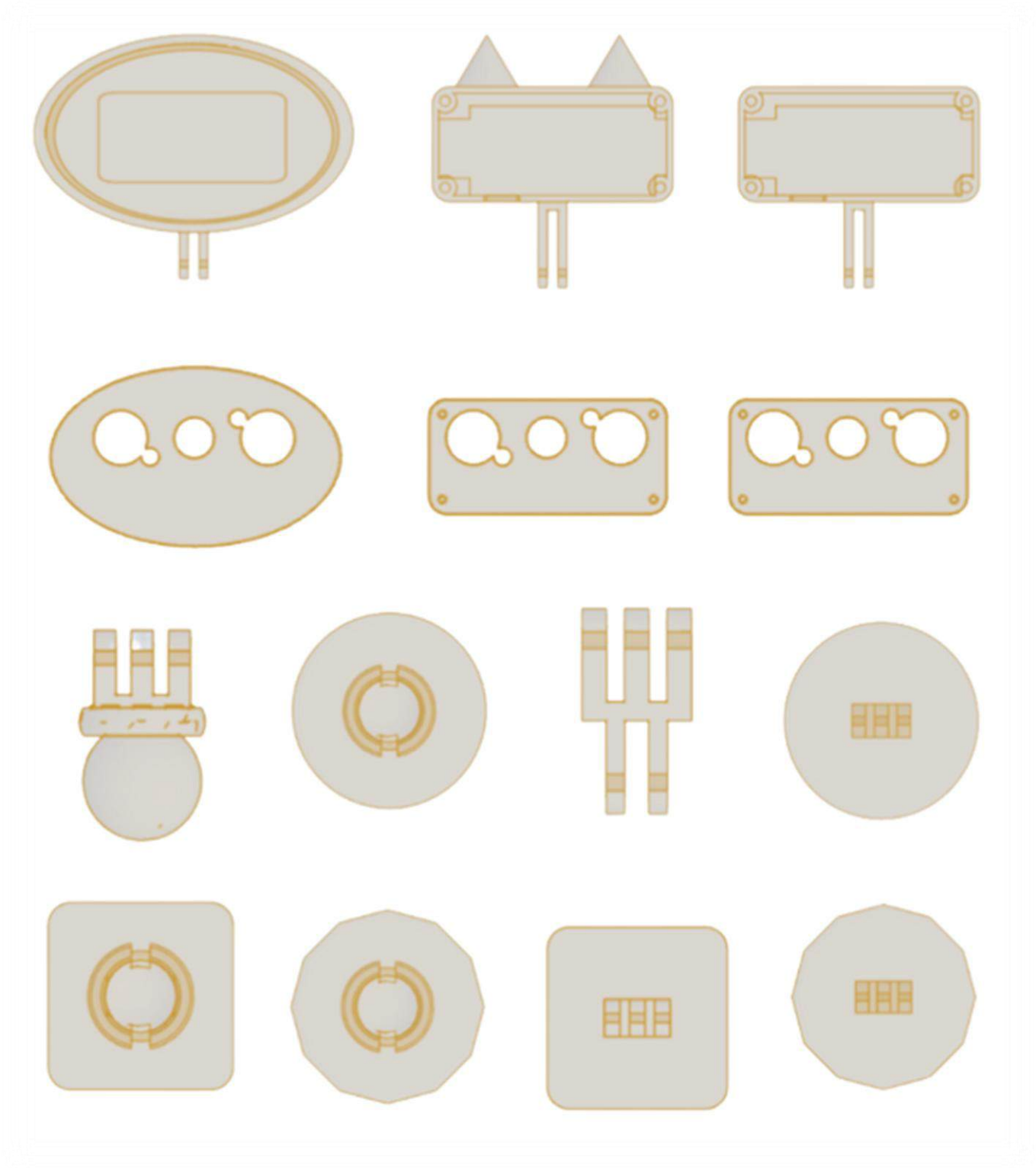
- **Design A: Front and Back**



- **Design B: Front and Back**



- **Two-dimensional view of the main housing components**



- **Selection and justification**

As part of the design development process for the Autonomous Vigilance surveillance system, I prototyped multiple housing concepts to strike a balance between aesthetic appeal and functional adaptability. After iterative testing and peer feedback, Design B – the Cat-Eared Rectangular Frame (as shown in the Model Blueprint) was selected as the final enclosure. A comprehensive evaluation informed this decision of user preference, hardware compatibility, manufacturing efficiency, and potential for future scalability.

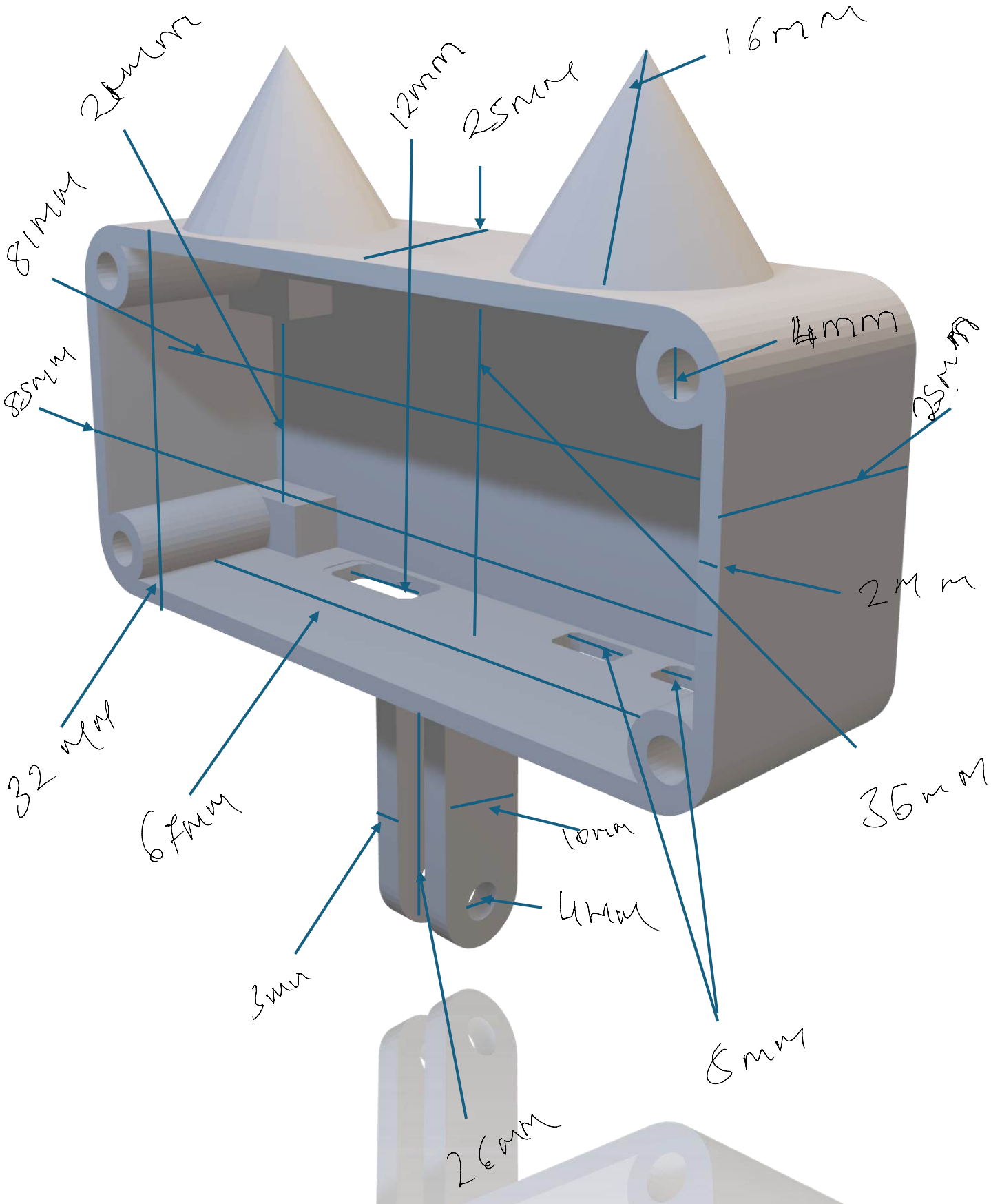
The chosen design features a playful yet professional cat-eared rectangular form, aimed at appealing to a broad audience, from DIY tech enthusiasts to educators and students. It strikes a balance between practical functionality and expressive design, making it suitable for both casual home use and educational or experimental setups. As the designer, I created a range of housing models (including circular, plain rectangular, and cat-eared versions) to provide users with aesthetic choice and contextual adaptability. This user-focused approach encourages engagement and personalization, allowing individuals to select the housing style that best suits their environment or taste.

Functionally, the rectangular base provides an optimal platform for integrating critical internal components, including the Raspberry Pi Zero, NoIR night vision camera, and IR LEDs. The flat, symmetrical interior simplifies component placement, ensures proper airflow, and facilitates efficient cable management. This minimizes the risk of component interference, overheating, or alignment issues, contributing to system reliability and ease of assembly.

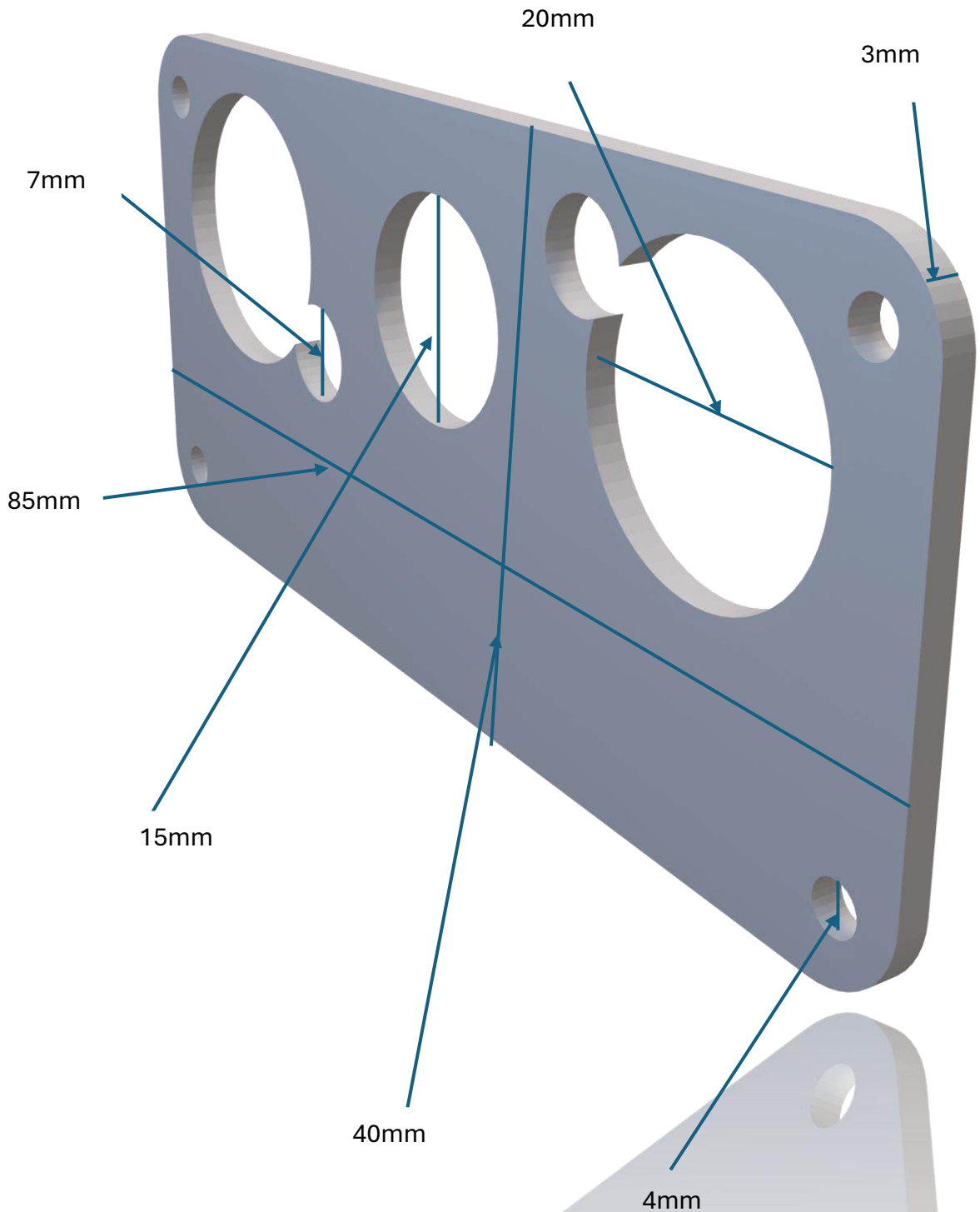
The design was also optimized for efficient 3D printing using PLA and PETG, selected for its durability, UV resistance, and weatherproofing, ideal for both indoor and outdoor installations. The geometric symmetry enhances print quality, while the integrated cat ears add a distinct visual identity without compromising structural integrity. This blend of design flair and engineering stability ensures both visual appeal and robust performance.

Peer feedback, gathered from classmates and teachers, strongly supported the cat-eared model. It was consistently described as “friendly, ” “innovative, ” and “distinctive, ” offering a refreshing alternative to the often bland or overly industrial design of commercial housing. Furthermore, the inclusion of alternate design options reflects real-world product development practices, where providing variation enhances user agency and market adoption.

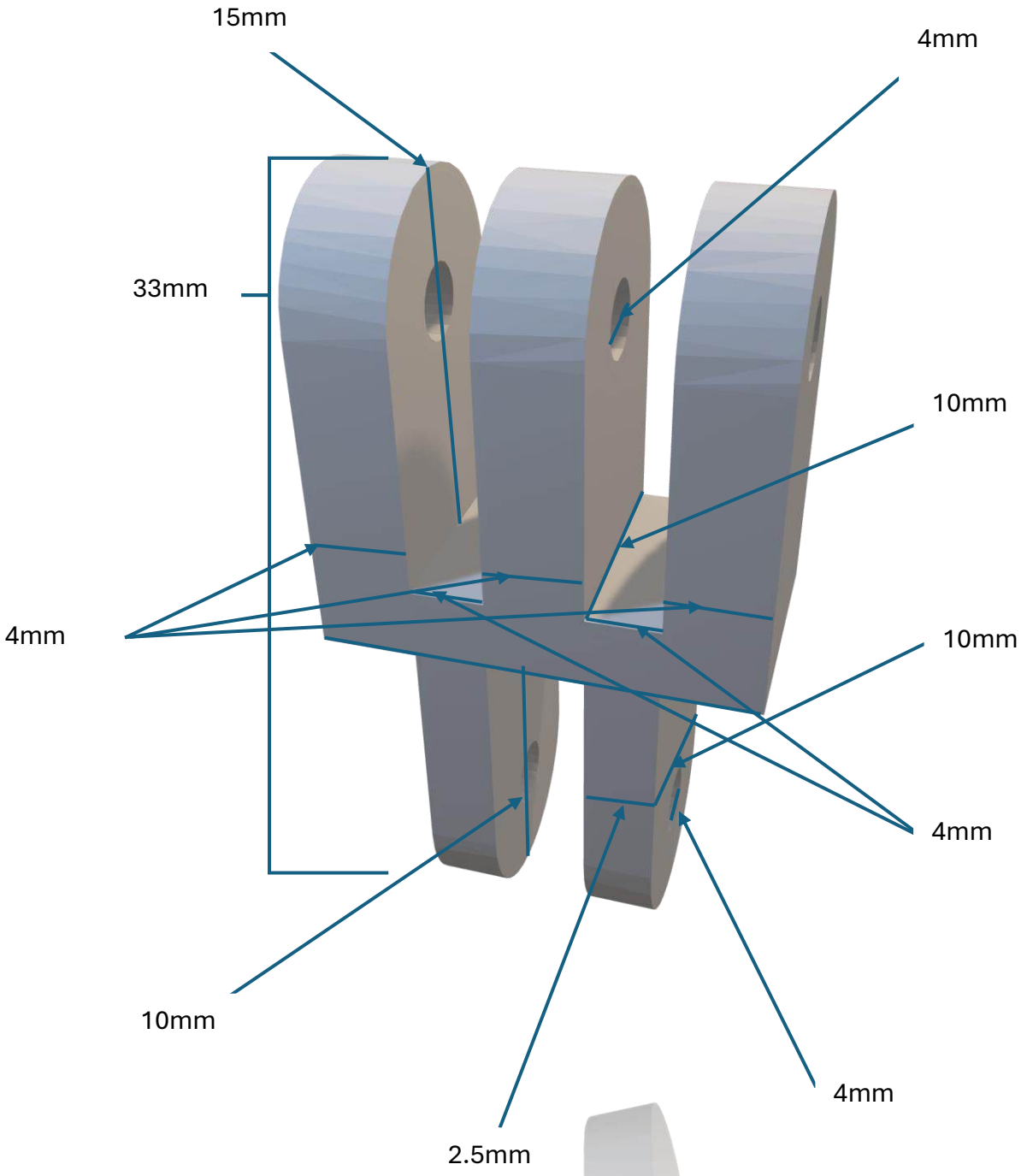
CAMERA HOUSING ENCLOSURE (MAIN BODY):



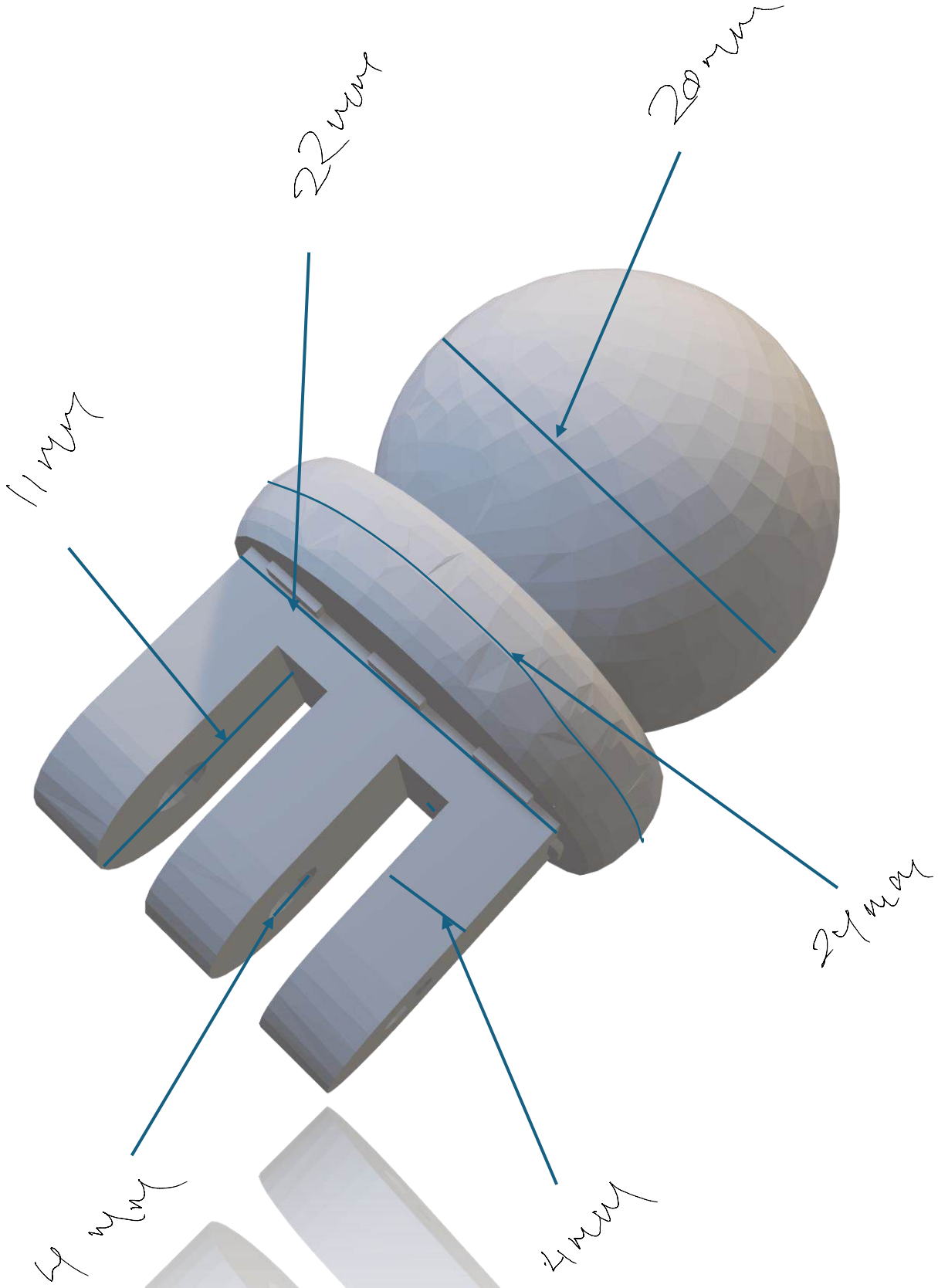
LENS AND SENSOR PORTS (FRONT PLATE):



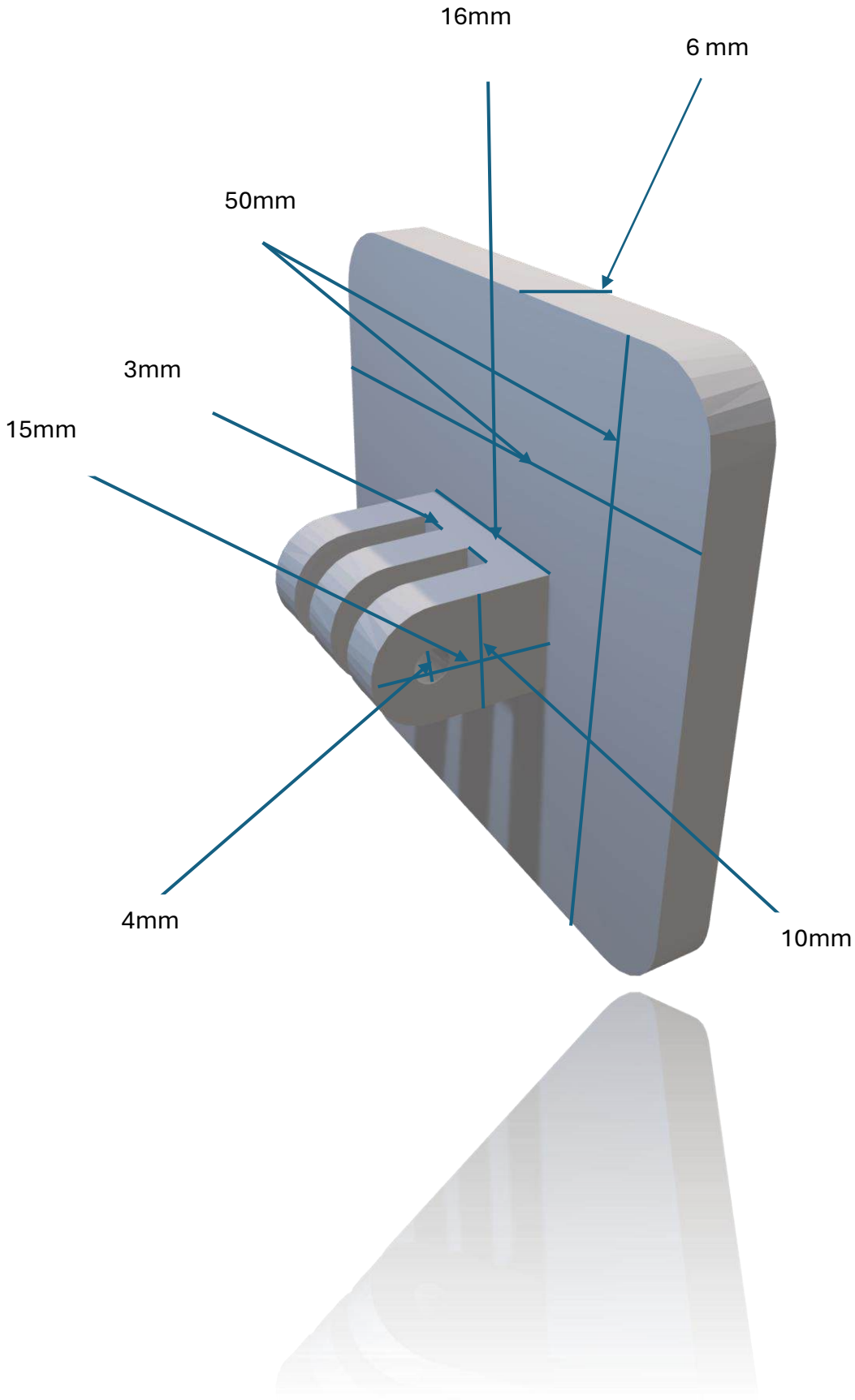
MOUNTING ARM (DOUBLE-HINGED):



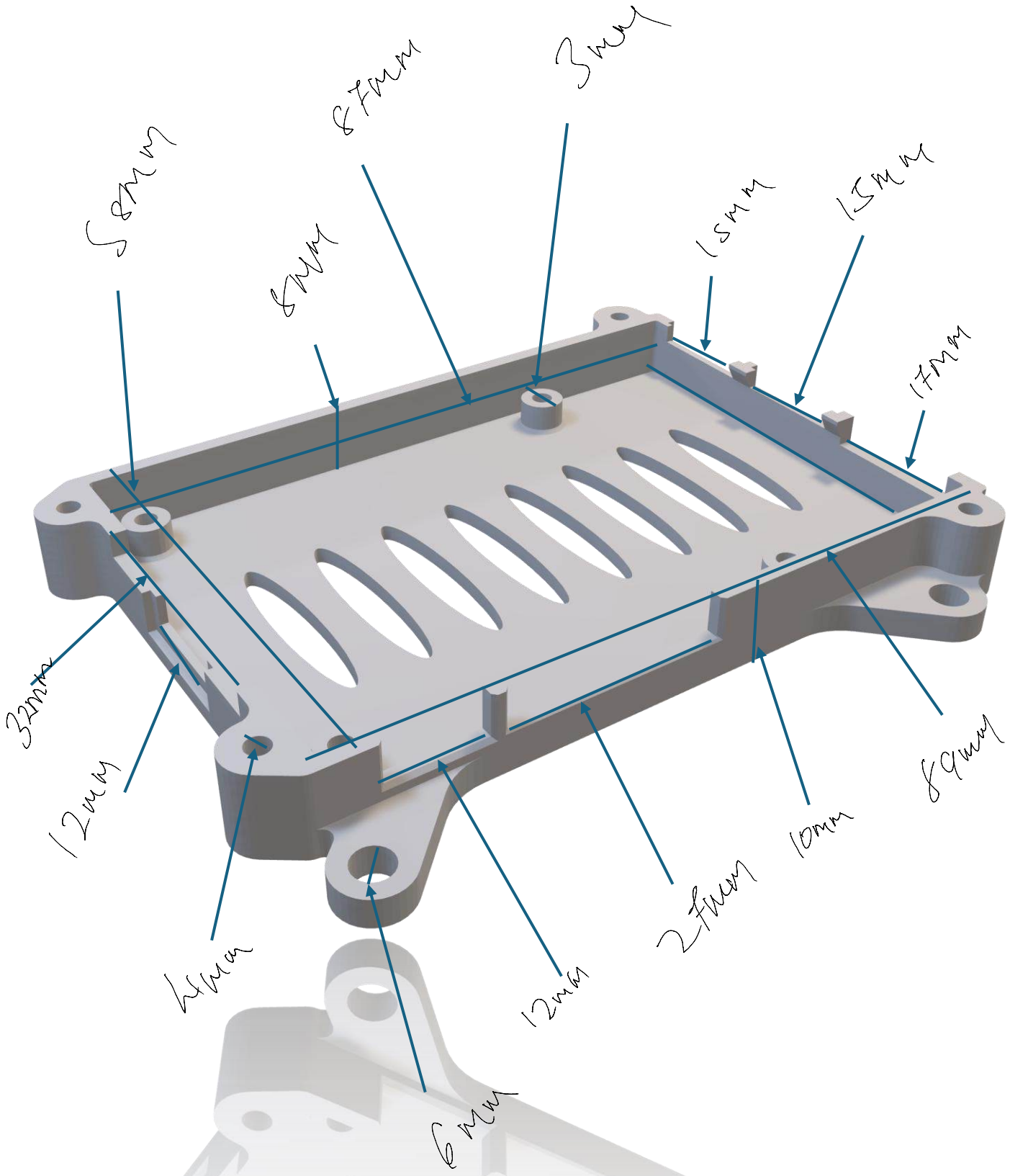
MOUNTED ARM (ROLLING BALL):



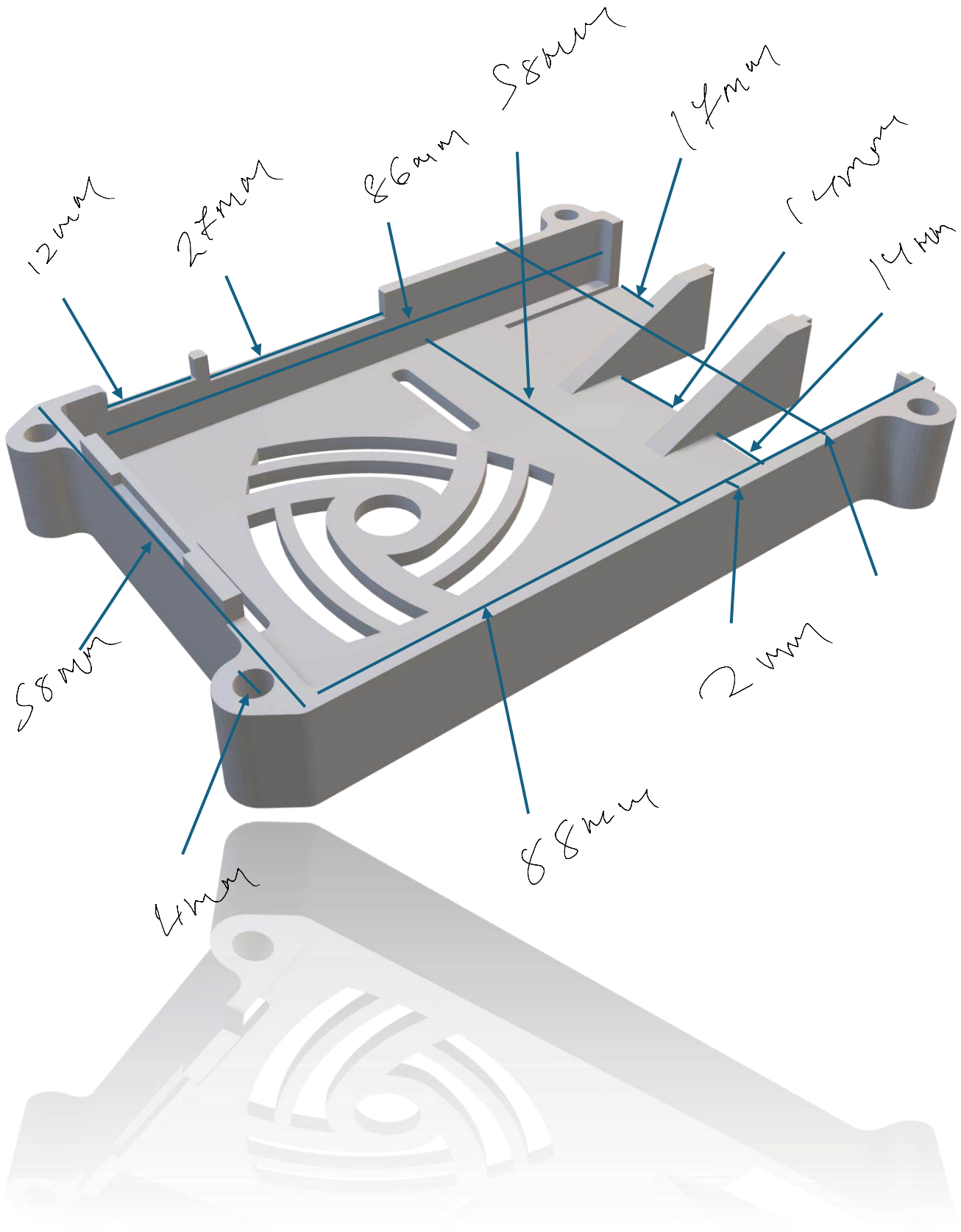
BASE PLATE:



RASPBERRY PI 5 CASE TOP



RASPBERRY PI 5 CASE BOTTOM

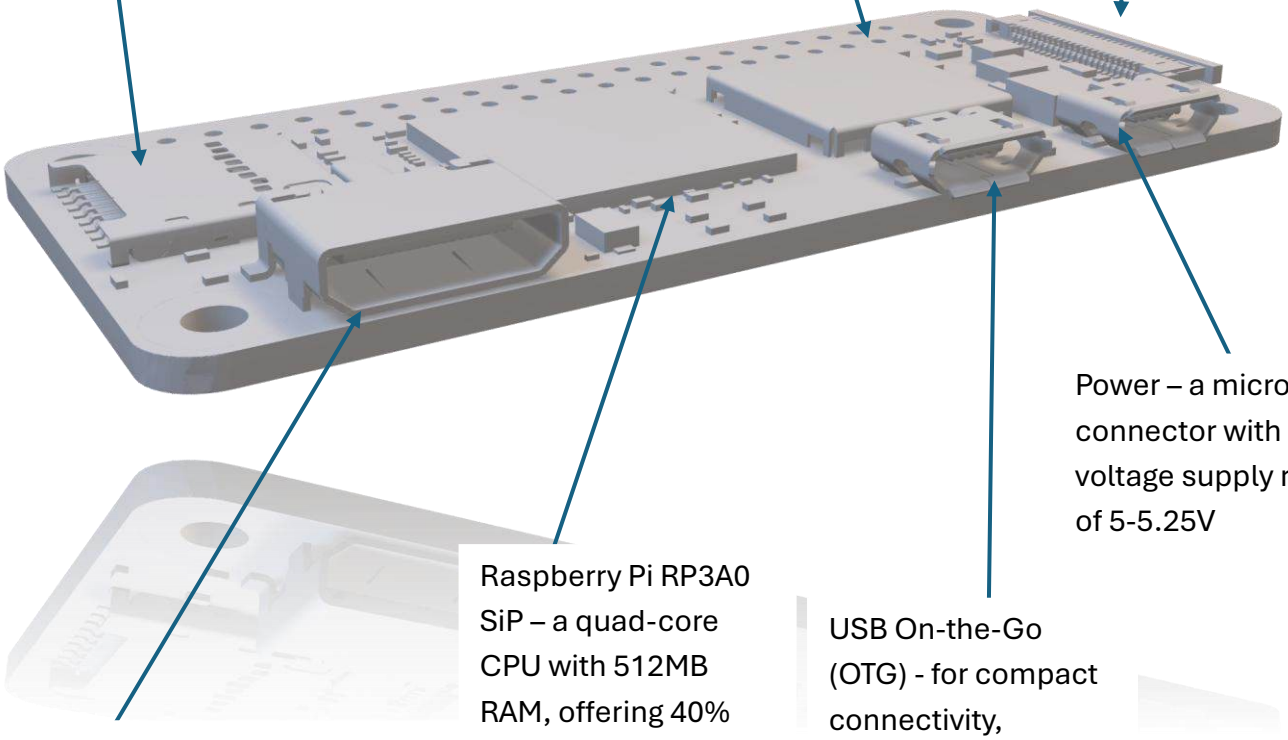


RASPBERRY PI ZERO 2W:

MicroSD Card Slot – the insert port for the microSD card that contains the Raspberry pi images files

CYW43439 – a wireless communication module integrated with 802.11b/g/n Wi-Fi and Bluetooth 4.2 BLE

Cammera Connector – used to attach the raspberry pi camera modules. Having 22pins o.5mm



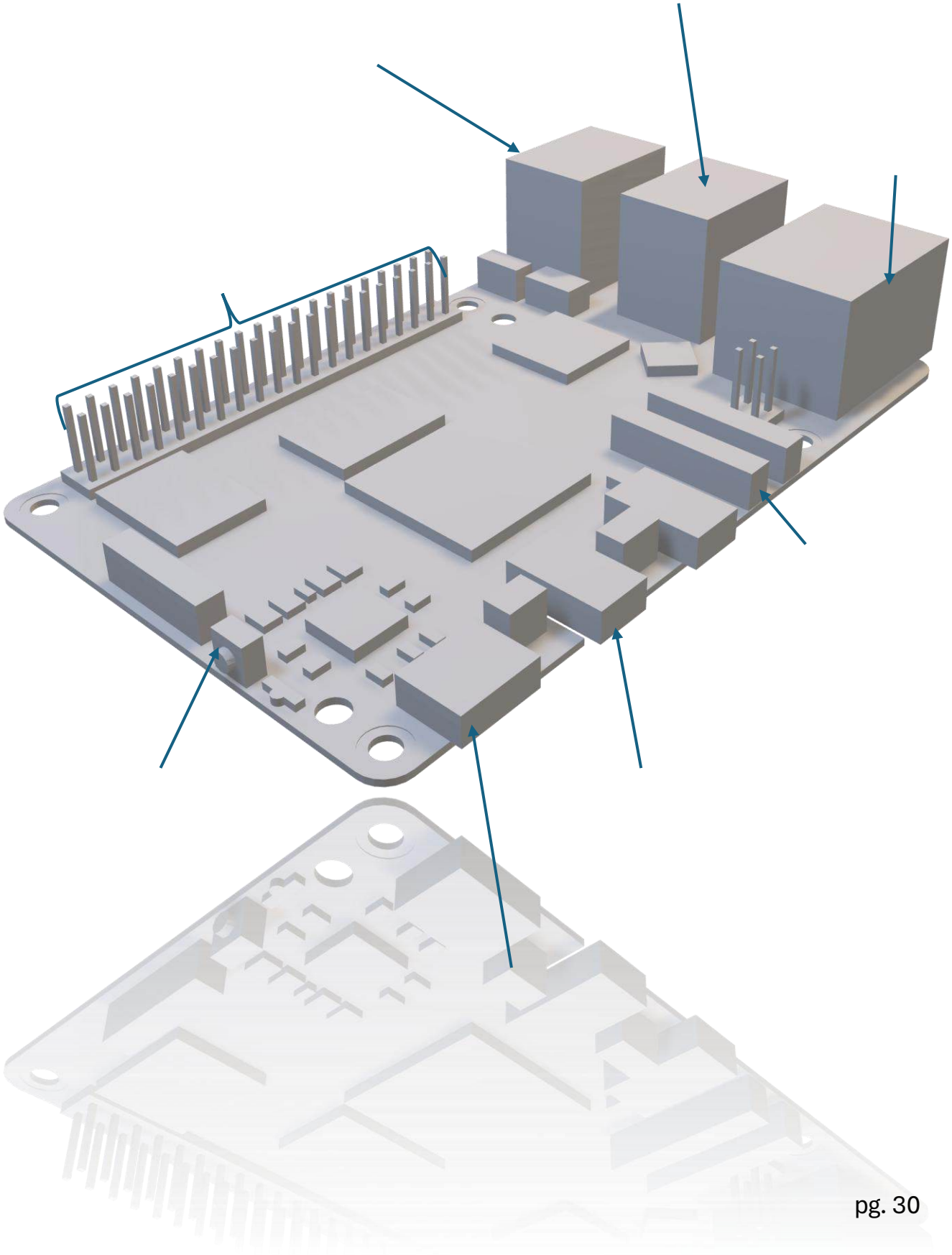
Power – a micro-USB connector with a voltage supply range of 5-5.25V

Mini HDMI – a standard HDMI connecter aiding in connecting the pi zero to a monitor via wire

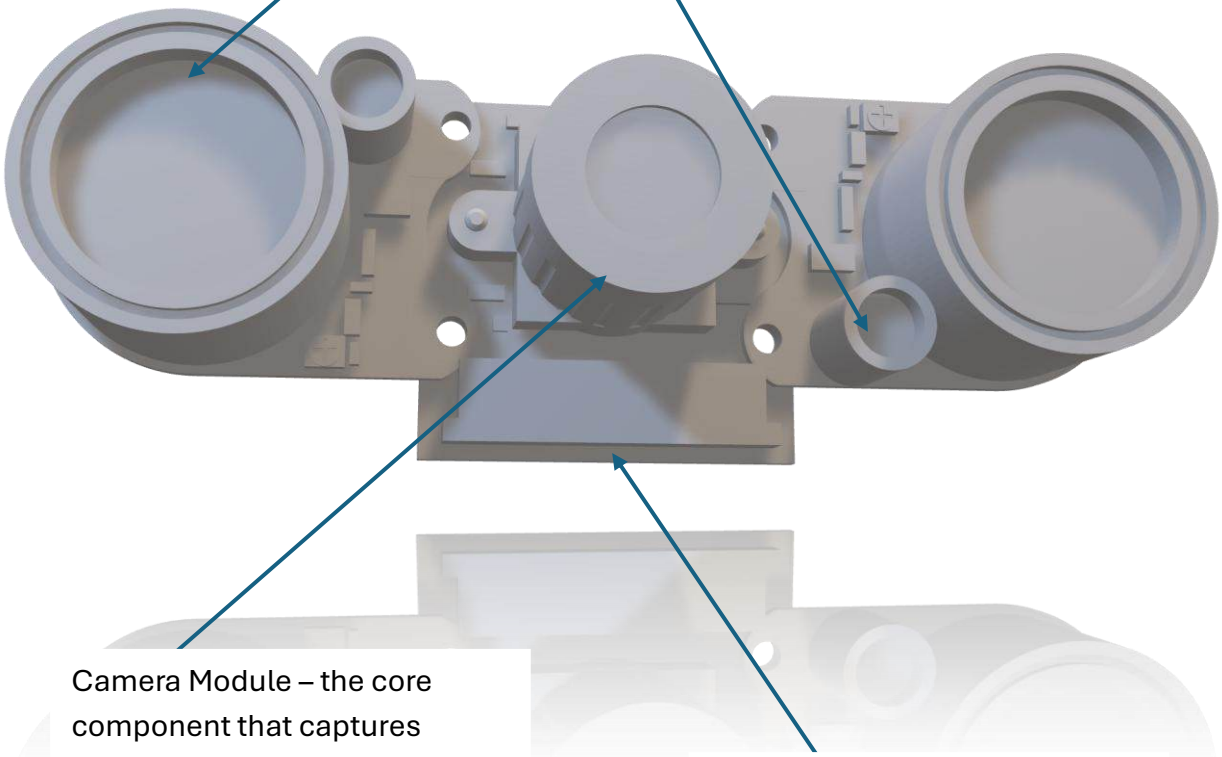
Raspberry Pi RP3A0 SiP – a quad-core CPU with 512MB RAM, offering 40% better single-thread and 5x multi-thread performance than its predecessor.

USB On-the-Go (OTG) - for compact connectivity, powered by a Broadcom IC, and supports devices via an OTG cable or powered USB hub.

RASPBERRY PI 5:

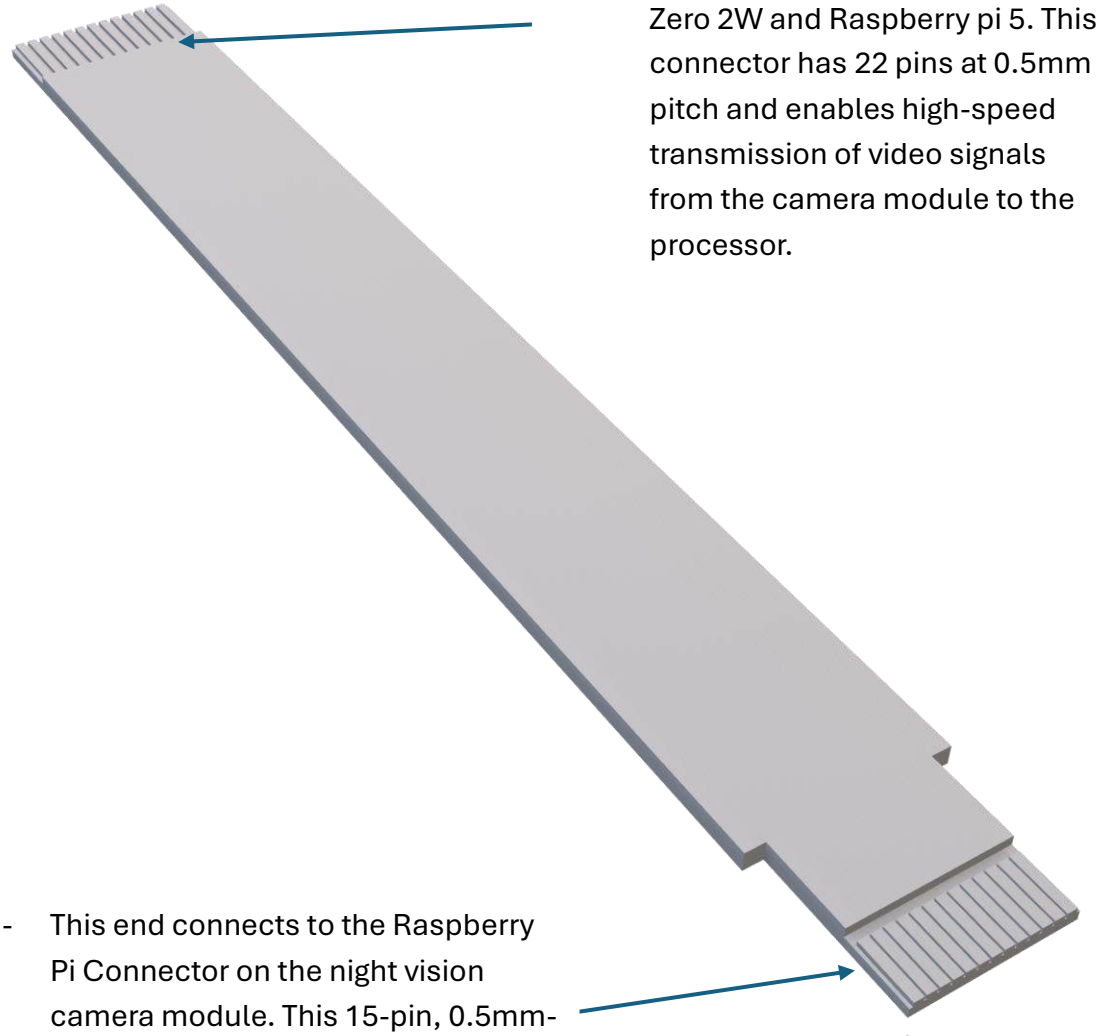


RASPBERRY PI NIGHT VISION CAMERA:



RASPBERRY PI ZERO & PI 5 CAMERA STRIP

- This end of the flexible flat cable (FFC) is inserted into the Camera Connector on the Raspberry Pi Zero 2W and Raspberry pi 5. This connector has 22 pins at 0.5mm pitch and enables high-speed transmission of video signals from the camera module to the processor.

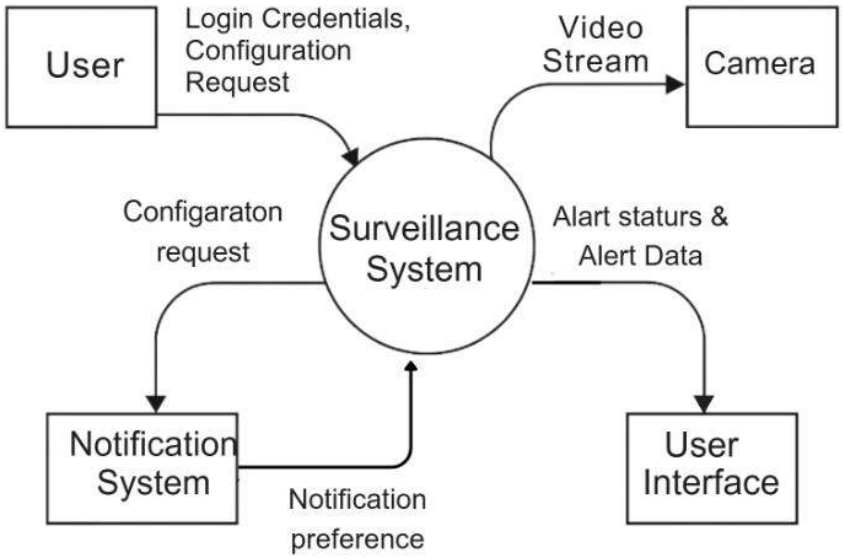


- This end connects to the Raspberry Pi Connector on the night vision camera module. This 15-pin, 0.5mm-pitch interface allows both power and video signal transmission, enabling the camera module to function in standard and infrared modes.

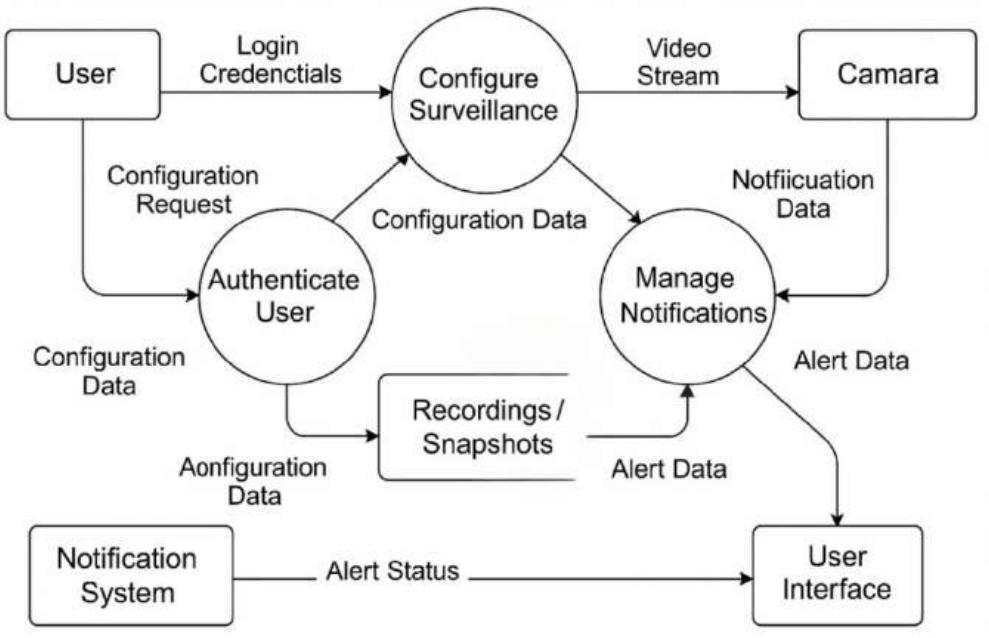
Developed system architecture flow data diagrams illustrating:

- The surveillance system's processes, including user authentication, configuration management, live video streaming from cameras, handling of recordings and snapshots, and the generation and management of notifications and alerts for users via a dedicated interface.

Level 0



Level 1



Brainstorming Sessions

Power Solutions:
Various power options were considered, including wired mains power and rechargeable battery systems. A portable power bank was ultimately selected due to its flexibility, ease of replacement, and suitability for temporary or remote deployments. This choice supports mobility without compromising performance and aligns with creating an autonomous surveillance solution.

Material Selection:
Material options were compared based on environmental resistance, printability, and structural integrity. PLA was chosen for indoor enclosures due to its ease of use, low cost, and aesthetic finish. For outdoor components, PETG was selected for its superior durability, UV resistance, and ability to withstand fluctuating weather conditions without warping or degradation.

Security Features:
Several advanced security measures were explored, with a focus on both software and hardware protection. Initial concepts included local data encryption (AES standards), multi-factor authentication for remote access, and experimental integration of AI-powered anomaly detection to enhance proactive threat identification. These ideas informed the development of a secure, reliable system tailored to real-world security concerns.

Criteria To Evaluate Success

- **Functionality**
    - **Reliable Motion Detection & Video Capture:**
      The system must consistently detect motion across a range of environments, including during variable lighting and movement speeds. Success will be determined through iterative testing of detection thresholds, false positive rates, and video quality at different distances and angles.
    - **Low-Light & Night Vision Performance:**
      Infrared (IR) capabilities will be assessed based on clarity, range, and accuracy in dark conditions. This includes testing both passive and active IR effectiveness, ensuring the system maintains usability and visual fidelity after sunset or in shadowed zones.
- **Performance & Efficiency**

- **Processing Speed & System Responsiveness:**
  The time between motion detection and video activation must be minimal (<1 second), and the system must operate with negligible latency to ensure real-time responsiveness. Benchmarks will be established through controlled timing tests.

- **Energy Consumption & Thermal Performance:**
  Power usage will be evaluated using monitoring tools to ensure sustainability and compatibility with long-term deployment (e.g., solar-powered setups and power banks). Efficient thermal management will also be considered to prevent overheating and extend component lifespan.

- **Code & Hardware Optimisation:**
  Emphasis on modular, efficient code and minimal redundancy. Evaluation includes system boot time, memory usage, and integration with lightweight libraries or frameworks.

- **User Experience (UX)**

  - **Intuitive Setup & Usability:**
    Setup time should be under 10 minutes with minimal technical knowledge required. Interface design will be critiqued for clarity, accessibility, and guided user flows. Success will also be evaluated through user testing with individuals unfamiliar with the system.

  - **Remote Access & Feedback Systems:**
    Usability of remote login, live stream functionality, and system alerts (e.g., email/push notifications) will be assessed. Responsive design and compatibility across devices (smartphones, tablets, desktops) are key indicators of success.

- **Security & Data Integrity**

  - **Robust Security Protocols:**
    The system must employ strong authentication methods (e.g., hashed passwords, 2FA optionality) and data encryption for both local and remote storage. Evaluation includes testing for potential vulnerabilities such as unauthorised access or data breaches.

  - **Secure Storage & Data Management:**

Storage must be protected against physical and cyber threats. Integration with cloud-based services (if applicable) must meet industry standards for data protection (e.g., GDPR, AES-256 encryption).

- **Cost-Effectiveness & Scalability**
    - **Affordability vs. Commercial Systems:**
      The final prototype will be evaluated against comparable commercial surveillance systems in terms of both upfront and long-term costs. Consideration includes component costs, maintenance, and upgrade potential.
    - **Breakdown of Project Costs:**
      All expenses will be transparently documented, including prototyping iterations, hardware, software licenses (if any), and time investment. Scalability will be considered in terms of how easily the system can be reproduced or expanded (e.g., adding more cameras).
- **Innovation & Design Justification**
    - **Originality in Design Approach:**
      Innovation will be evaluated based on how the system integrates existing technologies in a novel or efficient way, solves real-world problems, or enhances existing surveillance solutions.
    - **Alignment with Design Brief & End-User Needs:**
      The system's effectiveness will be directly linked to how well it meets the stated needs and constraints identified in the project brief and feedback from the target user base.

Project Management

Action Plan:

- Research → Hardware Setup → Software Development → 3D Printing → Testing → Refinements.

Time Management and Planning:

- **The school provided a time planner**

WEEKS	**MON**	**TUE**	**WED**	**THU**	**FRI**	**SAT**	**SUN**
TERM 4 - 2024 PROJECT PLANER							
• Factors affecting the design and production of needs analysis. Major design idea selection							
1							
2							
3							
• Areas of investigation, research methods, data collection/experimentation, communication, and presenting information. MDP progress							
4							
5							
6							
• Work of designers. Success and failure in design, Design practices, and progress. Selecting resources. (**MDP proposal**)							
7							
8							
9							

WEEKS	MON	TUE	WED	THU	FRI	SAT	SUN
10							
TERM 1 - 2025 PROJECT PLANER							
• Trends in designing and producing, evaluation criteria/ process, impact of MDP							
1							
2							
• Historical and cultural influences, impacts of design and innovation on society and environment, the importance of innovation, project management, research methods, and communication techniques. (**Innovation case study**)							
3							
4							
5							
6							
7							
8							
9							
10							
TERM 2 - 2025 PROJECT PLANER							
• Success of innovation, roles of agencies, and practices in industrial and commercial settings. And commercial settings. MDP progress							
1							

WEEKS	MON	TUE	WED	THU	FRI	SAT	SUN
2							
3							
4							
5							
• Entrepreneurial activity, continue with the construction of MDP (**Management plan and report for MDP**)							
6							
7							
8							
9							
10							
TERM 3 - 2025 PROJECT PLANER							
• Innovative approaches, creative thinking, the emergence of new technologies (**Trial HSC exams**)							
1							
2							
3							
4							
5							

WEEKS	MON	TUE	WED	THU	FRI	SAT	SUN
• innovative approaches, creative thinking, and the emergence of new technologies (**Trial HSC exams**)							
6							
7							
8							
9							
10							

- **Project time management, Gantt chart**

Project Phase	Key Tasks	Timeline (Weeks)	Notes
1. Research & Planning	- Needs analysis, problem identification	Term 4, Weeks 1-3	Includes user research, interviews, surveys, and literature review
	- Areas of investigation & technology research	Term 4, Weeks 2-5	Hardware, software, security protocols, and AI research
	- Project proposal and initial design brief	Term 4, Weeks 4-6	Finalize project scope and design criteria
2. Design & Prototyping	- Hardware selection & procurement	Term 4, Weeks 5-7	Raspberry Pi Zero 2 W, night vision camera, power supply
	- Software environment setup (OS, Frigate, Docker)	Term 4, Weeks 6-8	Install and configure software components
	- CAD design & 3D printing prototypes	Term 4, Weeks 7-9	Enclosure design iterations, material selection
3. Development	- Hardware assembly & wiring	Term 1, Weeks 1-3	Assemble Raspberry Pi, camera, power system
	- Software coding & integration (motion detection, UI)	Term 1, Weeks 2-5	Develop Python scripts, integrate OpenCV, Flask API
	- AI features implementation (facial recognition, AI detection)	Term 1, Weeks 4-6	TensorFlow Lite integration, model training/testing

4. Testing & Debugging	- Functional testing (motion detection, night vision)	Term 1, Weeks 6-8	Test system responsiveness, accuracy, and night vision performance
	- Security testing (encryption, access control)	Term 1, Weeks 7-9	Validate data protection and remote access security
	- User testing and feedback incorporation	Term 1, Weeks 8-10	Peer review, user experience improvements
5. Refinement & Finalisation	- Hardware & software refinements	Term 2, Weeks 1-4	Address bugs, optimise code, improve enclosure design.
	- Documentation & project report writing	Term 2, Weeks 3-6	Compile research, development process, evaluations
	- Final evaluation & comparison with commercial systems	Term 2, Weeks 5-7	Reflect on project success, future improvements
6. Presentation & Submission for feedback	- Prepare presentation materials (slides, charts)	Term 2, Weeks 6-8	Summarise key findings, project outcomes
	- Submit final project and portfolio	Term 2, Week 9	Complete all deliverables

Financial Planning:

Running under a budget of $250 for the main components

Item	Cost (AUD)
Raspberry Pi Zero 2W	$ 50 @ Amazon.com
Raspberry Pi 5	$100 @ Amazon.com
Raspberry Pi 5MP Night Vision Cam	$ 29 @ Amazon.com
MicroSD Card (64GB)	“Free from an old camera found at home” Prince on the market is $43 @ The Good Guys.
Power Supply/Power Bank	“Gift from a friend” Price on the market is $53.99 @ Amazon.
Camera Connector Strip	$0.81 @ AliExpress
Housing/Printing Plastic	$29 Amazon
Mini HDMI cable	$10 @Amazon
Total:	$218.80

Project Development and Realisation

Hardware Integration and Enclosure Optimisation

The hardware assembly began with the Raspberry Pi Zero 2 W, but as the project progressed, it became clear that its limited processing power would not support advanced features such as AI-powered detection and high-resolution, real-time video streaming. This prompted a pivotal upgrade to the Raspberry Pi 5, whose enhanced CPU, increased RAM, and improved connectivity provided a robust foundation for demanding surveillance tasks and ensured future scalability.

The Waveshare Raspberry Pi Night Vision Camera was selected for its superior infrared sensitivity and seamless compatibility with the Pi 5, enabling reliable 24/7 monitoring in diverse lighting conditions.

The camera and board were mounted in a custom 3D-printed enclosure designed for modularity, airflow, and ease of assembly. The enclosure's development was an iterative process, marked by several 3D printing challenges:

- **Over-Extrusion and Blobby Prints:**
  Early prints suffered from excess filament and poor detail, resulting in blobby or imprecise parts. This was resolved by reducing the nozzle temperature and recalibrating extrusion settings, which improved surface finish and dimensional accuracy.

- **Poor Bed Adhesion:**
  Some prints detached from the bed mid-process, leading to incomplete parts. This was addressed by thoroughly cleaning the print bed and slowing down the first layer print speed to enhance adhesion.

- **Incomplete Prints and Repairs:**
  When prints failed partway, salvageable components were reassembled using strong adhesive, allowing prototyping to continue without major delays. Design files were iteratively improved to minimise future failures.

These practical adjustments resulted in a robust, well-ventilated enclosure with secure mounting points for the Pi 5 and IR LEDs, and well-organised cable routing. Extended runtime and stress testing confirmed the enclosure's effectiveness in maintaining stable operating temperatures, even under continuous load.


IR LEDS
Camera
Air flow vents
Cable routing

Transition from Third-Party to Custom Software Stack

Initial development leveraged established surveillance platforms, MotionEyeOS, Frigate, and Kerberos.io, to accelerate deployment. However, each tool presented significant compatibility and performance roadblocks:

- MotionEyeOS was initially chosen for its lightweight design and ease of setup on the Pi Zero, but it lacked support for the modern libcamera stack required by the night vision camera. Even when tested on the Pi 5, MotionEyeOS was discontinued and could not reliably detect or stream from the camera.
- Frigate was considered for its AI detection, but its heavy reliance on x86 architecture and GPU acceleration made it unsuitable for the Pi 5, resulting in frequent crashes and unreliable video streams.
- Kerberos.io was explored as an alternative, but it also failed to detect the camera module, mirroring the issues faced with MotionEyeOS.

After multiple failed attempts and extensive troubleshooting, including re-flashing SD cards, updating firmware, and experimenting with different camera drivers, it became clear that a custom software solution was necessary. This pivot enabled full control over system features, reliability, and future scalability.

Custom Web Interface and Feature Set

The new custom stack was designed from the ground up with a focus on flexibility, performance, and user experience, incorporating a robust selection of technologies: HTML for structural content and controls, CSS for responsive and visually appealing layouts, JavaScript for dynamic features such as live video updates, user interaction, and advanced capabilities like motion detection, object recognition, and notifications, and Python (Flask) for backend logic, video streaming, and secure data management. The web interface was designed to include:

- Live Camera Feed: Real-time streaming from the night vision camera.

```
HTML

<!-- Live Camera Feed Section -->
<section id="videoSection" aria-label="Live Camera Feed">
 <div id="videoHeader">
  Live Camera Feed
  <div class="buttons" style="margin-left:auto; gap: 0.75rem;">
   <!-- Button to flip the camera -->
   <button id="flipCameraBtn" class="iconButton" aria-label="Flip
Camera" title="Flip Camera">
    <!-- SVG icon for flipping camera -->
    <svg viewBox="0 0 24 24" fill="none" stroke="currentColor"
stroke-width="2" stroke-linecap="round" stroke-linejoin="round"
style="width:16px; height:16px;">
     <path d="M7 7L3 12L7 17" />
```

```
     <path d="M17 7L21 12L17 17" />
     <line x1="12" y1="5" x2="12" y2="19" stroke-dasharray="4 2" />
     <path d="M12 3a3 3 0 0 1 3 3" />
     <path d="M12 21a3 3 0 0 0-3-3" />
    </svg>
   </button>
   <!-- Button to toggle fullscreen -->
   <button id="fullscreenBtn" class="iconButton" aria-label="Toggle
Fullscreen" title="Toggle Fullscreen">
    <svg viewBox="0 0 24 24"><path d="M7 14H5v5h5v-2H7v-3zm10-4h-
3v2h3v3h2v-5h-2zm-3-4v2h3v3h2V7h-5zm-4 0H5v5h2V7h3V5z"/></svg>
   </button>
  </div>
 </div>
 <!-- Video element to display the live feed -->
 <video id="webcamVideo" autoplay muted playsinline></video>
</section>
```

- Snapshot & Recording: On-demand image capture and video recording.

```
JAVASCRIPT

// Take Snapshot Functionality
snapshotBtn.addEventListener('click', () => {
 if (!currentStream) return; // Ensure there is an active stream
 const canvas = document.createElement('canvas');
 canvas.width = video.videoWidth; // Set canvas width to video width
 canvas.height = video.videoHeight; // Set canvas height to video
height
 const ctx = canvas.getContext('2d');
 ctx.drawImage(video, 0, 0, canvas.width, canvas.height); // Draw the
video frame on the canvas
 const imgUrl = canvas.toDataURL('image/png'); // Convert canvas to
image URL

 // Create a link to download the snapshot
 const a = document.createElement('a');
 a.href = imgUrl;
 a.download = `snapshot_${new Date().toISOString().replace(/[:.]/g,
'-')}.png`; // Set filename
 document.body.appendChild(a);
 a.click(); // Trigger download
 document.body.removeChild(a); // Clean up
});

// Recording Capability
playBtn.addEventListener('click', () => {
 if (!currentStream) {
  addEvent('No active webcam stream to record.'); // Notify if no
stream
  return;
 }
 if (!isRecording) {
  recordedChunks = []; // Reset recorded chunks
  mediaRecorder = new MediaRecorder(currentStream, { mimeType:
'video/webm' }); // Create MediaRecorder
  mediaRecorder.ondataavailable = (event) => {
   if (event.data.size > 0) {
    recordedChunks.push(event.data); // Store recorded data
   }
```

```
  };
  mediaRecorder.onstop = () => {
   const blob = new Blob(recordedChunks, { type: 'video/webm' }); //
Create a blob from recorded chunks
   const url = URL.createObjectURL(blob); // Create a URL for the
blob
   const a = document.createElement('a');
   a.style.display = 'none';
   a.href = url;
   a.download = `recording_${new
Date().toISOString().replace(/[:.]/g, '-')}.webm`; // Set filename
   document.body.appendChild(a);
   a.click(); // Trigger download
   setTimeout(() => {
    document.body.removeChild(a);
    URL.revokeObjectURL(url); // Clean up
   }, 100);
  };
  mediaRecorder.start(); // Start recording
  isRecording = true; // Update recording state
 } else {
  mediaRecorder.stop(); // Stop recording
  isRecording = false; // Update recording state
 }
});
```

- Motion Detection: Real-time alerts and automated recording on movements.

```
JAVASCRIPT

// Motion Detection Activation
motionDetectionBtn.addEventListener('click', () => {
 motionPressed = !motionPressed; // Toggle motion detection state
 if (motionPressed) {
  motionIconInner.style.display = 'block'; // Show active icon
  startMotionDetection(); // Start motion detection
  addEvent('Motion detection activated.'); // Log event
 } else {
  motionIconInner.style.display = 'none'; // Hide active icon
  stopMotionDetection(); // Stop motion detection
  addEvent('Motion detection deactivated.'); // Log event
 }
});

// Function to start motion detection
function startMotionDetection() {
 // Initialize canvases for motion detection
 canvas1 = document.createElement('canvas');
 canvas2 = document.createElement('canvas');
 ctx1 = canvas1.getContext('2d');
 ctx2 = canvas2.getContext('2d');
 motionDetectionStarted = true; // Update state
 requestAnimationFrame(motionLoop); // Start the motion loop
}

// Motion detection loop
function motionLoop() {
 if (!motionDetectionStarted) return; // Exit if not started
 if (!currentStream) {
  requestAnimationFrame(motionLoop); // Wait for stream
```

```
  return;
 }
 // Capture frames and detect motion...
}
```

- Object Recognition: AI-based identification of people, pets or vehicles.

```
JAVASCRIPT

// Object Recognition Activation
objectRecognitionBtn.addEventListener('click', () => {
 objectRecognitionPressed = !objectRecognitionPressed; // Toggle
object recognition state
 if (objectRecognitionPressed) {
  objectRecognitionIconInner.style.display = 'block'; // Show active
icon
  startRecognition(); // Start object recognition
  addEvent('Object recognition activated.'); // Log event
 } else {
  objectRecognitionIconInner.style.display = 'none'; // Hide active
icon
  stopRecognition(); // Stop object recognition
  addEvent('Object recognition deactivated.'); // Log event
 }
});

// Start object recognition
function startRecognition() {
 intervalId = setInterval(async () => {
  if (model && video.readyState === 4) {
   ctx.clearRect(0, 0, canvas.width, canvas.height); // Clear
previous drawings
   const predictions = await model.detect(video); // Get predictions
from the model
   drawPredictions(predictions); // Draw predictions on canvas
  }
 }, 500); // Run every 500ms
}

// Draw predictions on the canvas
function drawPredictions(predictions) {
 predictions.forEach(pred => {
  ctx.strokeStyle = "#00FF00"; // Set bounding box color
  ctx.lineWidth = 2; // Set bounding box line width
  // Calculate and draw bounding box...
 });
}
```

- Detection Zones: User-defined areas to minimise false alarms.

```
JAVASRIPT

// Drawing Detection Zones
overlayCanvas.addEventListener('mousedown', (evt) => {
 isDrawing = true; // Start drawing
 const pos = getMousePos(overlayCanvas, evt); // Get mouse position
 drawStartX = pos.x; // Store starting X
 drawStartY = pos.y; // Store starting Y
```

```
 detectionZone = null; // Reset detection zone
 drawDetectionZone(); // Draw initial zone
});

// Update detection zone on mouse move
overlayCanvas.addEventListener('mousemove', (evt) => {
 if (!isDrawing) return; // Exit if not drawing
 const pos = getMousePos(overlayCanvas, evt); // Get current mouse
position
 const x = Math.min(drawStartX, pos.x); // Calculate X
 const y = Math.min(drawStartY, pos.y); // Calculate Y
 const width = Math.abs(pos.x - drawStartX); // Calculate width
 const height = Math.abs(pos.y - drawStartY); // Calculate height
 detectionZone = { x, y, width, height }; // Update detection zone
 drawDetectionZone(); // Redraw zone
});

// Clear Detection Zone
clearZoneBtn.addEventListener('click', () => {
 detectionZone = null; // Clear detection zone
 drawDetectionZone(); // Redraw
 addEvent('Detection zone cleared. Motion detection applies to full
video frame.'); // Log event
});
```

- Notification Settings: Customisable email or push alerts

```
HTML

<!-- Notifications Settings Section -->
<section id="notifications" class="settingsSection" aria-
label="Notification Settings" role="tabpanel" tabindex="0"
style="display: none;">
 <h2>Notifications</h2>
 <label title="Toggle email notifications">
  <input type="checkbox" id="emailToggle" checked /> Email
Notifications
 </label>
 <label for="emailReceivers" title="Comma separated list of email
recipients">
  Email Receivers:
 </label>
 <textarea id="emailReceivers" placeholder="example1@mail.com,
example2@mail.com"></textarea>
 <label title="Toggle MQTT notifications">
  <input type="checkbox" id="mqttToggle" /> MQTT Notifications
 </label>
 <small class="description">
  MQTT notifications will send alerts to your smart home system.
 </small>
</section>
```

- Event Logs: History record of detections and system events

```
JAVASCRIPT

// Event Logging Function
function addEvent(message) {
```

```
 const eventTime = new Date().toLocaleTimeString(); // Get current
time
 const div = document.createElement('div'); // Create new event entry
 div.className = 'event-entry';
 div.innerHTML = `<span class="event-
timestamp">[${eventTime}]</span>${message}`; // Format entry
 eventsDiv.insertBefore(div, eventsDiv.firstChild); // Add to event
log
 while (eventsDiv.childNodes.length > 50) {
  eventsDiv.removeChild(eventsDiv.lastChild); // Limit log size
 }
}

// Clear Event Log Button
clearEventsBtn.addEventListener('click', () => {
 eventsDiv.innerHTML = '<p style="text-align:center; color:#555;">No
events recorded yet.</p>'; // Reset log
});
```

- User Authentication: Secure login to protect system settings and footage.

```
JAVASRIPT

<!-- Login Modal -->
<div id="loginModal" aria-modal="true" role="dialog" aria-
labelledby="loginTitle" aria-describedby="loginDesc" style="position:
fixed; inset: 0; background: rgba(18, 24, 38, 0.95); display: flex;
justify-content: center; align-items: center; z-index: 9999;">
 <div style="background: #1e2235; padding: 2rem; border-radius: 16px;
box-shadow: 0 6px 15px rgba(0, 0, 0, 0.7); width: 320px; font-family:
'Montserrat', sans-serif; color: #f1f1f1;">
  <h2 id="loginTitle" style="margin-top: 0; margin-bottom: 1rem;
font-weight: 600; text-align: center;">Login</h2>
  <form id="loginForm" novalidate>
   <label for="username">Username</label>
   <input type="text" id="username" name="username" required />
   <label for="password">Password</label>
   <input type="password" id="password" name="password" required />
   <div id="loginError" style="color: #ff6347; display:
none;">Invalid username or password.</div>
   <button type="submit">Log In</button>
  </form>
 </div>
</div>

// Login Form Submission
loginForm.addEventListener('submit', (e) => {
 e.preventDefault(); // Prevent default form submission
 const username = loginForm.username.value.trim(); // Get username
 const password = loginForm.password.value; // Get password

 if (username === 'admin' && password === 'secret') {
  loginError.style.display = 'none'; // Hide error
  loginModal.style.display = 'none'; // Close modal
 } else {
  loginError.style.display = 'block'; // Show error
 }
});
```

- Responsive Design: Accessible from desktop or mobile device.

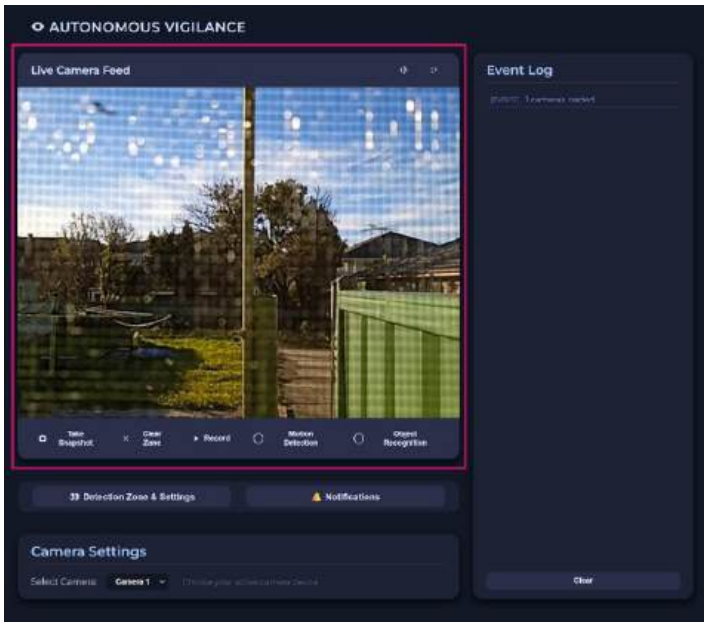
```
CSS

<!-- Login Modal -->
<div id="loginModal" aria-modal="true" role="dialog" aria-
labelledby="loginTitle" aria-describedby="loginDesc" style="position:
fixed; inset: 0; background: rgba(18, 24, 38, 0.95); display: flex;
justify-content: center; align-items: center; z-index: 9999;">
 <div style="background: #1e2235; padding: 2rem; border-radius: 16px;
box-shadow: 0 6px 15px rgba(0, 0, 0, 0.7); width: 320px; font-family:
'Montserrat', sans-serif; color: #f1f1f1;">
  <h2 id="loginTitle" style="margin-top: 0; margin-bottom: 1rem; font-
weight: 600; text-align: center;">Login</h2>
  <form id="loginForm" novalidate>
   <label for="username">Username</label>
   <input type="text" id="username" name="username" required />
   <label for="password">Password</label>
   <input type="password" id="password" name="password" required />
   <div id="loginError" style="color: #ff6347; display: none;">Invalid
username or password.</div>
   <button type="submit">Log In</button>
  </form>
 </div>
</div>

// Login Form Submission
loginForm.addEventListener('submit', (e) => {
 e.preventDefault(); // Prevent default form submission
 const username = loginForm.username.value.trim(); // Get username
 const password = loginForm.password.value; // Get password

 if (username === 'admin' && password === 'secret') {
  loginError.style.display = 'none'; // Hide error
  loginModal.style.display = 'none'; // Close modal
 } else {
  loginError.style.display = 'block'; // Show error
 }
});
```

- User-Friendly Controls: Intuitive layout for non-technical users.

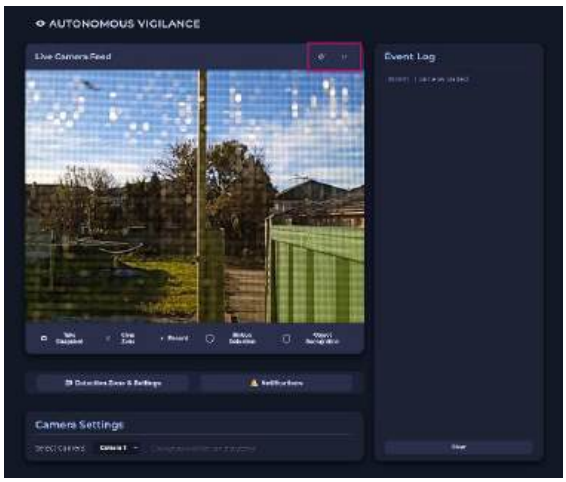
```
HTML

<!-- User Interface Elements -->
<div id="container">
 <div id="leftPanel">
  <!-- Video Section and Settings -->
 </div>
 <section id="eventLog" aria-label="Event Log">
  <h2>Event Log</h2>
  <div id="events">
   <p style="text-align:center; color:#555;">No events recorded yet.</p>
<!-- Placeholder for events -->
  </div>
  <button id="clearEventsBtn">Clear</button> <!-- Button to clear event
log -->
 </section>
</div>
```

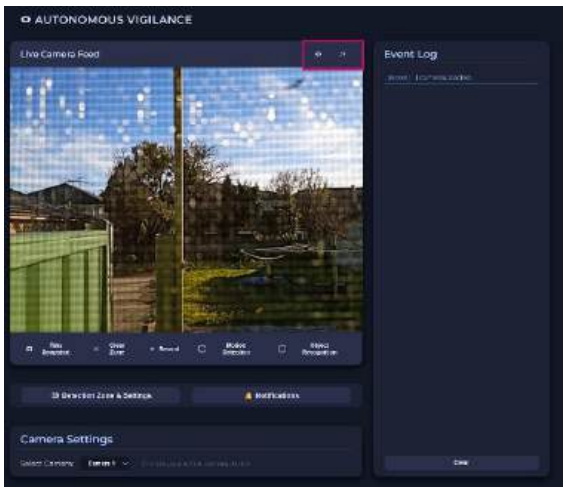
The custom web interface integrates all core functionalities into a unified, intuitive dashboard. Designed for clarity and efficiency, the interface enables users to access live video streaming, initiate recordings, and capture snapshots directly from their browser. Camera controls, including view flipping and full-screen toggling, are easily accessible, supporting flexible installation and immersive monitoring. Advanced features such as motion detection, adjustable detection zones, and real-time object recognition overlays empower users to tailor the system to their specific security needs. Each component of the interface is mapped to user requirements and system documentation, ensuring that the design not only meets technical specifications but also delivers a seamless and accessible user experience across devices. The following section highlights these key features of the final prototype, demonstrating their alignment with project requirements and their role in addressing the real-world problems defined at the outset of this project.

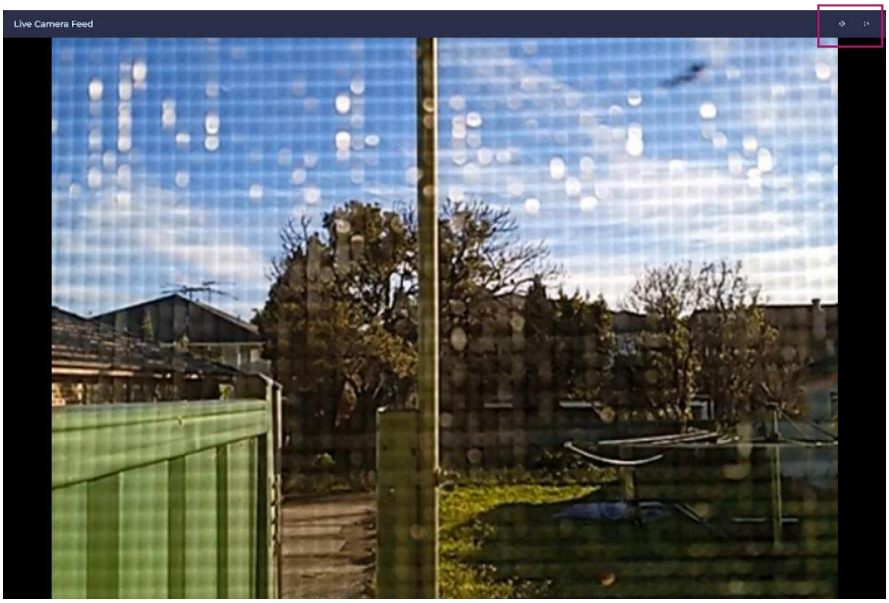
Final Prototype Features Showcase



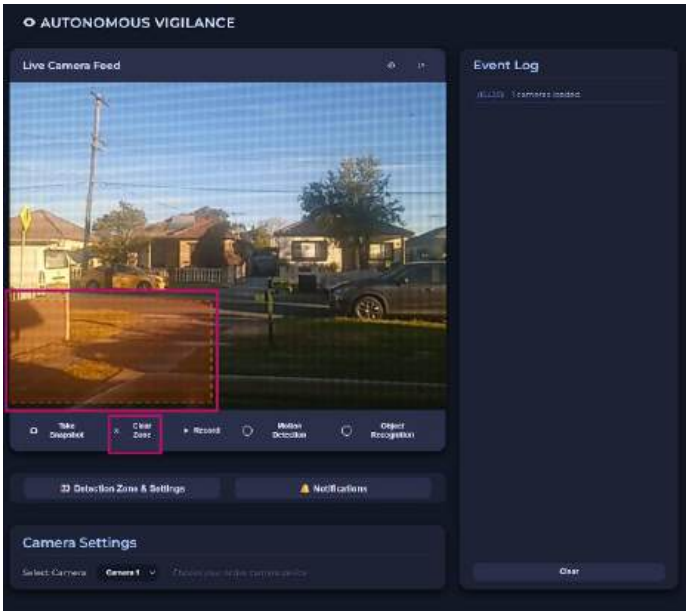
The live camera feed offers immediate, dependable visual monitoring, directly addressing the need for real-time surveillance identified in the Statement of Intent and Identification and Exploration of the Need. This feature allows users, including homeowners, renters, and small business owners, to monitor their premises at any time, as highlighted in the Target Audience section. The capability to record video streams as WEBM files and capture snapshots in PNG format facilitates evidence collection and review, meeting the requirements for local, secure storage and user-controlled data outlined in the System Requirements and Criteria to Evaluate Success.

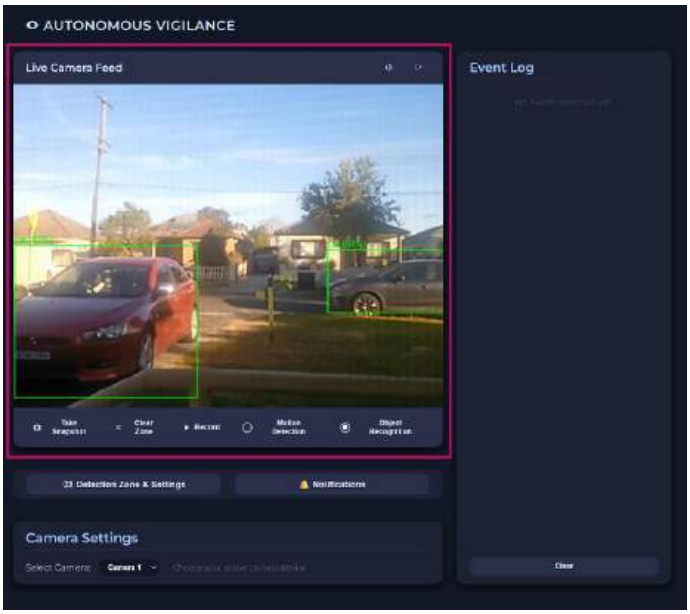




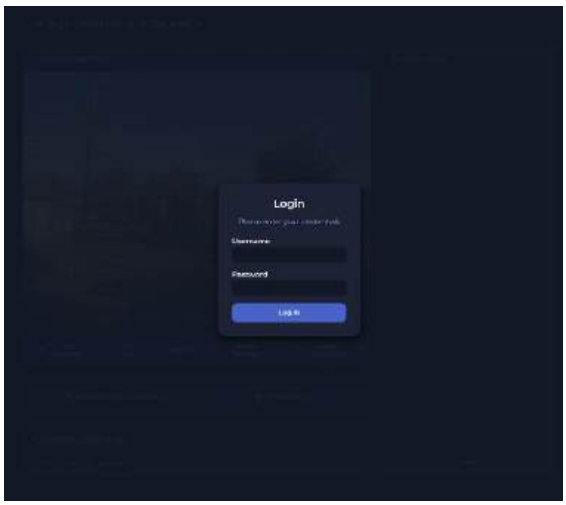


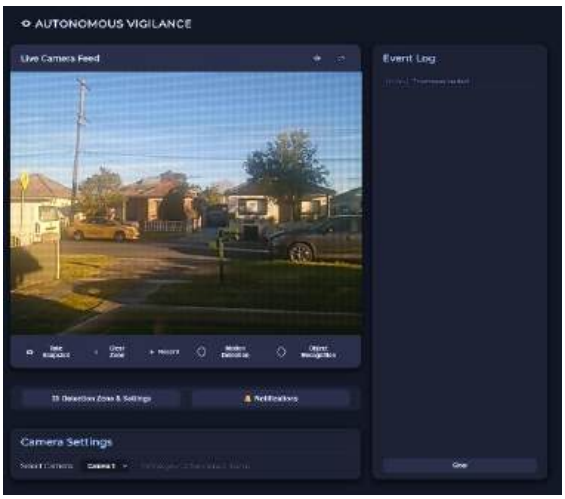
The camera control panel offers intuitive options such as flipping the camera view and toggling full-screen mode. These controls were developed in response to feedback gathered during User Research, which emphasised the need for a flexible, user-friendly interface. By supporting easy camera setup and immersive viewing, these controls fulfil the requirement for usability and adaptability described in both the Project Scope and Criteria to Evaluate Success.

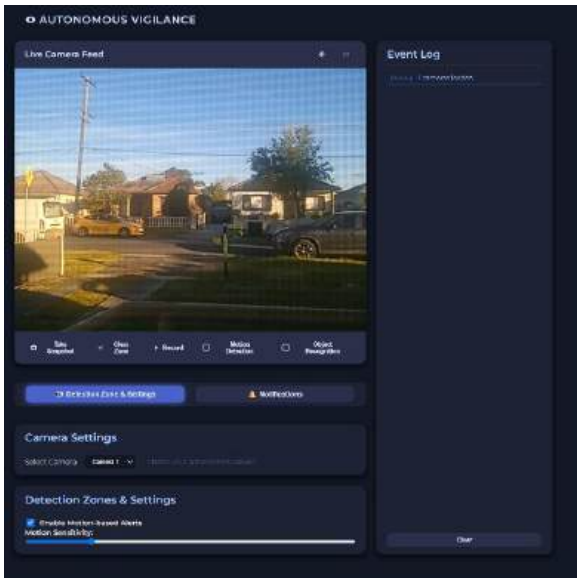


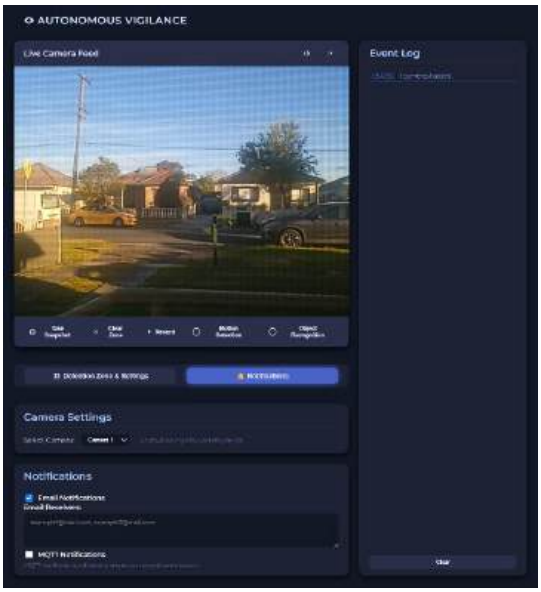


Motion detection and AI-powered object recognition are central features of Autonomous Vigilance, designed to address key user needs identified in the Statement of Intent, User Research, and Problem Statement. The motion detection system enables users to turn detection on or off, adjust sensitivity, and set custom detection zones, ensuring that alerts are relevant while minimising false positives. This directly addresses frustrations with commercial systems and the need for intelligent, user-defined monitoring. Furthermore, the integration of AI-powered object recognition allows the system to differentiate between people, vehicles, and animals in real time, significantly reducing false alarms and enhancing situational awareness. This advanced capability results directly from insights gained through Technology Research and meets the system requirements for adaptive, accurate detection. Together, these features demonstrate the project's commitment to innovation, continuous learning, and delivering a practical, user-centred security solution that fully aligns with the Evaluation Criteria and the overarching goals set out in the project documentation.









The dashboard is designed for clarity and accessibility, with clearly labelled buttons and a responsive layout that adapts to various devices. This approach directly addresses the requirement for a user-friendly setup and operation experience, as identified in the Statement of Intent, User Research, and Evaluation Criteria. The interface reflects feedback from user testing and iterative prototyping, ensuring a modern, intuitive, and robust experience for all users.

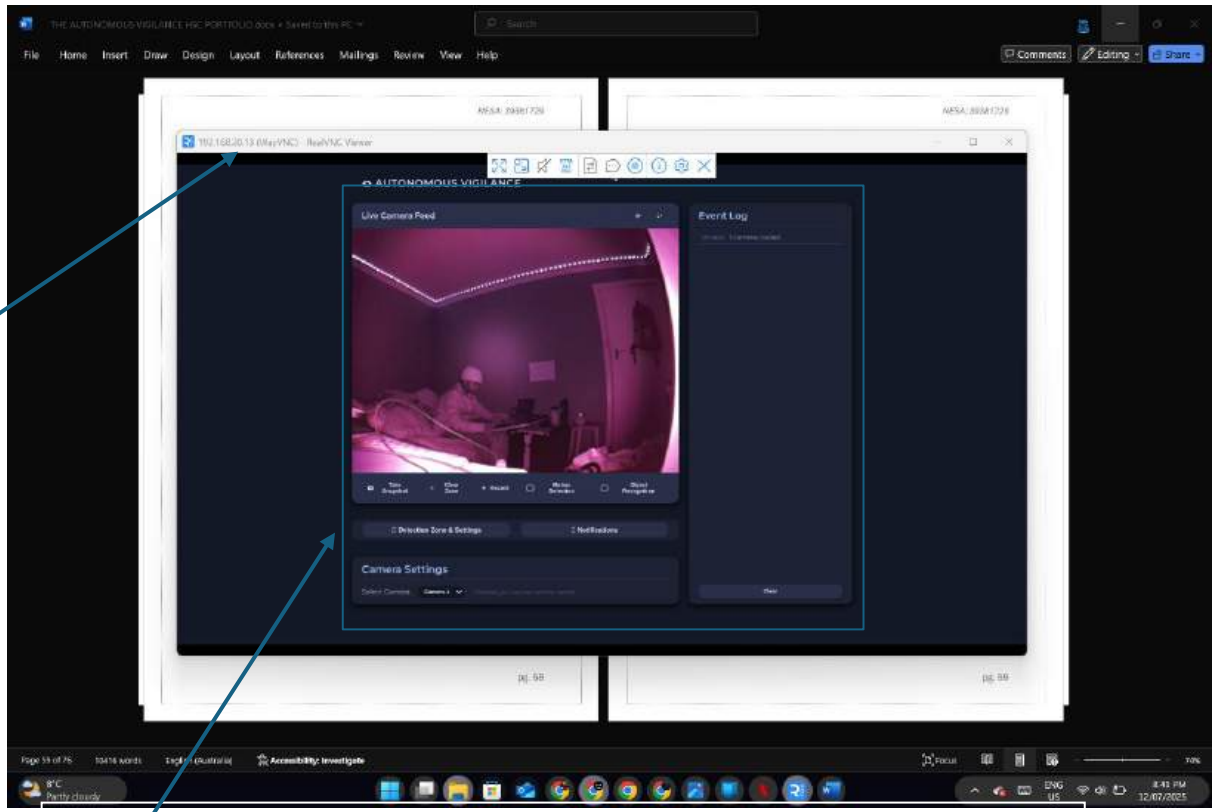
Feature Justification Summary Table

Feature	**Portfolio Reference(s)**	**Requirement/Need Addressed**
Live Streaming	Statement of Intent, Identification and Exploration of the Need, Criteria to Evaluate Success	Real-time, reliable monitoring; secure, local storage
Camera Controls	User Research, Project Scope, Criteria to Evaluate Success	Usability, flexibility, and a user-friendly experience
Motion Detection	User Research, Problem Statement, Evaluation Criteria	Custom alerts, reduced false positives
Object Recognition	System Requirements, Technology Research, Statement of Intent	AI-powered detection, innovation, and reduced false alarms
User Interface	Statement of Intent, User Research, Evaluation Criteria	Accessible, intuitive, and modern design

Remote Access and System Administration

To support flexible administration and troubleshooting, VNC Viewer was configured for secure, graphical remote access to the Raspberry Pi 5:

- Implementation:
    - VNC Server installed and configured, with access restricted to trusted devices.
    - Enabled headless operation, allowing all configuration and debugging to be performed remotely.
    - Supported multi-user access for collaborative troubleshooting and updates.
- Benefits:
    - Rapid Debugging: GUI-based access streamlines diagnosis of camera or network issues.
    - Enhanced Usability: Provides a familiar desktop for less technical users to adjust settings or review logs.
    - Security: Access protected by strong authentication and, where possible, SSH tunnelling.



VNC live view session on my windows pc

Live preview footage of the software running on the raspberry pi

Testing, Troubleshooting and Optimisation:

The development of Autonomous Vigilance required rigorous testing and continuous optimisation to ensure reliability, accuracy, and user satisfaction. This phase involved systematic scenario testing, interactive debugging, and targeted improvements based on real-world feedback and prototype evaluation.

Function and Testing

Scenario-Based Testing:

The system was tested in a variety of real-world conditions, including:

- **Day/Night Operation:**
  The camera's infrared capabilities were assessed in both bright and low-light environments. Night vision performance was validated by recording at various lux levels, confirming clear image capture even in total darkness.

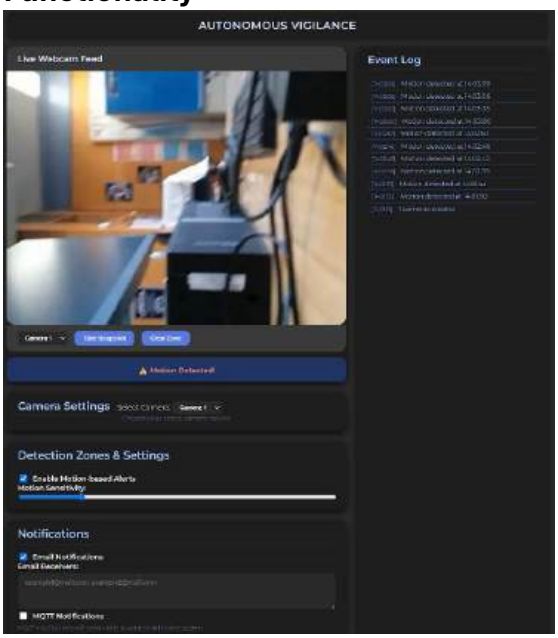
- **Motion Detection Accuracy:**
  Multiple test runs were conducted to evaluate the sensitivity and reliability of the motion detection algorithm. Adjustments to threshold values and detection zones were made to minimise false positives (e.g., pets, moving shadows) and false negatives (e.g., slow-moving intruders).

- **Web Interface Usability:**
  Prototypes of the web dashboard were tested on desktop and mobile devices to ensure responsive design, intuitive navigation, and real-time video streaming without lag. User feedback was gathered from peers and iteratively applied to improve button placement, event log clarity, and notification settings.

Prototype Evaluation and Iterative Improvements

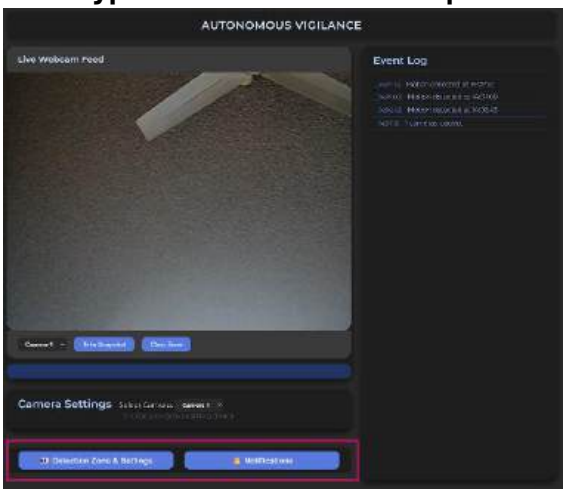
A key part of the development process for *Autonomous Vigilance* was the creation and evaluation of multiple dashboard prototypes. Each iteration addressed feedback from systematic testing and user evaluation, resulting in a more robust, intuitive, and feature-rich user experience. Below is a summary of each prototype stage and the targeted improvements made:

Prototype 0: Initial Features and Functionality



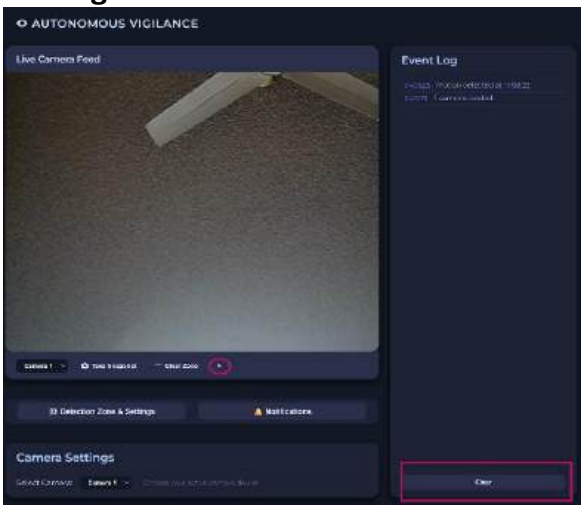
The initial prototype established a foundation with a dark-themed interface, live webcam feed, essential camera controls, and a simple alerts section. User interaction was limited to camera selection and snapshot capture, and the alerts section did not update in real time. This version demonstrated the system's core potential but highlighted the need for greater interactivity and responsiveness.

Prototype 1: Enhanced User Experience



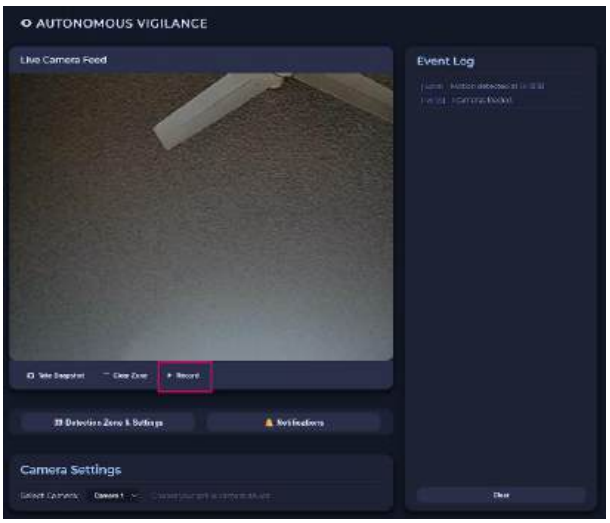
Building on the initial design, Prototype 1 introduced a more consistent colour scheme and cleaner layout, improving readability and visual appeal. The dashboard was restructured to separate the live feed from control panels, and the alerts section was made more prominent. These changes enhanced navigation and user engagement, making the dashboard more user-friendly.

Prototype 2: Introduction of Tabbed Settings



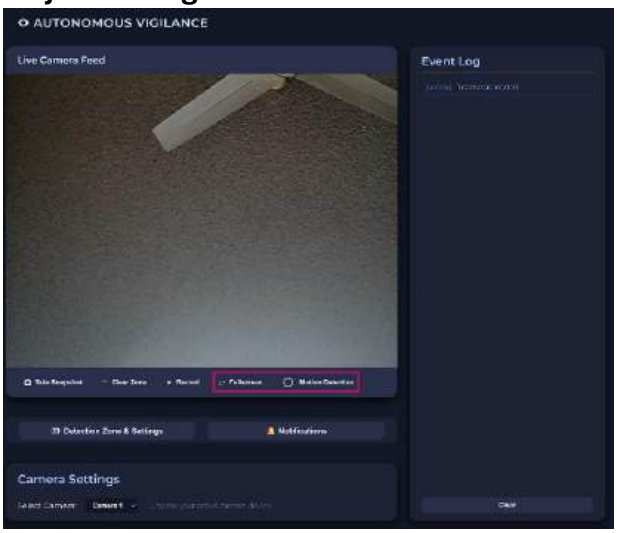
Prototype 2 implemented a tabbed interface for settings, allowing users to switch between Camera Settings, Detection Zones, and Notifications. Toggles were added to show or hide each section, reducing visual clutter. A recording feature was introduced, enabling video clip capture from the live feed, and the event log was improved for easier management. The login modal was also refined for clearer input focus and validation feedback.

Prototype 3: Dynamic Interaction and Feedback



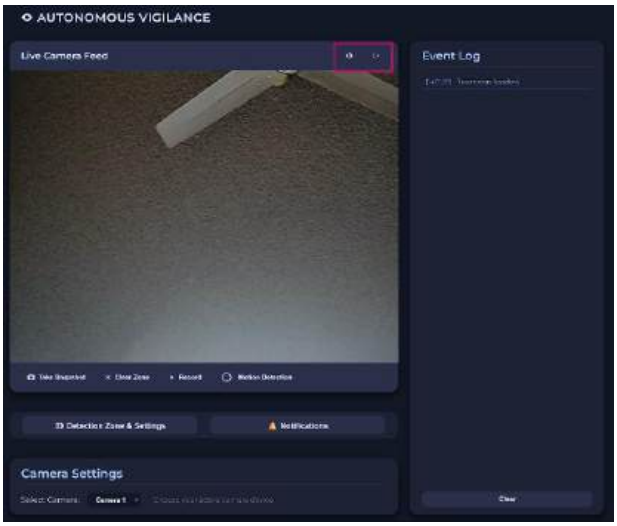
This iteration focused on real-time feedback and control. Motion detection updates were promptly logged in the event area, and the motion detection toggle included a clear visual indicator. Controls were grouped beneath the live feed with intuitive icons and hover effects. Accessibility was enhanced with ARIA roles and keyboard navigation, improving usability for a wider audience.

Prototype 4: Advanced Features and Object Recognition



Prototype 4 marked the introduction of AI-powered object recognition. Users could activate this feature via a dedicated toggle, with clear visual feedback. Detected objects were highlighted with bounding boxes and labels over the video stream. Additional controls, including full-screen mode and camera flipping, were added for flexibility. The design used a deep blue palette to highlight detection overlays while maintaining clarity.

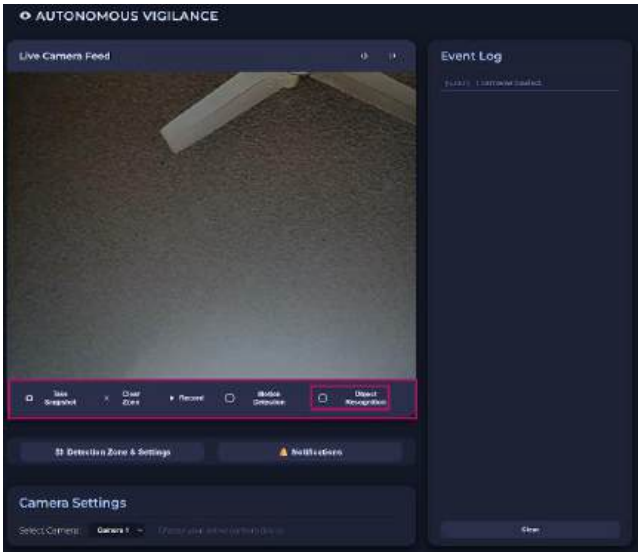
Prototype 5: Performance and Usability Enhancements



This version refined button presentation with subtle shadows and transitions, and the record button's label and icon toggled between "Record" and "Pause" for clarity. Video display improvements ensured distortion-free feeds across devices, and the overlay canvas adapted responsively to window resizing and full-screen toggles. These changes resulted in a more polished and intuitive experience

.

Prototype 6: Final Refinements



The final prototype synthesised all previous improvements into a cohesive, polished interface. A subtle logo and clear typography reinforced branding, while motion detection and object recognition toggles were distinct and accessible. The overlay canvas reliably supported both detection zones and object highlights, adapting to video and window changes. Recording and snapshot features operated smoothly, and performance optimisations ensured efficient resource use. Accessibility remained a priority, ensuring usability for all users. This final version embodies a modern, minimalist, and robust design aligned with user needs and professional standards.

Prototype Evolution Summary Table

Prototype	Focus / Title	Key Improvements and Features
Prototype 0	Initial Features and Functionality	- Dark-themed interface - Live webcam feed - Basic camera controls - Simple alerts section - Limited interaction (camera selection, snapshots) - No real-time alerts
Prototype 1	Enhanced User Experience	- Consistent colour scheme - Cleaner layout - Separated live feed and controls - More prominent alerts - Improved navigation and engagement

Prototype	Focus / Title	Key Improvements and Features
Prototype 2	Introduction of Tabbed Settings	- Tabbed settings (Camera, Detection Zones, Notifications) - Section toggles reduce clutter - Recording feature added - Improved event log - Refined login modal
Prototype 3	Dynamic Interaction and Feedback	- Real-time event logging - Motion detection toggle with indicator - Grouped controls with icons - Hover effects - Enhanced accessibility (ARIA, keyboard navigation)
Prototype 4	Advanced Features and Object Recognition	- AI-powered object recognition - Toggle for object detection - Bounding boxes and labels on video - Full-screen and camera flip controls - Deep blue palette for overlays
Prototype 5	Performance and Usability Enhancements	- Improved button styles (shadows, transitions) - Dynamic record/pause toggle - Distortion-free video display - Responsive overlay canvas - Smoother, more intuitive experience
Prototype 6	Final Refinements	- Cohesive, polished interface - Branding (logo, typography) - Distinct motion/object toggles - Reliable overlay canvas - Smooth recording/snapshot features - Performance optimisations - Accessibility prioritised

Debugging and Troubleshooting

Hardware Troubleshooting:

- **Camera Detection Issues:**
  Early on, the camera module was not recognised by third-party platforms (MotionEyeOS, Kerberos.io). This was traced to incompatibility with the libcamera stack. The solution involved switching to a custom Python/OpenCV pipeline using picamera2, which restored camera functionality and enabled direct control over image capture and streaming.

- **Thermal Management:**
  Stress testing revealed that the Raspberry Pi 5 could overheat during extended AI processing. This was mitigated by redesigning the enclosure for improved airflow and adding passive heat sinks.

- **3D Print Failures:**
  Issues such as over-extrusion, poor bed adhesion, and incomplete prints were resolved by calibrating print settings (lowering nozzle temperature, cleaning the print bed, and slowing the first layer) and, when necessary, using adhesive to repair salvageable parts.

Software Troubleshooting:

- **Software build:**
  During the development of Autonomous Vigilance, software troubleshooting was a continuous and integral process that addressed a wide range of technical issues throughout the codebase. Webcam access errors were common, including failures to obtain user media due to denied permissions or missing cameras, “no video devices found” messages, and errors when starting or stopping streams. The Media Recorder integration presented several challenges, including unsupported MIME types, limited browser support, and errors during recording start and stop events. Motion detection logic sometimes failed to initialise if video metadata was not loaded or if the video frame size was zero; scaling issues with detection zones occasionally led to inaccurate results, and the motion cooldown logic sometimes caused missed detections. Object recognition debugging included TensorFlow model loading failures, video readiness issues, improper clearing of detection intervals, and canvas drawing problems such as scaling and flipping errors. The user interface and event handling also required extensive troubleshooting, including insecure hardcoded login validation, potential race conditions with asynchronous camera enumeration, event log overflow, Fullscreen API compatibility issues, and occasional desynchronisation between

the camera flip state and the actual video feed. Managing canvas resizing and drawing was crucial to keep overlays aligned with the video during window or Fullscreen changes, addressing mismatches between video and canvas sizes. Accessibility was considered throughout, with ARIA attributes implemented for screen reader compatibility, though some dynamic changes still require further enhancement. Each of these issues was systematically identified through rigorous testing and debugging, and iterative solutions were developed to improve system reliability, user experience, and cross-platform compatibility.

- **Storage Corruption:**
  During extended testing, frequent SD card writes, particularly during unexpected power outages, resulted in file system corruption and occasional data loss. To address this, I implemented automated backups of critical system files to a portable USB flash drive, ensuring that recent data could be recovered even if the SD card became corrupted. Additionally, to further safeguard the Raspberry Pi 5 and its storage media, I positioned the device near an external fan to enhance cooling, supplementing the onboard cooling fan. This dual approach, combining regular USB backups with improved thermal management, significantly increased data integrity and system reliability, thereby providing a smooth build despite software and hardware failures.

- **Notification Reliability:**
  Email alert functionality is currently under development and exists as a prototype within the system. While the logic to determine when alerts should be triggered, whether by motion detection, object recognition, or a combination of both, has been implemented using JavaScript email APIs, this feature is not yet fully functional due to limited development time. Testing has identified areas for improvement, such as missed or duplicate notifications, which are actively being addressed through code refinement and the addition of retry mechanisms. Even after project submission, ongoing development will continue to finalise and optimise the email alert system. This iterative approach ensures that the feature will be robust and reliable in future releases, with room for further enhancement as user feedback is gathered and technical challenges are resolved.

Evaluation

Self-Assessment:

Project Overview

The Autonomous Vigilance project has successfully fulfilled the intentions outlined in my Statement of Intent by delivering a functional, cost-effective surveillance system that empowers users with enhanced security, privacy, and control. The integration of motion detection, local encrypted storage, and a customisable web interface directly addresses the limitations of commercial surveillance systems, such as high cost, limited customisation, and privacy concerns. These were identified as core issues in my initial research and design brief.

Key Achievements

One of the project's major achievements is the reliable performance of the night vision module, which ensures effective monitoring in low-light conditions. The development of a simple and user-friendly web interface has also streamlined setup and operation for end users, directly supporting my goal of creating an accessible, user-centred system. These outcomes demonstrate my ability to translate user needs and design criteria into practical solutions. The system's local, encrypted storage ensures privacy and data security, while the inclusion of customisable detection zones and real-time notifications has made the system more responsive and adaptable to different user environments. The modular hardware and software design also supports scalability, allowing for future expansion and upgrades.

Challenges and Problem Solving

During development, I encountered challenges in reducing power consumption for the Raspberry Pi and refining the motion detection algorithm to minimise false positives. These issues were resolved through repetitive prototyping, scenario-based testing, and targeted troubleshooting. Addressing these challenges also enabled me to advance my Python programming skills, creating efficient code for embedded systems, and to expand into front-end technologies such as HTML, CSS, and JavaScript, broadening my understanding of software development beyond what I initially knew.

Areas for Future Development

While the system performed reliably in most conditions and met nearly all of the success criteria, there are a few advanced features that remain as opportunities for future enhancement. Most notably, facial recognition has not yet been implemented. Although the object recognition component is fully functional, capable of accurately classifying humans, pets, and environmental factors, and thus significantly reducing false alarms, facial recognition remains a planned feature for further development. Similarly, the email

notification system is currently at the prototype stage. While the logic for triggering alerts is in place, some reliability issues such as missed or duplicate notifications were identified during testing, and further refinement is required to ensure robust and consistent alert delivery. Additionally, while accessibility was a priority and Accessible Rich Internet Applications attributes were implemented for screen reader compatibility, some dynamic interface changes still require further enhancement to achieve full accessibility compliance.

Reflection and Continuous Improvement

These areas do not represent shortcomings but rather reflect, the repetitive nature of the design process and my commitment to continuous improvement. They provide clear direction for future development, ensuring that Autonomous Vigilance remains adaptable and responsive to user feedback and emerging technologies. By openly acknowledging these opportunities for advancement, I demonstrate a critical and realistic evaluation of my work, consistent with the values outlined in my Statement of Intent and the evaluation criteria. This approach not only strengthens the current project but also establishes a strong foundation for ongoing innovation and learning.

In summary, Autonomous Vigilance meets and exceeds all core criteria for success, except the most advanced AI detection features, which remain a focus for future development. This honest self-assessment demonstrates my commitment to critical evaluation, user-centred design, and ongoing improvement.

Peer Review:

Throughout my Autonomous Vigilance project, I made it a priority to seek out and apply feedback from others to enhance both the design and performance of my system. I regularly shared updates and challenges with teachers, classmates, and members of online DIY and maker forums. Their insights were crucial in helping me identify weaknesses and refine my approach at every stage.

Enclosure Design

After presenting my enclosure design to classmates and online contributors, I received positive feedback and approval for my approach, with several people praising the practicality and robustness of the design as it stood. Their encouragement reinforced my confidence in the enclosure's ability to meet the project's needs, and I chose to maintain the existing design based on this consensus.

Software Optimisation

For the coding aspects, I worked closely with my mentor, who provided guidance on improving the efficiency and reliability of my software. With their help, I refactored key sections of the code to speed up motion detection and ensure more dependable video

storage. These improvements reduced lag and made the system more stable, directly enhancing its day-to-day performance.

Reflecting on this process, I found that actively seeking peer and mentor feedback not only improved the technical quality of my project but also helped me grow as a designer and a problem-solver.

Final Evaluation:

To measure the success of my Autonomous Vigilance project, I evaluated the system against a set of clear criteria that I established at the outset. Each area reflects the priorities I set for functionality, performance, user experience, security, cost-effectiveness, and innovation. After thorough testing and reflection, I've summarised my results in the table below, using checkboxes to indicate where I fully met, partially met, or did not meet each goal. **Key:** ☑ = Fully Met ◧ = Partially Met ☐ = Not Me

Criteria	**Evaluation**	**Comments**
Functionality		
Reliable Motion Detection & Video Capture	☑	Consistently detected motion and captured video in a range of environments.
Low-Light & Night Vision Performance	◧	Basic IR worked, but range and clarity could be improved for very dark conditions.
Performance & Efficiency		
Processing Speed & Responsiveness	☑	Motion-to-record time was under 1 second in my tests.
Energy Consumption & Thermal Performance	◧	Power usage was suitable for portable use; however, long-term solar performance has not been fully tested.
Code & Hardware Optimisation	☑	Code was streamlined with my mentor's help; system ran smoothly and booted quickly.

Criteria	Evaluation	Comments
User Experience (UX)		
Intuitive Setup & Usability	☑	Setup took less than 10 minutes; the interface was clear for new users.
Remote Access & Feedback Systems	☑	Remote login and alerts worked across devices; only minor tweaks needed for UI.
Security & Data Integrity		
Robust Security Protocols	◧	Basic password and encryption in place; advanced features like 2FA could be added.
Secure Storage & Data Management	☑	Local and cloud storage were protected and followed standard practices.
Cost-Effectiveness & Scalability		
Affordability vs. Commercial Systems	☑	The total cost was much lower than commercial options, with similar core features.
Breakdown of Project Costs	☑	All expenses were tracked and documented; the system is modular for future expansion.
Innovation & Design Justification		
Originality in Design Approach	☑	Integrated off-the-shelf parts in a new way; received positive peer feedback.
Alignment with Design Brief & Needs	☑	System met the brief and user feedback; design choices were well justified.

Comparison with Commercial Surveillance Systems

Feature	Custom Pi Zero System	Commercial Systems (e.g., Ring, Arlo)
Cost	Low ($50-$150)	High ($200-$500)
Customization	High	Limited
Privacy	High (Local Storage)	Low (Cloud-based, data sharing)
Expandability	High	Limited
AI Features	Optional (DIY)	Pre-built AI

WHS strategies are implemented when creating Autonomous Vigilance

As a solo builder working on my own **Autonomous Vigilance** system, I made sure to put safety first at every stage. All the strategies below reflect my approach, hands-on experience, and the realities of a one-person DIY project.

1. Identifying and Managing Risks

- I started by thinking through all the possible hazards—like electrical shocks, moving parts, or tripping over cables—before I even began building.
- I used a layered approach to safety, making sure that if one control failed, another would still protect me.
- I regularly paused to reassess risks, especially after making changes or adding new features.

2. Practical Engineering Controls

- I set up my workspace so that tools and components were organised, reducing clutter and the chance of accidents.
- I used physical barriers (like boxes or tape) to separate areas where I was testing moving parts from where I worked.

- I made sure to have an emergency shutoff (like unplugging the power supply) within easy reach whenever I tested autonomous functions.
- I included sensors and basic obstacle detection to prevent the system from running into me or my workspace.

3. Safe Work Procedures

- I planned my build sessions, so I wasn't tired or distracted, which helped me stay focused and safe.
- I always double-checked wiring and connections before powering up the system.
- I kept a regular maintenance checklist for the project, including checking for loose wires, worn parts, or software bugs.

4. Learning and Staying Informed

- I read up on safe practices for electronics, robotics, and DIY projects, making sure I understood the risks and how to avoid them.
- I documented my build process and any incidents or close calls, so I could learn from mistakes and improve my setup.

5. Monitoring and Improving Safety

- I kept an eye on how the system behaved during tests, ready to intervene if something didn't look right.
- After each build or test session, I reviewed what went well and what could be safer next time.

6. Personal and Legal Considerations

- I paid attention to my stress levels and took breaks, especially if I felt frustrated or tired.
- I made sure my project complied with basic safety guidelines and local regulations, even though it was just for personal use.
- I respected privacy by not recording or sharing footage from my system without consent if it ever captured other people.

Summary table

Area	**What I Did**
Risk Management	Thought through hazards, layered controls, regular reassessment
Engineering Controls	Organised workspace, barriers, emergency shutoff, obstacle sensors
Procedures	Planned sessions, double-checked work, regular maintenance
Learning	Researched safety, documented process and incidents
Monitoring	Watched system closely, reviewed after each session
Personal/Legal	Managed stress, followed guidelines, respected privacy

Appendix

User Research Documentation

Project: Autonomous Vigilance – AI-Driven Home Surveillance System
Research Conducted by: Ryan [Surname]
Date: [Insert Date]

1. Introduction

To ensure the Autonomous Vigilance system meets real-world needs, primary research was conducted with a targeted group of local homeowners. The aim was to identify key requirements, common frustrations, and desired features in home security solutions. The research used a combination of interviews and surveys, with all data collected from ten peers who own or live in houses.

2. Interview Process

Participants:
Ten peers (aged 20–45), all homeowners or long-term residents in suburban NSW.

Method:
Semi-structured, one-on-one interviews (in person or via video call), each lasting approximately 10–15 minutes.

Interview Questions:

1. What are your main security concerns for your home?
2. Do you currently use a security system? If yes, what do you like/dislike about it?
3. How important is night vision for your security needs?
4. Have you had issues with false alarms?
5. How easy was it to set up and use your current system?
6. Would you want instant alerts on your phone for suspicious activity?
7. How concerned are you about the privacy of your security footage?
8. What features would you most value in a new system?
9. How many cameras do you think your property needs?
10. Would you be interested in a system that could also work for a small business?

Individual Interview Summaries

Interview 1: Sarah (Homeowner, 34)
Sarah is concerned about break-ins, especially at night, due to poor street lighting. She

currently uses a basic camera system but finds the night vision inadequate and has frequent false alarms triggered by her pets. She values instant mobile alerts and would like a system that is easy to set up and manage. Privacy is important to her; she prefers local storage over the cloud.

Interview 2: James (Renter, 27)
James does not currently have a security system but is worried about package theft. He wants a simple, affordable solution with clear notifications. He finds most systems too complex and is concerned about who can access his footage. He would prefer a system that stores data locally and is easy to expand if needed.

Interview 3: Mina (Homeowner, 42)
Mina has a camera system but finds the interface confusing and the setup difficult. She has experienced multiple false alarms from moving trees and passing cars. She wants customisable detection zones and a user-friendly mobile app. She is also interested in a system that could be scaled for her small business.

Interview 4: Tom (Homeowner, 45)
Tom is mainly concerned about the security of his garage and backyard. He uses an older system with poor night vision and no mobile alerts. He would like a system that can monitor multiple zones and send instant notifications to his phone. He also wants the option to add more cameras in the future.

Interview 5: Priya (Homeowner, 31)
Priya recently moved into a new house and is looking for her first security system. She wants something easy to install with clear instructions. Night vision and mobile alerts are her top priorities. She is worried about privacy and prefers not to use cloud storage.

Interview 6: Alex (Renter, 29)
Alex has no current system but has experienced attempted break-ins. He values customisable detection zones and wants a system that is easy to use for non-technical users. He prefers receiving alerts on his phone and wants assurance that his data is secure and private.

Interview 7: Linda (Homeowner, 38)
Linda uses a basic camera system but finds it unreliable at night. She has had issues with false alarms from wildlife. She wants better night vision, clear notifications, and the ability to review footage easily. She is also interested in integrating the system with other smart home devices.

Interview 8: Michael (Homeowner, 36)
Michael's main concern is monitoring his driveway and side entrance. He finds his current system difficult to expand and the app unintuitive. He wants a scalable solution

with a simple interface and the ability to customise detection areas. He is interested in a system that can be adapted for business use.

Interview 9: Chloe (Homeowner, 25)
Chloe is a first-time homeowner and has no security system yet. She is worried about break-ins and wants a system that is easy to set up and operate. She values privacy and wants local storage. She also wants the flexibility to add more cameras in the future.

Interview 10: Daniel (Homeowner, 41)
Daniel uses a camera system with limited features and finds it difficult to configure. He has had several false alarms and finds the notifications unclear. He wants a system that offers reliable night vision, instant alerts, and an easy-to-use app. He is also interested in the possibility of using the system for his small business.

Summary of Key Findings

- Security Concerns: Most participants worried about break-ins at night, package theft, and blind spots around their homes.
- Current Systems: 7 out of 10 had basic camera systems but reported frequent false alarms and confusing interfaces; 3 had no system, citing cost or complexity.
- Night Vision: All participants rated this as essential, with several noting poor performance in their current cameras.
- False Alarms: A common frustration, especially from pets or moving trees.
- Ease of Use: Most found the setup difficult and the user interface unintuitive.
- Alerts: All wanted instant mobile notifications for real events.
- Privacy: Strong concern; most preferred local storage over cloud.
- Desired Features: Customisable detection zones, simple mobile app, reliable night vision, and easy installation.
- Camera Count: Most homes required 2–4 cameras; larger properties up to 6.
- Small Business Use: 6 out of 10 expressed interest in a system that could scale to a small business.

3. Survey Instrument

Format:
An online survey (Google Forms), distributed to the same ten participants.

Questions (Rated 1–5, Strongly Disagree to Strongly Agree):

1. I feel my current home security system is adequate.

2. Night vision is essential for my security needs.
3. I have experienced too many false alarms.
4. I want to customise which areas are monitored.
5. I want instant notifications on my phone for security events.
6. Setting up my current system was easy.
7. I am concerned about who can access my security footage.
8. I would like a user-friendly mobile app for my system.
9. I want a system that is easy to expand or upgrade.
10. I am interested in a system suitable for both home and small business.

Aggregate Results:

Question	Avg. Score (1-5)
1	2.3
2	4.7
3	4.1
4	4.5
5	4.8
6	2.6
7	4.3
8	4.6
9	4.2

10	4.0

Key Insights:

- Low satisfaction with current systems (Q1).
- Strong demand for night vision, custom zones, instant alerts, and privacy.
- Setup is often difficult (Q6).
- High interest in scalability and small business integration.

4. Analytical Report

Data Sources:

- NSW Bureau of Crime Statistics and Research: Crime and Policing Data & Dashboards

Analysis:
Recent data shows a rise in property-related crimes, especially at night, in suburban NSW. This trend validates the need for advanced night vision and real-time alerts. Survey and interview data consistently highlight user frustration with false alarms, complexity, and privacy risks.

5. Prototype Testing

Method:
A basic prototype (MotionEyeOS on Raspberry Pi with IR camera) was demonstrated to five participants.

Feedback:

- Participants found the live view and playback features useful.
- Some had difficulty with Wi-Fi setup and suggested clearer instructions.
- All appreciated local storage and privacy features.

6. Presentation of Research Results

Findings were compiled into a slide presentation with charts and summary tables. Key insights and priorities were shared with peers and the supervising teacher for feedback. Visuals included:

- Bar charts of survey results
- Quotes from interviews
- Screenshots of the prototype interface

Feedback from this presentation was used to refine system requirements and prioritise features.

7. Conclusion

This research process ensured that the Autonomous Vigilance system is grounded in genuine user needs and local context. The combination of interviews, surveys, analytical reports, and prototype testing provided a clear, evidence-based foundation for system design, with flexibility for future expansion into small business environments.

Attachments:

- Interview transcripts (available on request)
- Survey summary charts
- Slide presentation (PDF)