ÇANKAYA UNIVERSITY
FACULTY OF ENGINEERING
COMPUTER ENGINEERING DEPARTMENT

CENG 408

PROJECT REPORT

EDA NUR ALTUNOK 201911008

EKİN NALÇACI 202011030

GÜNALP GÜNGÖR 202011026

UTKU ÖZBEK 201911048

ZEYNEP POLAT 202011063

Contents

Introduction 7
Project Plan 8
LITERATURE REVİEW 9
Özet 9
Abstract 10
1. Introduction 11
2. Main Findings 12
2.1. Challenges and Difficulties in Human Detection and Counting 12
2.2. Computer Vision Based Approaches to Crowd Counting 14
3. Evaluating Model Accuracies 17
3.1. Unified Evaluation Protocol 19
4. Deployment Challenges and Solutions in Dockerized Object Detection Systems 20
5. Conclusion 21
SOFTWARE REQUIREMENT SPECIFICATION 22
1 Introduction 22
1.1. Purpose 22
1.2. Scope 23
2 General Description 24
2.1. Glossary 24
2.2. User Characteristics 25
2.3. Overview of Functional Requirements 27
2.3.1. Designing User Interfaces: 27
2.3.2. Overview of Functional Requirements: 27
2.4. General Constraints and Assumptions 28
2.4.1. Constraints 28
2.4.2. Assumptions 29
3 Specific Requirements 30
3.1. Interface Requirements 30
3.1.1. Account Creation 30
3.1.2. Camera Setup 30
3.1.3. Statistics Access 31
3.2. Detailed Description of Functional Requirements 31
3.2.1. User Authentication 31
3.2.2. Camera Configuration 32
3.2.3. Zone Definition 32
3.2.4. Object Counting and Tracking 32
3.2.5. Data Processing 32
3.2.6. Statistical Visualization 33
3.2.7. User Password Reset 33

3.2.8. Error Handling and Logging 33
3.3. Detailed Description of Non-Functional Requirements 33
3.3.1. Security 34
3.3.2. Performance 34
3.3.3. Fault Tolerance 34
3.3.4. Maintainability 34
3.3.5. Internationalization and Localization 35
3.3.6. Server System Capacity 35
3.3.7. Connection 35
3.3.8. Usability 35
3.3.9. Documentation 35
3.3.10. Data Accuracy 36
4 UML - Use Cases 37
4.1. INS-UC-001: Dashboard Integration 37
Preconditions: 38
Postconditions: - 38
Alternate Flow for INS-UC-001: 38
4.2. INS-UC-002: Server-Client Communication 39
Preconditions: 40
Postconditions: 40
Main Flow for INS-UC-002: 40
Alternative Flow for INS-UC-002: 41
4.3. INS-UC-003: AI-Driven Object Tracking 42
Preconditions: 43
Postconditions: 43
Main Flow for INS-UC-003: 43
Alternative Flow for INS-UC-003: 43
4.4. INS-UC-004: User Account Creation 44
Related Use Cases: - 44
Preconditions: 44
Postconditions: 45
Main Flow for INS-UC-004: 45
Alternative Flow for INS-UC-005: 45
4.5. INS-UC-005: Camera Configuration 46
Related Use Cases: - 46
Preconditions: 46
Postconditions: 47
Main Flow for INS-UC-005: 47
Alternative Flow for INS-UC-005: 47
5 DFDs Overview 48

5.1. Level 0 48
5.2. Level 1 49
6 Conclusion 50
SOFTWARE DESIGN DOCUMENT 51
1. Introduction 51
1.1. Purpose 51
1.2. Definitions 51
2. System Overview 52
3. System Design 53
3.1. Architectural Design 53
3.2. Decomposition Description 55
4. System Modeling (Activity & Sequence Diagrams) 56
5. Interface Design 61
5.1. Login 61
5.2. Dashboard 61
5.2.1. Example 1 61
5.2.2. Example 2 62
5.3. Account 62
5.4. Camera Configuration 63
5.4.1. Edit mode off 63
5.4.2. Edit mode on 63
5.4.3. “Draw Zone” modal 64
5.5. Settings and Logout Modal 64
5.5.1. Settings 64
5.5.2. Logout Modal 65
6. TEST PLAN 66
6.1. Introduction 66
6.1.1. Version Control 66
6.1.2. Overview 66
6.1.3. Scope 66
6.1.4. Terminology 66
6.2. Features To Be Tested 67
6.2.1. UI Interactions of the Application (UI) 67
6.2.2. Client-Server Communication (CS) 67
6.2.3. AI Driven Object Tracking (AI) 67
6.2.4. Security and Performance 67
6.3. Features Not To Be Tested 68
6.3.1. External System Integrations 68
6.3.2. Computers Without CUDA 68
6.3.3. Real-Time Multi-User Interaction 68

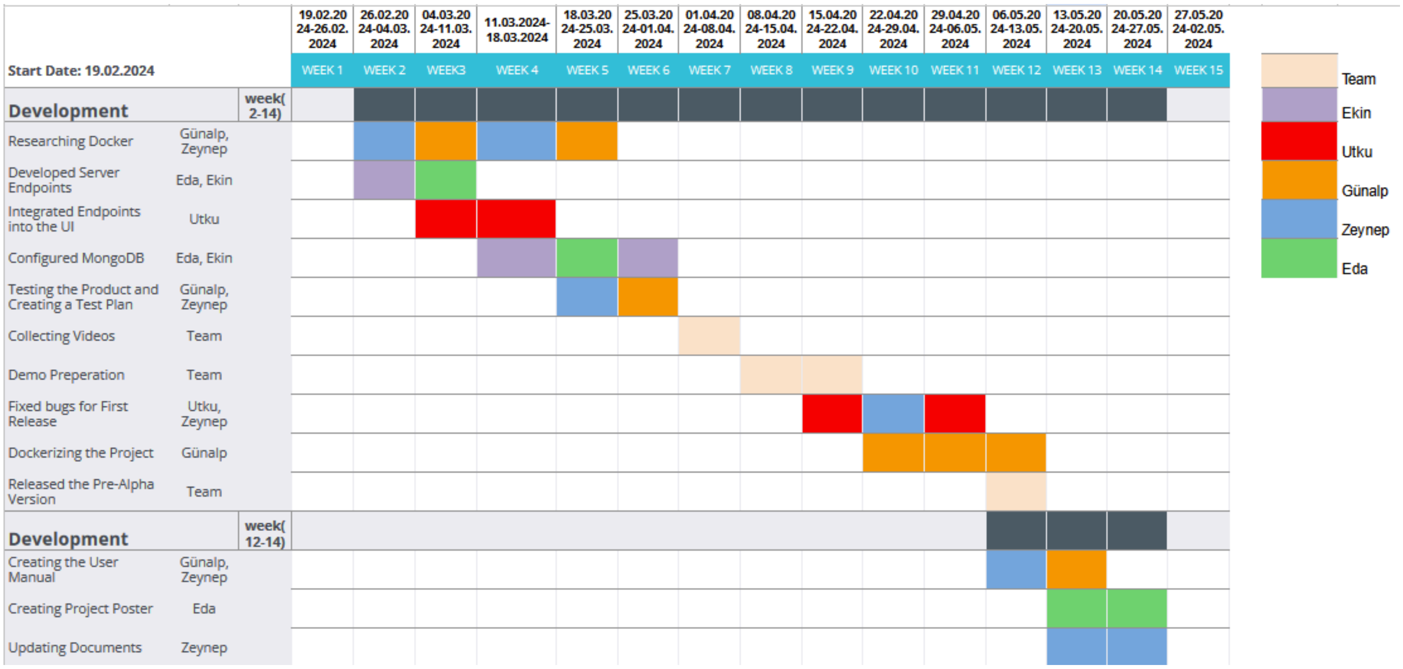
6.4. Item Pass/Fail Criteria........ 68
6.4.1. Exit Criteria........ 68
6.5. References........ 69
6.6. Test Design Specifications........ 69
6.6.1. UI Interactions of the Application........ 69
6.6.1.1. Subfeatures to be Tested........ 69
6.6.1.1.1. Account Edit (UI.AE)........ 69
6.6.1.1.2. Camera Setup (UI.CS)........ 69
6.6.1.1.3. Statistical Data (UI.SD)........ 69
6.6.1.2 Test Cases........ 69
6.6.2 Client-Server Communication........ 69
6.6.2.1. Subfeatures to be Tested........ 69
6.6.2.1.1 Camera Configuration (CS.CC)........ 70
6.6.2.1.2 Authenticate Users (CS.AU)........ 70
6.6.2.1.3 Account Operations (CS.AO)........ 70
6.6.2.2. Test Cases........ 70
6.6.3 AI Driven Object Tracking (AI)........ 70
6.6.3.1. Subfeatures to be Tested........ 70
6.6.3.1.1 Zone Definition (AI.ZD)........ 70
6.6.3.1.2 Object Counting and Tracking (AI.OCT)........ 70
6.6.3.1.3 Data Accuracy (AI.DA)........ 71
6.6.3.1.4 Real-Time Object Recognition (AI.RTOR)........ 71
6.6.3.2. Test Cases........ 71
6.6.4 Security and Performance........ 71
6.6.4.1. Subfeatures to be Tested........ 71
6.6.4.1.1 Response Time (SP.RT)........ 71
6.6.4.1.2 RAM Usage (SP.RAM)........ 72
6.6.4.1.3 Password Hashing (SP.PH)........ 72
6.6.4.1.4 Access Control (SP.AC)........ 72
6.6.4.1.5 Audit Logging (SP.AL)........ 72
6.6.4.2. Test Cases........ 72
6.7. Detailed Test Cases........ 73
6.7.1. UI.AE.01........ 73
6.7.2. UI.CS.01........ 74
6.7.3. UI.SD.01........ 75
6.7.4. CS.CC.01........ 76
6.7.5. CS.CC.02........ 77
6.7.6. CS.CC.03........ 78
6.7.7. CS.AU.01........ 79
6.7.8. CS.AU.02........ 80

6.7.9. CS.AU.03 81
6.7.10. CS.AO.01 82
6.7.11. AI.ZD.01 83
6.7.12. AI.OCT.01 84
6.7.13. AI.OCT.02 85
6.7.14. AI.DA.01 86
6.7.15. AI.TOR.01 87
6.7.16. SP.RT.01 88
6.7.17. SP.RT.02 89
6.7.18. SP.PH.01 90
6.7.19. SP.AC.01 91
6.7.20. SP.AL.01 92
7. RESULTS 93
8. USER MANUAL 95
8.1. Scope of Document 95
8.1.1. Release Dates 95
8.1.2 Glossary 95
8.2. Introduction 96
8.2.1. Requirements 96
8.2.1.1. Hardware Requirements 96
8.2.1.2. Software Requirements 96
8.3. Preparation for Installing the Application 96
8.4. Getting Started 97
8.5. Using Insightio 101
8.5.1 Login 101
8.5.2 Main Dashboard 102
8.5.3 Select Time Frame 102
8.5.4 Select Target 103
8.5.5 Account Page 103
8.5.6 Configure Camera settings 104
8.5.6.1 Choose Targets 105
8.5.6.2 Choose Draw Zones 105
8.5.6.3 Add/Edit Camera 106
8.5.10. Settings 107
9. Conclusion 108
10. References 109

Introduction

This document encompasses a literature review, Software Requirements Specification (SRS), Software Design Document (SDD), Test Plan, Test Results and User Manuel for the Insightio platform. The literature review provides a summary of existing information about the advanced features and use cases of the platform. Subsequently, the SRS section elucidates the purpose, scope, and general requirements of Insightio. Following the SRS, the SDD intricately presents the architecture, functionality, and design details of the platform from a user perspective. By step-by-step narration of the project's overall structure and content, this document provides a comprehensive overview of the Insightio platform.

Project Plan



			19.02.2024-26.02.2024	26.02.2024-04.03.2024	04.03.2024-11.03.2024	11.03.2024-18.03.2024	18.03.2024-25.03.2024	25.03.2024-01.04.2024	01.04.2024-08.04.2024	08.04.2024-15.04.2024	15.04.2024-22.04.2024	22.04.2024-29.04.2024	29.04.2024-06.05.2024	06.05.2024-13.05.2024	13.05.2024-20.05.2024	20.05.2024-27.05.2024	27.05.2024-02.05.2024
Start Date: 19.02.2024			WEEK 1	WEEK 2	WEEK3	WEEK 4	WEEK 5	WEEK 6	WEEK 7	WEEK 8	WEEK 9	WEEK 10	WEEK 11	WEEK 12	WEEK 13	WEEK 14	WEEK 15
Development		week(2-14)		■	■	■	■	■	■	■	■	■	■	■	■	■	
Researching Docker	Günalp, Zeynep			🟦	🟧	🟦	🟧										
Developed Server Endpoints	Eda, Ekin			🟪	🟩												
Integrated Endpoints into the UI	Utku				🟥	🟥											
Configured MongoDB	Eda, Ekin					🟪	🟩	🟪									
Testing the Product and Creating a Test Plan	Günalp, Zeynep						🟦	🟧									
Collecting Videos	Team								▨								
Demo Preperation	Team									▨	▨						
Fixed bugs for First Release	Utku, Zeynep										🟥	🟦	🟥				
Dockerizing the Project	Günalp											🟧	🟧	🟧			
Released the Pre-Alpha Version	Team													▨			
Development		week(12-14)												■	■	■	
Creating the User Manual	Günalp, Zeynep													🟦	🟧		
Creating Project Poster	Eda														🟩	🟩	
Updating Documents	Zeynep														🟦	🟦	

▨	Team
🟪	Ekin
🟥	Utku
🟧	Günalp
🟦	Zeynep
🟩	Eda

LITERATURE REVIEW

Özet

Bu araştırma, perakende sektöründen acil tahliye senaryolarına kadar farklı alanlarda, insan varlığını etkili bir şekilde kontrol etmek amacıyla kullanılan insan sayım sistemlerinin giderek artan önemini ele almaktadır. Bu makale, kalabalık alanlarda bireyleri doğru bir şekilde belirlemenin ve nicelendirmenin zorluklarına odaklanmakta ve bunun için aydınlatma, hava koşulları ve görüntü kalitesi gibi faktörleri dikkate almaktadır. Ayrıca, daha geleneksel yoğunluk regresyon tekniklerini yerine koyan daha sofistike bilgisayar görüsü tabanlı stratejilerin gelişimini göstermektedir. Farklı çalışma hedeflerinde homojen değerlendirmeler sağlamak için standartlaştırılmış değerlendirme yöntemlerinin önemini vurgulamaktadır. Bu analiz, kalabalık sayımın çağdaş operasyonel ve güvenlik önlemlerinde oynadığı hayati rolü vurgulayarak, bu alandaki başarıları ve zorlukları kapsamlı bir şekilde sunmaktadır.

Abstract

This study of research investigates the expanding importance of people counting systems in efficiently controlling human presence in a variety of contexts, from retail to emergency evacuation situations. The paper discusses the challenges of precisely identifying and quantifying individuals in crowded locations, taking into account elements like lighting, weather, and image quality. It also illustrates the development of computer vision-based strategies, showing how more sophisticated approaches replaced more conventional density regression techniques. In order to provide uniform assessments across various study objectives, the analysis underlines the need for standardized evaluation techniques. In summary, this analysis highlights the critical role that crowd counting plays in contemporary operational and safety measures by offering a comprehensive overview of the field's accomplishments and obstacles.

1. Introduction

In today’s global landscape, grappling with the complexities of managing human presence has become increasingly challenging due to the ever-expanding population. Businesses, public spaces, and communities seeking to efficiently navigate and analyze crowded environments require robust tools for data acquisition and manipulation. The global market for people counting systems reached a valuation of 969.9 million USD in 2021, with a projected annual growth rate of 12.2% from 2022 to 2030 (People Counting System Market Size, Share & Trends Analysis Report by End-use (BFSI, Corporate), by Offering (Hardware, Software), by Mounting Platform (Ceiling, Wall), by Type, by Technology, and Segment Forecasts, 2022 - 2030, n.d.).

Importantly, people counting methods are essential in many industries, retail being one of the best examples. The transition from manual counting to sophisticated computer solutions has been crucial for retail organizations seeking to modernize operations and improve consumer experiences. According to Chakraborty (2023), these systems provide real-time analysis, identify high-traffic areas, optimize staffing, examine consumer behavior, and facilitate forecasting.

Real-time tracking also extends its reach to emergency evacuations, where human-centered sensing proves invaluable for monitoring crowd behavior in real-time, ultimately bolstering incident management strategies (Crowd Models for Emergency Evacuation: A Review Targeting Human-Centered Sensing, 2013).

These statistics underscore the profound impact of people counting technology across diverse industries. Evidently, businesses and organizations are increasingly embracing this technology to enhance operational efficiency, elevate security measures, and foster data-informed decision- making processes.

2. Main Findings

The literature review findings are bifurcated into two distinct parts. The first part scrutinizes the challenges entailed in accurately discerning individuals within crowded spaces. This encompasses considerations such as variations in lighting, weather, and image quality, in addition to complexities arising from overlapping individuals and occluded areas.

The following section then explores the development of computer vision-based approaches. The noteworthy advancements in computational approaches and their usefulness in crowd counting are emphasized. This includes talking about methods such as density regression and dense detection and how they can be used to improve the accuracy and efficiency of this field. All of these studies provide an extensive understanding of the basic concepts and recent developments in crowd counting.

2.1. Challenges and Difficulties in Human Detection and Counting

Accurately identifying and counting people in congested spaces is a complex task with many moving parts. External factors that introduce unpredictability and can have a major impact on the accuracy of human detection include variations in lighting, weather, and image quality (Raghavachari et al., 2015). Accurate counting is made more difficult by overlapping people and occluded areas, necessitating complex discerning techniques (Sam et al., 2020).

Public spaces, characterized by dynamic crowd densities, pose another layer of complexity. Maintaining consistent accuracy becomes a formidable challenge as these spaces transition from peak to off-peak crowd conditions (Raghavachari et al., 2015). Additionally, the choice of camera orientation is critical in determining detection efficacy, with different angles yielding distinct perspectives on human objects, each presenting unique challenges (Sam et al., 2020).

Unfavorable circumstances, such as poor lighting and camera glare, can cause problems like visual blurriness, which makes it harder for people to perceive objects of interest (Bhangale et al., 2020). The already difficult task of precise counting is made more problematic by these environmental influences. Researchers use sophisticated computer vision techniques and machine learning algorithms in response. For example, deep learning models have the potential to improve human detection accuracy in a variety of circumstances (Bhangale et al., 2020). To handle particular issues like occlusion and different crowd sizes, complementary techniques are also suggested. These include trajectory clustering, feature-based regression, and individual pedestrian detection (Huang & Chung, 2004). These group efforts serve as an example of the continuous struggle to overcome the difficulties associated with human identification and counting in various dynamic environments.

2.2. Computer Vision Based Approaches to Crowd Counting

Crowd counting has witnessed a notable evolution in computer vision algorithms. Initially, density regression techniques were employed, estimating crowd density maps to infer counts (Raghavachari et al., 2015). However, this approach faced limitations, particularly in scenarios with dense crowds (Sam et al., 2020). To address this, recent advancements have shifted towards dense detection methods, which present more practical and accurate solutions (Sam et al., 2020). Notably, deep learning models, including Convolutional Neural Networks (CNNs) like VGG andResNet, have played a crucial role in this transition.

According to Bhangale et al. (2020), these models excel in feature extraction and pattern recognition, rendering them well-suited for crowd counting tasks. Moreover, they have been adeptly adapted to handle the intricacies of crowd analysis, effectively capturing complex spatial arrangements and density variations (Sam et al., 2020; Sjöberg & Hyberg, 2023; Hussain, 2023).This adaptability showcases the versatility of CNNs in addressing the challenges posed by dynamic crowd environments.

In addition to CNNs, YOLO (You Only Look Once) models have emerged as powerful tools in object detection, including humans, and have been particularly effective in crowd counting tasks (Hussain, 2023). These models utilize a single neural network to simultaneously predict bounding boxes and class probabilities for multiple objects in a single pass. This makes them well-suited for real-time processing and applications like object detection and tracking in video streams. YOLO models, including the latest YOLO-v8 release, have demonstrated high- classification performance and fast detection capabilities, making them increasingly relevant in industrial settings (Hussain, 2023).

Additionally, techniques like multi-scale analysis have proved indispensable, given that crowd images often encompass individuals of varying sizes (Huang & Chung, 2004; Hussain, 2023). Methods such as image pyramid decomposition and feature pyramids facilitate the analysis of images at different resolutions. This approach ensures that individuals are detected and counted irrespective of their scale, thereby bolstering the overall robustness of the counting system (Huang & Chung, 2004; Hussain, 2023). This emphasis on multi-scale analysis underscores its pivotal role in accurately estimating crowd counts, especially in scenarios with diverse crowd compositions.

Furthermore, the integration of detection and tracking systems has demonstrated remarkable efficacy in dynamic environments. As suggested by Huang and Chung (2004) and Sjöberg and Hyberg (2023), detection algorithms identify individuals, while tracking algorithms ensure object continuity across frames. This synergistic approach significantly enhances crowd counting reliability, particularly in situations characterized by occlusions and dynamic movement (Sjöberg & Hyberg, 2023; Hussain, 2023). The fusion of detection and tracking mechanisms addresses challenges associated with the inherent movement and occlusions within crowd scenes.

Some other methods have also significantly contributed to the evolution of crowd counting algorithms. For instance, CSRNet, a CNN tailored for congested scene recognition, has emerged as a prominent tool in crowd counting applications (Bhangale et al., 2020). This model employs a multi-column architecture in conjunction with top-down modulation, allowing for the precise detection and localization of individuals within densely populated scenes. Notably, CSRNet's proficiency in generating accurate crowd density maps from point annotations underscores its robustness in crowd counting scenarios.

In addition to CSRNet, the integration of advanced image descriptors like Perspective Invariant Histograms of Oriented Gradients (HOGp) has revolutionized crowd counting, eliminating the need for camera calibration (Reis, 2014). This innovative technique not only addresses privacy concerns but also proves particularly adept in systems with limited resources. Consequently, it stands as a valuable tool in urban planning and crowd control applications. By emphasizing these image descriptors, this approach offers an alternative avenue to achieve accurate crowd counting.

Furthermore, the application of Hidden Markov Models (HMM) shows promise in recognizing various human dynamics, including walking and sitting (Huang & Chung, 2004). HMMs provide a robust framework for understanding temporal dependencies within crowd behavior, introducing an additional layer of sophistication to crowd counting algorithms (Huang & Chung, 2004). The fusion of HMMs with other detection and tracking methodologies presents a holistic approach to crowd analysis.

In summary, the landscape of crowd counting has witnessed a transition from traditional density regression techniques to the adoption of dense detection strategies. While deep learning models like VGG, ResNet, and YOLO have played pivotal roles, it is imperative to recognize the substantial contributions of other noteworthy models like CSRNet, innovative techniques like HOGp descriptors, and the integration of HMMs. These diverse methods collectively underscore the versatility and adaptability of computer vision in achieving accurate crowd estimates across various domains, ranging from smart building management to urban planning and public safety.

3. Evaluating Model Accuracies

Precisely evaluating crowd counting models' performance is a complex process that depends on a number of variables and methodological techniques. The assessment procedure depends on the particular approach taken. When it comes to urban areas that use the HOGp approach (Reis, 2014), comparing models that integrate HOGp with those that do not provide a solid framework for measuring accuracy when it comes to counting disparities. This method clearly illustrates the usefulness and effectiveness of the idea by reducing counting errors without requiring camera calibration.

For evaluating the accuracy of YOLOv8 in pedestrian detection (Sjöberg & Hyberg, 2023), mean Average Precision at 50 (mAP50) scores serve as a pivotal metric, illuminating the nuanced interplay between accuracy and model performance across diverse lighting conditions. This assessment provides critical insights into the model's viability for applications such as self- driving vehicles, while also highlighting areas for refinement, particularly under varying illumination scenarios.

The contour-based matching method for human detection (Rahman, 2017) relies on metrics like average accuracy and precision in still images to gauge accuracy. This approach acknowledges the trade-offs inherent to still-image analysis, especially in scenarios with overlapping human entities, emphasizing the method's efficacy within these constraints.

In the context of agricultural pest and disease detection (Zhang, Ding, Li, & Li, 2023), the focus of accuracy evaluation lies in model performance metrics encompassing detection accuracy and recognition. This assessment highlights improvements in model precision, indicating concrete strides in the field of agricultural pest and disease detection.

Metrics such as mean absolute error (MAE) and mean square error (MSE) are used by real-time crowd counting systems (Bhangale et al., 2020) to evaluate accuracy. This approach highlights the model's higher accuracy compared to traditional methods, confirming its usefulness for crowd counting applications in various real-time settings.

For YOLO models (Mokayed et al., 2022), accuracy percentages stand as critical metrics in the evaluation process, drawing attention to the intricate balance between processing speed and model accuracy. This assessment paradigm highlights the trade-offs that must be considered when deploying YOLO models in applications where both speed and accuracy are paramount.

3.1. Unified Evaluation Protocol

To guarantee consistent model assessment across various situations, datasets, and research goals, a unified evaluation protocol is necessary. But its foundation is hindered by differences in datasets, contextual differences, and the quick development of approaches. Because of these elements, developing uniform evaluation frameworks is both necessary and challenging. The scientific community is working to create more standardized standards in response to this need.

The lack of a single, universally applicable evaluation process results from the many different uses of computer vision and machine learning, each with its own set of objectives for research. The creation of a uniform evaluation framework is further complicated by differences in dataset quantity, quality, and annotations. Because of these differences, developing an assessment process that stays up to date in the face of deep learning and computer vision techniques' rapid advancement is difficult.

Despite these challenges, there is a growing recognition within the research community of the necessity for unified evaluation protocols. This collective effort signifies a commitment to enhancing the rigor and comparability of evaluations in the field. As the field continues to advance, establishing standardized evaluation protocols remains a crucial area of focus to ensure impartial and consistent assessments across various research endeavors.

4. Deployment Challenges and Solutions in Dockerized Object Detection Systems

Docker plays a key role in enabling the uniform deployment of object detection systems across various computing environments. On the other hand, setting up a Docker environment that works with GPU-accelerated applications requires matching up compatible NVIDIA drivers and PyTorch frameworks with CUDA versions. The constant updates and compatibility problems that come with quickly developing technology make this task more difficult. Docker makes it easier to deploy applications with multiple components by offering environments that are isolated and consistent, which is essential for scalability and reproducibility.

The need for a structured deployment strategy is further highlighted by the complexity of troubleshooting issues like backend connectivity and CUDA initialization errors. In spite of these obstacles, the scientific community is beginning to acknowledge Docker's value in expediting the deployment process across heterogeneous infrastructures. This acknowledgment drives the ongoing effort to enhance deployment methodologies, ensuring efficient and reliable setup of computer vision systems.

5. Conclusion

The culmination of our research and literature review unequivocally underscores the substantial potential of pre-trained networks, particularly YOLO-based models, in pivotal domains of visual processing such as object detection, head counting, and human counting. These studies shed lighton the instrumental role such technologies can play in real-world applications. Real-time object detection and counting stand out as paramount in domains like security, surveillance, and crowd management, where the algorithms driving object tracking and detection form the cornerstone of effectiveness.

Beyond mere head counting, density estimation presents an opportunity for a more nuancedanalysis of visual data. However, for research in this domain to progress, reliance on larger, high-resolution datasets and standardized evaluation protocols is imperative to ensure both accuracy and practicality. The imperative for faster and more precise real-time counting methods in various applications remains a critical requirement. Studies comparing different algorithms illuminate the robustness and adaptability of methods like HoG in diverse scenarios. While these methods demonstrate promise, it is essential to address challenges, such as varying lighting conditions, through further research.

In summary, our reviews and research studies unveil numerous opportunities and potential for advancement in the fields of object detection, human counting, and density estimation. The application of standardized evaluation protocols and specialized algorithms serves as a catalyst for further progress in these domains. It is clear that these research contributions wield a significant impact on crucial areas like real-time applications, security, and crowd management. The ongoing collaborative efforts in the research community to establish unified evaluation protocols signal a commitment to enhance the rigor and comparability of evaluations in the field. As the field continues its forward momentum, the establishment of standardized evaluation protocols remains a pivotal area of focus, ensuring impartial and consistent assessments across diverse research endeavors.

SOFTWARE REQUIREMENT SPECIFICATION

1 Introduction

The Insightio platform provides crowd/object counting and tracking solutions for businesses. Users have the capability to establish an account, set up cameras, and define the classes of tracked objects. In this section, Insightio is described in broad strokes.

1.1. Purpose

This SRS is about the Insightio platform, which provides solutions for crowd/object counting and tracking to businesses. In 1. Introduction section, we aimed to give a brief information about the purpose and scope of our system. The 2. General Description section splits into four parts: 2.1 Glossary, include any specialized jargon and terminology used by the system, 2.2 User Characteristics, explains the features of the users, 2.3 Overview of Functional Requirements, describes the functions briefly which are used in the system, 2.4 General Constraints and Assumptions, the system's limits and expectations are defined. The following section, 3. Specific Requirements, comprises three units:
3.1 Interface Requirements, which delineates features for the user interface, such as account setup, camera configuration, and access to statistical data. 3.2 Detailed Function Requirement describes functions with specifications designed based on test-ability principles. 3.3 Non-Functional Requirements elucidate performance and hardware type requirements. The 4. Analysis UML section is composed of two parts: 4.1 Use Cases : This section elaborates in detail on how users will interact with the system and carry out tasks. 4.2 Functional Modeling (DFD): This section, demonstrating the system's boundaries through data flow diagrams, assists us in comprehending the functioning of the system. Additionally, it clarifies the relationships between objects through a data dictionary. The 5. References section encompasses the resources used in the document, including literature, documents, and source materials referenced during analysis and design.

1.2. Scope

The Insightio platform offers crowd/object counting and tracking solutions for businesses. When businesses need periodic statistics on specific object counts, they can utilize the Insightio platform.

A business looking to use the platform must first create an account and purchase a suitable license for commercial use. Once the business has an account on the platform, they can begin their operations by adding cameras and specifying the object classes they want to track. Optionally, they can limit counting in certain areas on camera images.

Once a customer defines and configures cameras on the platform, they can access statistics related to the previously determined object counts. When camera configurations are complete, Insightio performs the real counting and tracking of objects. The data on the number of objects captured by the platform is processed to create visualizations, such as graphs and drawings, presenting statistics based on the preferred time frame (day, week, month, year, etc.) for the user.

2 General Description

This section provides a foundational overview of Insightio, starting with a glossary to define essential terms. It then outlines the characteristics of users before delving into the system's functional requirements, emphasizing user interface design. The subsequent exploration encompasses general constraints and assumptions that shape Insightio's operational context.

2.1. Glossary

User	Individuals who use and interact with the application for various purposes.
Actor	Any entity, including users or other software systems that interact with the primary system.
Object Detection	A computer vision technique that enables systems to identify objects in digital images, videos, and visual inputs.
YOLO Models	YOLO Models are a type of object detection algorithm used in computer vision.
Javascript	A scripting language commonly used to enhance the interactivity and functionality of web pages.
SvelteJS	A JavaScript framework used to build user interfaces.
Kotlin	A programming language often used for developing Android applications.
Python	A versatile programming language widely used for various applications.

2.2. User Characteristics

Understanding the diverse characteristics of the intended user groups for the Insightio platform is paramount for designing an interface and functionalities that align with their needs and capabilities. The following user categories are suggested to represent a varied user base:

- Business Owners and Administrators: Business owners and administrators exhibit diverse educational backgrounds, ranging from business management to technical expertise. With substantial industry experience, their focus is on strategic decision-making. Their moderate to high technical proficiency enables them to comprehend the business implications of technical features.
- Operations Managers: Operations managers bring varied educational backgrounds, including operations management and technology. With substantial experience in day-to-day operations management, they possess moderate technical proficiency and are comfortable with the operational aspects of the platform.

- Security Personnel: Security personnel encompass diverse educational backgrounds, emphasizing security and emergency management. With substantial experience in security operations, they exhibit moderate to high technical proficiency, particularly in using real-time tracking features.

- Data Analysts: Data analysts typically hold degrees in data science, statistics, or related fields. With a strong analytical background and experience in interpreting statistical visualizations, they demonstrate high technical proficiency for in-depth data analysis.

- General Users: General users have varied educational backgrounds, including non-technical fields. With limited to moderate experience in data analysis tools, they exhibit limited technical proficiency and prefer user-friendly interfaces.

- Technical Support Staff: Technical support staff possess technical degrees or certifications related to IT support. With extensive experience in providing technical support, they have high technical proficiency and are capable of troubleshooting.

2.3. Overview of Functional Requirements

With solutions for crowd and object monitoring, Insightio is a strong platform that meets the counting and tracking needs of businesses. The technical backbone of the platform is composed of a server that directly communicates with object tracking software.

2.3.1. Designing User Interfaces:

- Account Creation: Simplified procedure for opening an account.
- Camera Setup: Tracking zones and object classes are intuitively configured.
- Statistics Access: User-friendly presentation of statistical data through charts.

2.3.2. Overview of Functional Requirements:

- User authentication: Strong security to allow users to log in.
- Camera Configuration: An easy-to-use configuration for both object tracking and cameras.
- Zone Definition: Creating polygonal zones for accurate tracking.
- Object Counting and Tracking: Reliable tracking is achieved with server-side scripts.
- Data processing: Optimized server-side operations to ensure effective data transfer.
- Object count statistics are visualized statistically using dynamic charts.
- User Password Reset: Safe methods for changing passwords.
- Error handling and logging: all-inclusive troubleshooting systems.

2.4. General Constraints and Assumptions

Insightio operates within a framework of constraints and assumptions that influence its functionality and user interactions. Understanding these factors is crucial for a comprehensive perspective on the platform's capabilities.

2.4.1. Constraints

- Technology Compatibility: Supervision and YOLO models are integrated with Javascript, SvelteJS, Kotlin, and Python to form the foundation of Insightio. The compatibility of these technologies may impose limitations or restrictions that could affect the platform's functionality.

- Hardware Requirements: The object detection features of Insightio require GPUs to support CUDA, so it must be installed on hardware that meets the requirements. Lack of CUDA support can severely impair functionality and performance, reducing operational effectiveness.

- Network Dependency: In order to ensure smooth data transfer between the server and frontend, the platform relies on a dependable network connection. The processing and visualization of data in real time may be impacted by any network connectivity issues or disruptions.

- User Compliance: It is anticipated that users will abide by system guidelines. Access to specific features or services may be restricted in the event of non- compliance.

- Security Regulations: User authentication, data transmission, and system integrity all depend on adherence to security measures. Violating security protocols may compromise user data integrity and confidentiality.

2.4.2. Assumptions

- User Environment: It is anticipated that users will use common web browsers with JavaScript enabled to access Insightio. Any modifications to this assumed user environment could have an effect on the platform's usability and functionality.
- Email Reliability: Users' access to a dependable email service is assumed by the email verification process. Email delivery issues or delays might cause problems with the registration and verification procedures.

- Camera Functionality: Insightio makes the assumption that the platform's cameras are operational and able to take the desired pictures. Accurate object counting and tracking may be impacted by any hardware issues or camera capacity restrictions.

- Server Reliability: The platform counts on the client-side scripts that track and count objects to run consistently and dependably. Real-time tracking and data processing may be momentarily hampered by server outages or performance problems.

- Data Integrity: The accuracy and integrity of the data depends on the cameras being configured correctly and the object counting algorithms operating as meant. Inconsistencies in statistical visualization can result from deviations.

- Legal Compliance: It is expected of users to follow the laws governing the use of recording devices and the privacy of personal information. Legal violations and illegal use of the platform might be outside of Insightio's responsibility.

3 Specific Requirements

Insightio is a comprehensive platform designed to address the counting and tracking needs of businesses, offering solutions for both crowd and object monitoring. This section outlines the specific requirements that govern the functionality and user interface of the Insightio platform. The following subsections detail the interface requirements and provide a comprehensive description of the functional and non-functional requirements, each formulated to be precise, testable, and directly aligned with the project's objectives.

3.1. Interface Requirements

The interface of Insightio is designed to provide an intuitive and user-friendly experience for businesses seeking crowd and object counting solutions. The specifications for the user interface, such as account setup, camera configuration, and statistical data access, are described in this section.

3.1.1. Account Creation

- The system shall provide a user registration page with fields for username, email, and password.
- System must allow passwords to be at least 8 characters long and to include at least one uppercase, one lowercase, one special character and one number.
- Upon successful registration, the system shall send a confirmation email to the provided email address for account verification.
- The user interface shall include a login page with fields for username and password.

3.1.2. Camera Setup

- The system shall provide a dashboard for users to add and configure cameras.
- Users shall be able to specify the class of objects they intend to track for each camera.
- Optionally, users may define polygonal zones on camera images to restrict counting to specific areas.

3.1.3. Statistics Access

- Upon successful camera configuration, users shall have access to statistical information related to the tracked subject such as average population, rush and non- peek hours.
- The platform shall offer an intuitive interface for users to view and interact with the generated charts and plots.

3.2. Detailed Description of Functional Requirements

The particular functional requirements that specify how the Insightio platform behaves are described in this section. Each functional requirement listed here is designed with the ideas of precision and testability in mind.

3.2.1. User Authentication

- The system shall authenticate users based on their provided username and password.
- System must log every login attempt for observability and reliability.
- Failed login attempts shall trigger appropriate error messages for incorrect credentials.
- Successful authentication shall grant users access to their personalized dashboard.

3.2.2. Camera Configuration

- Users shall be able to add new cameras to their account, providing necessary details such as camera name and IP address.
- For each camera, users shall be able to specify the class of objects they wish to track.

3.2.3. Zone Definition

- Optionally, users may define polygonal zones on camera images to restrict counting to particular areas of interest.
- The system shall provide an intuitive interface for users to draw and modify these zones.

3.2.4. Object Counting and Tracking

- Once cameras are configured, a server-side script shall execute to perform object counting and tracking.
- The script shall run concurrently for each active camera, relaying gathered information to the server.

3.2.5. Data Processing

- The server shall process the received data from active cameras for transmission to the platform's frontend.
- Data processing shall include aggregation and formatting of object count information.

3.2.6. Statistical Visualization

- The platform shall generate charts and plots based on the received object count data.
- Users shall have the ability to select specific time periods (day, week, month, year, etc.) for visualization.

3.2.7. User Password Reset

- The system shall allow users to reset their password by providing their registered email address.
- A password reset link shall be sent to the user's email for security verification.

3.2.8. Error Handling and Logging

- The platform shall implement robust error handling mechanisms to gracefully manage unexpected events.
- Detailed logs shall be maintained to track system behavior and aid in debugging processes.

3.3. Detailed Description of Non-Functional Requirements

This section outlines the non-functional requirements essential for the effective implementation and performance of Insightio. Non-functional requirements encompass aspects such as system reliability, performance, security, and user experience. The specifications in this section provide a foundation for evaluating and ensuring the overall quality and effectiveness of the Insightio platform.

3.3.1. Security

- The system must guarantee data transmission that is both secure and protected from threats.
- The system must protect user data privacy and forbid unauthorized access.
- The platform should have multi-factored authentication such as email verification to improve security.
- Logs should be used to monitor actions of users, system events and modifications, ensuring auditing.

3.3.2. Performance

- The system must operate efficiently with little lag time at 30 ms maximum.
- Response times for tracking should not vary by more than 20% during peak and off-peak hours.

3.3.3. Fault Tolerance

- The system should include redundancy planning to make sure backup systems to take over in cases where core components break down.
- Services must be designed to be isolated from each other, preventing the spread of faults and impacting functionality of other components.

3.3.4. Maintainability

- The system should evolve to meet the changing needs such as the subject being tracked.
- The system should be able to scale horizontally to work with an increasing number of cameras and users.
- Software should be designed such that it optimizes utilization of resources such as CPU load and memory consumption to ensure efficient performance.
- Software should be able to integrate with other components such as third party tools.

3.3.5. Internationalization and Localization

- The system should be easily adaptable to different languages and cultural preferences in cases where in the future, users of different regions can benefit from the usage of the application.

3.3.6. Server System Capacity

- The server infrastructure, including CPU, GPU and storage, must be capable of handling the computational load generated by real-time tracking and data processing.

3.3.7. Connection

- Connection must remain continuous with cameras to ensure real-time tracking, it should automatically attempt to reconnect in cases where connection is lost.

3.3.8. Usability

- User interfaces must be designed such that it's easy to use, accessible and consistent by users. Software should be tested thoroughly in order to ensure user experience is seamless and useful.

3.3.9. Documentation

- Documentation should be provided to end users demonstrating the usage and the benefits of the system.
- Technical documentation should also be included for developers such as API documentation.

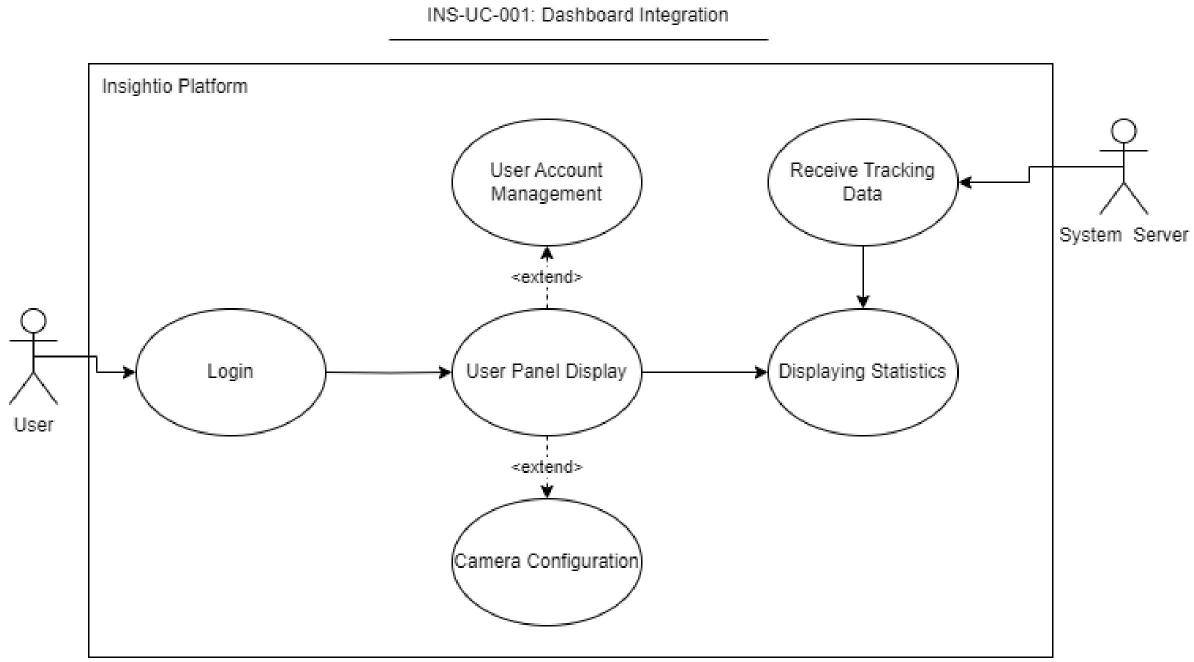
3.3.10. Data Accuracy

- The tracking algorithm must have an accuracy rate of at least 90%, and statistical reports created by the system should include precise and up-to-date information.

4 UML - Use Cases

Use cases in Insightio represent essential functionalities that cater to the diverse needs of its users. Each use case encapsulates a specific interaction scenario within the platform, contributing to its overall functionality.

4.1. INS-UC-001: Dashboard Integration



Use Case Name: INS-UC-001

Related Use Cases: INS-UC-005

Actors: User

Description: Utilizing the Insightio platform enables seamless crowd and object counting, providing businesses with valuable statistics and insights. The process begins with creating an account and obtaining a commercial license. After registration, businesses configure cameras and optionally define restricted counting areas. Insightio then takes control of real-time counting and tracking.

The platform offers a user-friendly interface, providing access to comprehensive statistics and presenting graphs and drawings based on the user-defined time intervals. This allows businesses to make data-driven decisions, optimize operations, and gain effective insights into object counts within their operational environments.

Preconditions:

- User must have internet connection
- User must have an account and must login

Postconditions: -

Main Flow for INS-UC-001:

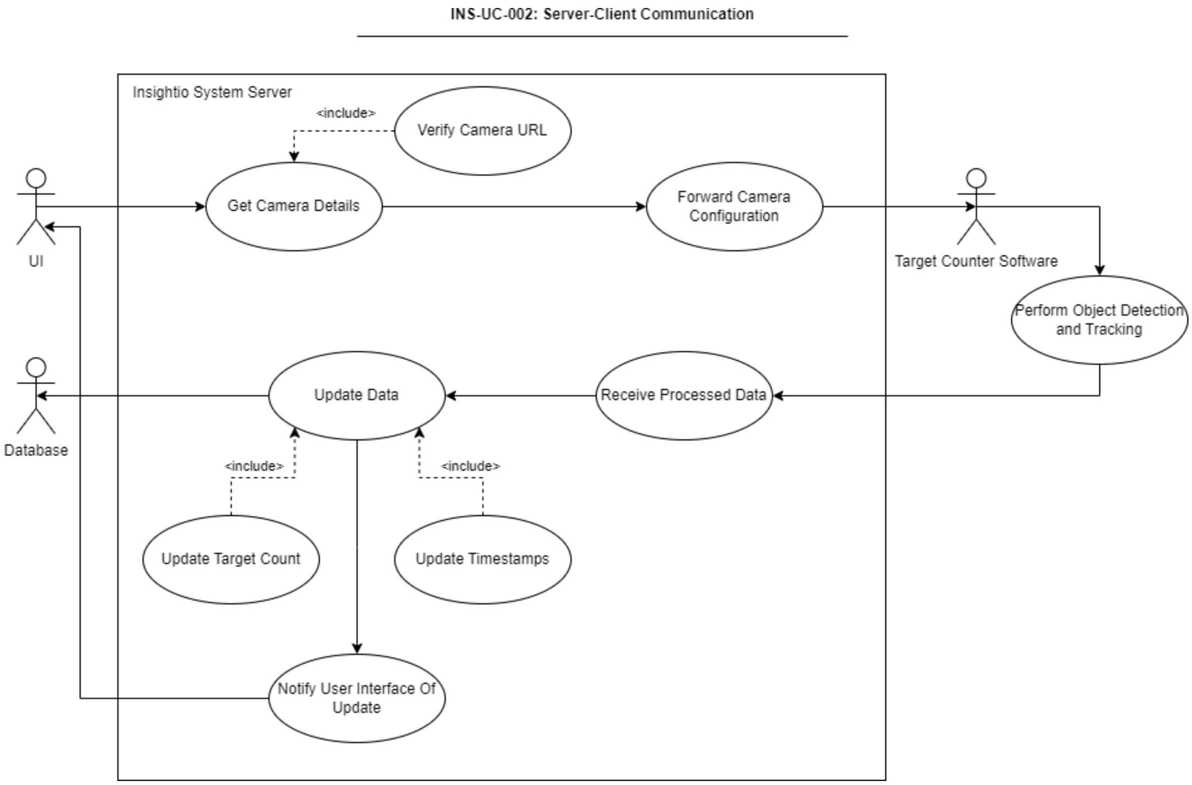
1. The user logs into the Insightio platform with valid credentials.
2. The system authenticates the user successfully.
3. After successful authentication, the system displays the user dashboard.
4. The user navigates to the statistics section through user interface.
5. The system retrieves statistical data based on the configured cameras and track- ing settings.
6. The retrieved data is presented to the user through visualizations such as charts and plots.

Alternate Flow for INS-UC-001:

2.a. If the user fails to authenticate, an error message is displayed, and feedback is provided to try logging in again.

4.a. If the user encounters issues executing functions in the user panel due to internet or system problems, they receive an error message.

4.2. INS-UC-002: Server-Client Communication



Use Case Name: INS-UC-002

Related Use Cases: INS-UC-003

Actors: Target Counter Software, Database, UI

Description In this use case, the process begins with the user configuring the camera through the interface, followed by the verification of the camera device URL, which is then sent to the tracker software for object detection and tracking. The tracker software reports the total target count within the designated time frame, and this data is subsequently inserted or updated in the database for continuous monitoring. In the event of potential errors, the system is equipped to handle them effectively. If the camera device URL cannot be verified, an error is promptly returned to the user interface, ensuring transparency about the issue. Similarly, if the tracker software fails to send data, the server patiently waits until new data is received, preventing data loss. Additionally, in cases where data transmission encounters errors, the system ensures proper error handling by returning an error message to the user interface. These error-handling mechanisms contribute to the overall reliability and resilience of the target counting system, ensuring a smooth user experience.

Preconditions:

- Stable connection to a valid camera device.
- Stable connection to database service.

Postconditions:

- Tracking target counts in the database are updated.
- Alternatively, if update/insertion fails, the server waits until tracker software sends new data in the next time frame.

Main Flow for INS-UC-002:

1. Camera configuration is received from the user interface.
2. Camera device URL is verified and sent to the tracker software.
3. Tracker software performs object detection and tracking.
4. In a given time frame, tracker software sends total target count at the end of that time frame.
5. Data is received from the tracker software and is inserted or updated in the database.
6. The same is done for the following time frame(s).
7. When the user asks to see target count data, count data is transmitted to the user interface.

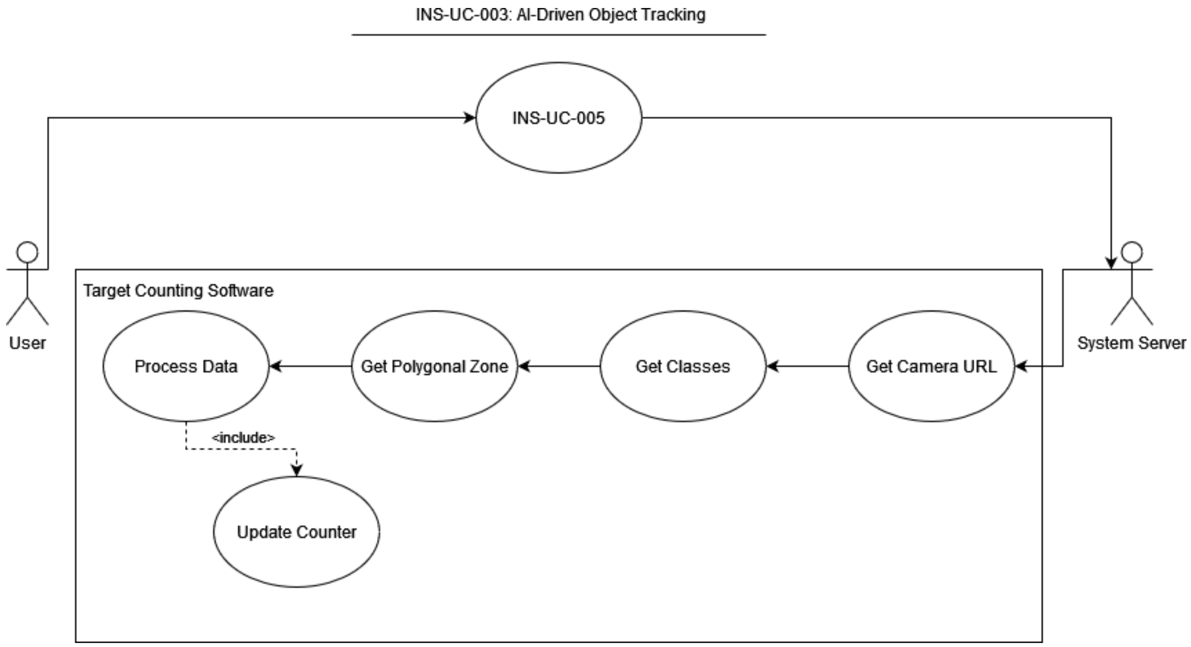
Alternative Flow for INS-UC-002:

2.a. If camera device cannot be verified with the provided URL, an error is returned to the user interface.

4.a. If the tracker software fails to send the data, the server waits until new data is received.

7.a. Should data transmission fail, an error is returned to the user interface.

4.3. INS-UC-003: AI-Driven Object Tracking



Use Case Name: INS-UC-003

Related Use Cases: INS-UC-005

Actors: User, System Server

Description: In the Insightio platform workflow, users initiate the process by retrieving essential information, such as the camera URL, and ensuring system connectivity to the video source. They then acquire knowledge about available object classes to customize tracking capabilities. Users can define polygonal zones on camera images, focusing counting efforts on specific regions. With these configurations in place, the system proceeds to process data, establishing connections and employing advanced AI algorithms for object recognition and tracking. This seamless workflow enables comprehensive data processing and analysis within the Insightio platform.Periodic updates of counters occur when trackers are identified. This structured sequence ensures that the Insightio platform is optimally configured to efficiently count, and track objects based on user-specified criteria.

Preconditions:

- The user is logged into the Insightio platform.
- The user has navigated to the camera setup and configuration section.

Postconditions:

- The camera is successfully configured with the provided settings.
- The system is connected to the specified video source.
- Object classes are identified and available for tracking.
- Polygonal zones, if defined, are established on the camera images.
- The system is ready for data processing and analysis.

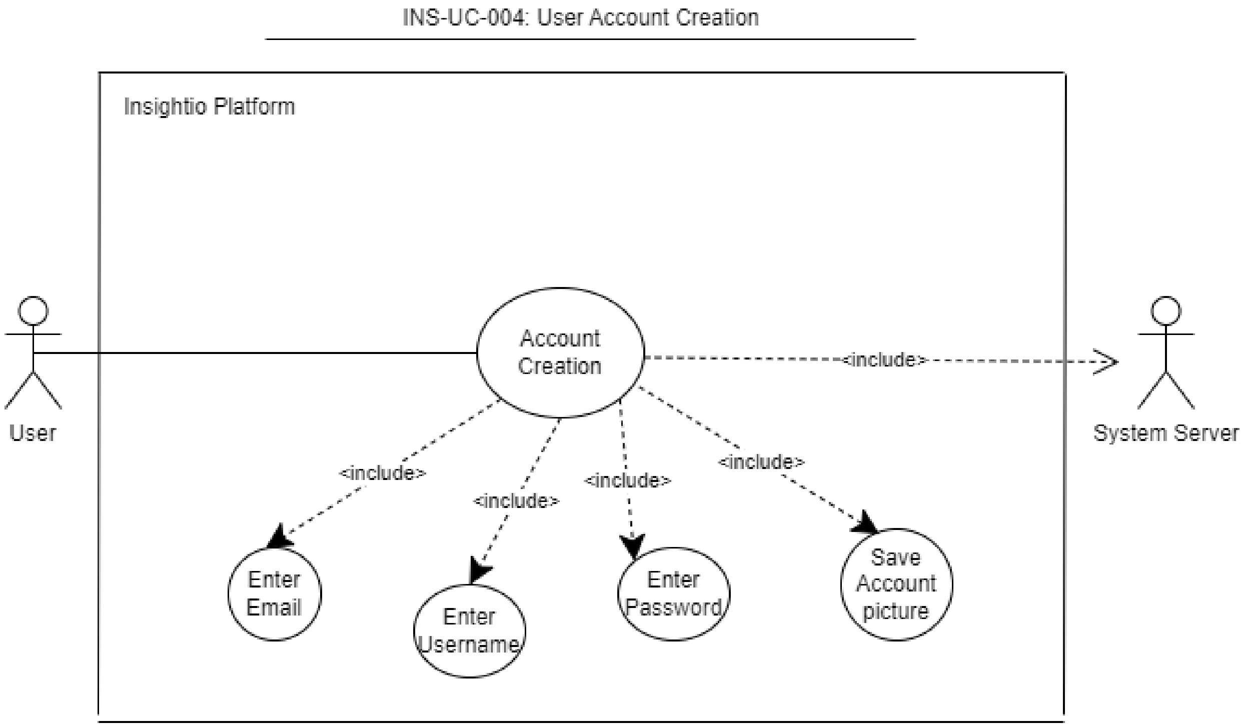
Main Flow for INS-UC-003:

1. User provides the camera URL to ensure system connectivity.
2. User acquires information about available object classes.
3. User defines polygonal zones on camera images for concentrated counting efforts.
4. User reviews and confirms the preliminary configurations.
5. The system processes data by establishing connections, applying AI algorithms, and initiating the periodic update of counters when trackers are identified.

Alternative Flow for INS-UC-003:

5.a. If there is an issue establishing connections or applying AI algorithms, the system returns an error message, and the user interface displays the error to the user.

4.4. INS-UC-004: User Account Creation



Use Case Name: INS-UC-004

Related Use Cases: -

Actors: User, System Server

Description: The procedure through which a user creates an account on the Insightio platform is represented by this use case. It includes necessary actions like supplying an email address, generating a username, and establishing a strong password.

Preconditions:

- The user has access to the Insightio website.
- The user has a stable internet connection.
- The user has not created an account on Insightio with the provided email address.

Postconditions:

- The user has successfully created an account on Insightio.

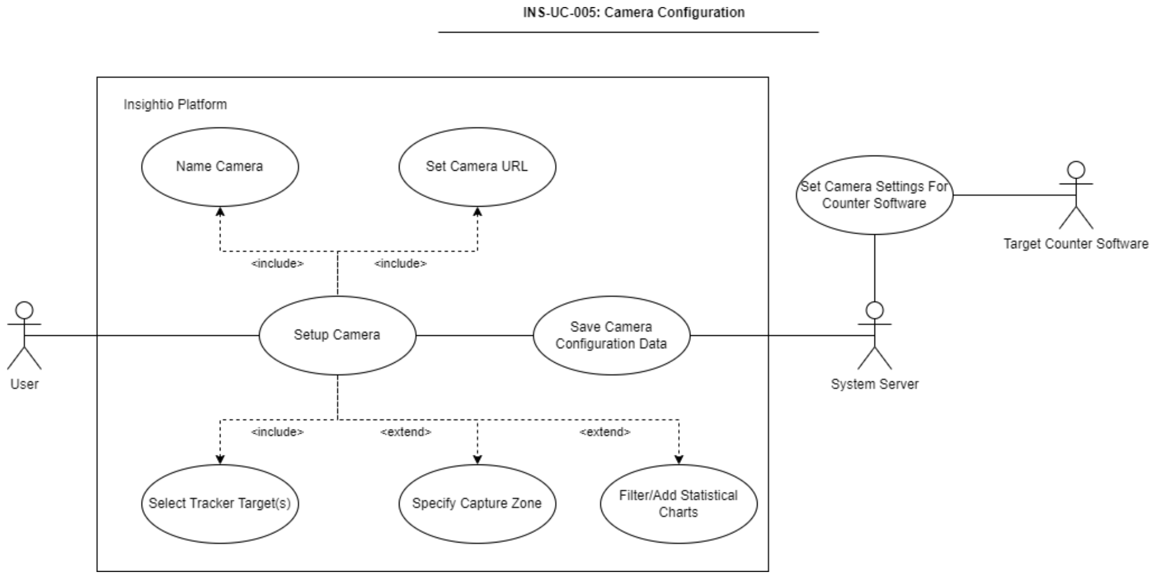
Main Flow for INS-UC-004:

1. The user initiates the account creation process by visiting the Insightio website.
2. The system prompts the user to enter their email address, desired username, and a secure password.
3. Upon successful submission, the system verifies the provided information and creates a new user account.

Alternative Flow for INS-UC-005:

4.a. If there is an issue during the account creation process (e.g., password minimum length requirement), the system provides appropriate error messages.

4.5. INS-UC-005: Camera Configuration



Use Case Name: INS-UC-005

Related Use Cases: -

Actors: User, System Server, Target Counter Software

Description: To initiate the recording of target counts, users must configure a camera through the platform interface. This setup involves assigning a name to the camera, specifying the URL of the camera device, and choosing target class(es) to inform the counting software about what to track. Optionally, at this stage, users can also establish capture zones known as polygon zones, concentrating counting efforts on specific regions within the camera feed. Users also have the option to choose or exclude statistical charts displaying the desired data from the captured camera feed. After completing the configuration, the camera settings are transmitted to the server for distribution to the counting software.

Preconditions:

- User logged into the system.
- User navigated to the camera configuration page.

Postconditions:

- A new camera is accepted as a data source or a an existing camera is given new targets.
- Alternatively, if provided device URL is invalid, camera configuration insertion or update fails.

Main Flow for INS-UC-005:

1. User specifies a camera name and provides the device URL.
2. User chooses target class(es) for tracking.
3. User optionally configures capture zones within the camera feed.
4. User optionally customizes statistical charts for desired data display.
5. User reviews and confirms the camera settings.
6. User transmits the configured settings to the server.
7. The server distributes the settings to the counting software.
8. Server confirms validity of device URL.
9. Counting software initiates recording based on the configured settings.

Alternative Flow for INS-UC-005:

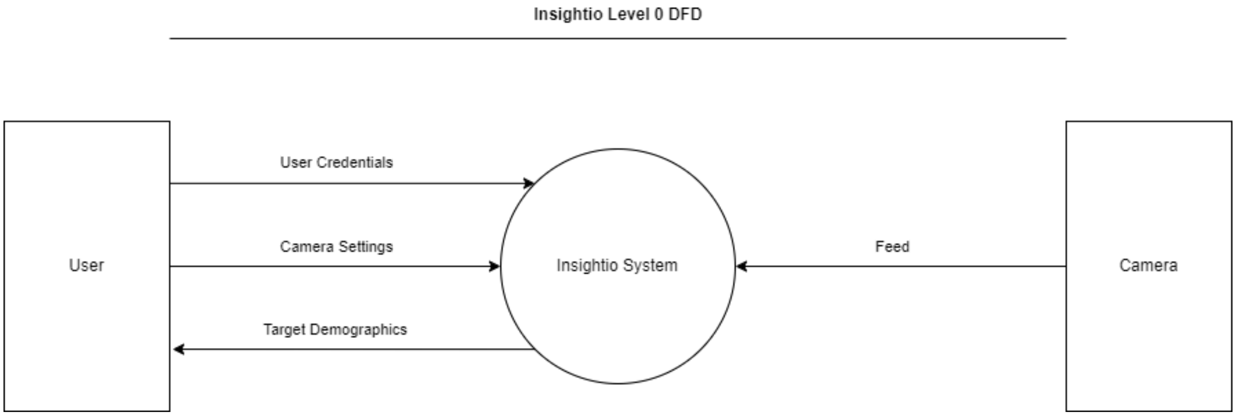
8.a. If the server cannot confirm the provided device URL belongs to a valid camera device, server does not forward the settings to the tracking software and returns an error to the user interface which the interface then displays to the user.

5 DFDs Overview

The Data Flow Diagrams (DFDs) for Insightio offer a concise portrayal of the system's data interactions. At Level 0, key components of the ecosystem are highlighted. These entities form the foundational structure, showcasing the primary data flows between users, cameras, and the Insightio System. The DFDs collectively illustrate the initial data pathways and entities that shape the broader Insightio data ecosystem.

5.1. Level 0

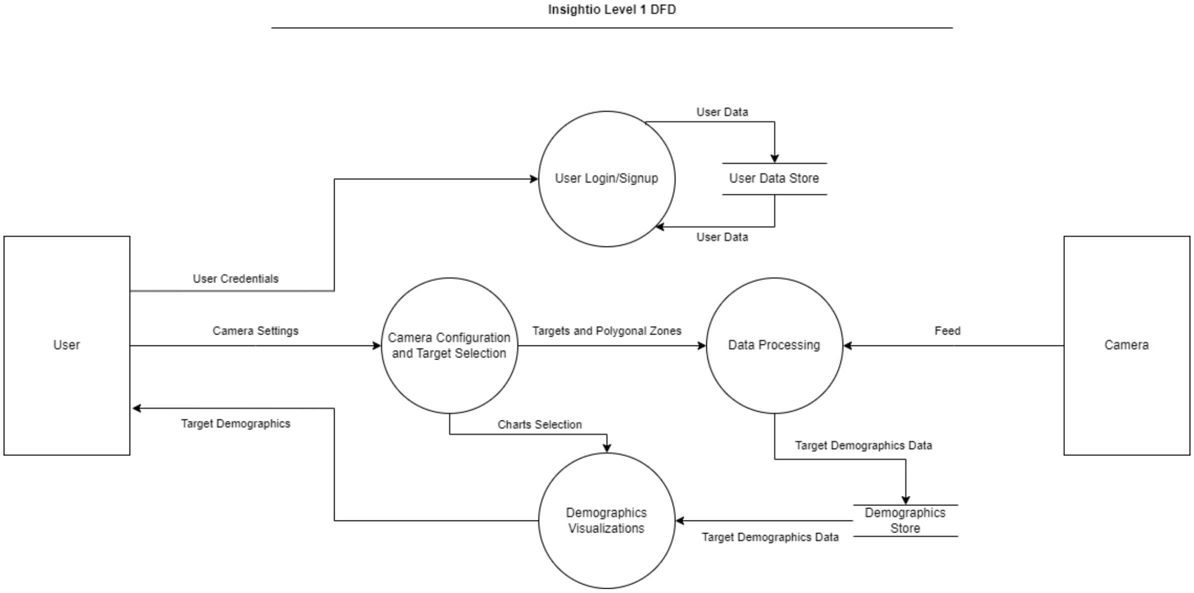
In Level 0, the Data Flow Diagram (DFD) accentuates pivotal external factors and entities integral to the initial data exchanges. This delineation encapsulates the data transactions between the Insightio System and users, incorporating the camera devices they utilize. As illustrated in the diagram, Insightio assumes the role of an intermediary, managing and interpreting camera feeds to enhance user convenience.



5.2. Level 1

Transitioning to Level 1, a more comprehensive exploration unveils specific processes and data stores, offering a detailed insight into the operations performed by the Insightio System. The diagram categorizes these operations into four main components: User Login/Signup, encompassing processes related to user account management; Camera Configuration and Target Selection, involving procedures for establishing camera connections and specifying demographic data targets; Data Processing, representing all processes related to extracting target demographics from camera feed frames; and Demographics Visualizations, consolidating processes designed to present demographic data to users through charts and plots.

This meticulous breakdown enriches understanding of the intricate data movements and processes within the Insightio platform, providing clarity from user interactions to data processing and visualization.



6 Conclusion

In summary, Insightio is a platform designed to meet businesses' crowd and object counting and tracking needs. This Software Requirements Specification (SRS) document provides an objective overview of the system's purpose, scope, and functional details.

The platform streamlines critical operations such user account creation, camera setting, and statistical data access. It caters to a variety of user roles, including business owners, operations managers, security personnel, data analysts, general users, and technical support staff, with features tailored to their specific needs and technical competence.

The functional requirements emphasize clear and testable specifications and cover critical areas such as user authentication, camera setup, real-time tracking, and data processing. Security, performance, fault tolerance, maintainability, internationalization, server system capacity, connectivity, usability, documentation, and data accuracy are examples of non-functional requirements.

The use cases present practical situations exhibiting user interactions with the platform, hence assisting in a thorough knowledge of system functionality

In conclusion, Insightio is a comprehensive solution, and the requirements mentioned serve as a guide for the development process, ensuring the delivery of a secure, stable, and user-friendly platform.

SOFTWARE DESIGN DOCUMENT

1. Introduction

The Insightio platform provides crowd/object counting and tracking solutions for businesses. Users have the capability to create an account, set up cameras, and define classes for monitored objects. This section explains the system design and user perspective of Insightio.

1.1. Purpose

Software Design Document provides a comprehensive overview of the architecture and system design of the Insightio application. In this document, functionality and systemdesign are explained from the user's perspective. Its purpose is to offer users a holistic view.

1.2. Definitions

Name	Definitions
User	Individuals who use and interact with the application for various purposes.
Actor	Any entity, including users or other software systems, that interacts with the primary system.
Object Detection	A computer vision technique that enables systems to identify objects in digital images, videos, and visual inputs.

Computer Vision	A subfield of Artificial Intelligence focused on making systems see and recognize objects using visual inputs.
YOLO Models	YOLO Models are a type of object detection algorithm used in computer vision.
Javascript	A scripting language commonly used to enhance the interactivity and functionality of web pages.
SvelteJS	A JavaScript framework used to build user interfaces.
Kotlin	A programming language often used for developing Android applications.
Python	A versatile programming language widely used for various applications.
Database	Organized collection of structured data, storing user settings, camera configurations, and statistical data for local processing in Insightio.
System Server	Component managing licenses in the current Insightio design, with potential future integration for enhanced license control and cloud-based solutions.
IP	Internet protocol
Id	Unique Identifier
UI	Interface between the user and the application

2. System Overview

The system is designed as a comprehensive platform offering users crowd and object counting capabilities, providing valuable statistical insights to businesses. Operating under the brand name "Insightio," the platform is accessible through a user- friendly web interface and offers simplicity from account creation to data analysis with the business-specific account.

Users engage with Insightio by creating accounts, a straightforward process involving the provision of basic details such as email addresses, usernames, and secure passwords.

The core functionality of Insightio revolves around real-time counting and tracking. Upon successful account creation, users can configure cameras, define tracking settings, and initiate the counting process. The platform seamlessly communicates with cameras, retrieves data, and employs advanced AI algorithms for object recognition and tracking.

Users have the flexibility to configure cameras, provide device URLs, choose target classes, and define capture zones for concentrated counting efforts. The server manages the distribution of camera settings to the counting software, ensuring accurate and reliablerecording based on user-configured parameters.

In summary, Insightio is a versatile platform that caters to the needs of businesses seeking to leverage crowd and object counting for informed decision-making. With its user-centric design, robust functionality, and emphasis on real-time data analysis, Insightio stands as a valuable tool for businesses across various industries.

3. System Design

Insightio's system architecture is designed to support desktop applications, with the main purpose of providing businesses object counting and tracking solutions. Along with its 3 layered architecture, this design choice allows the application to be deployed locally on clients, offering centralized enhanced control, reduced dependency and direct management of data processing.

3.1. Architectural Design

Insightio's system architecture is conceived to empower a desktop application aimed at providing businesses with crowd and object counting and tracking solutions. Thisdesign choice allows the application to run locally on a client's machine, offering enhanced control and direct management of data processing.

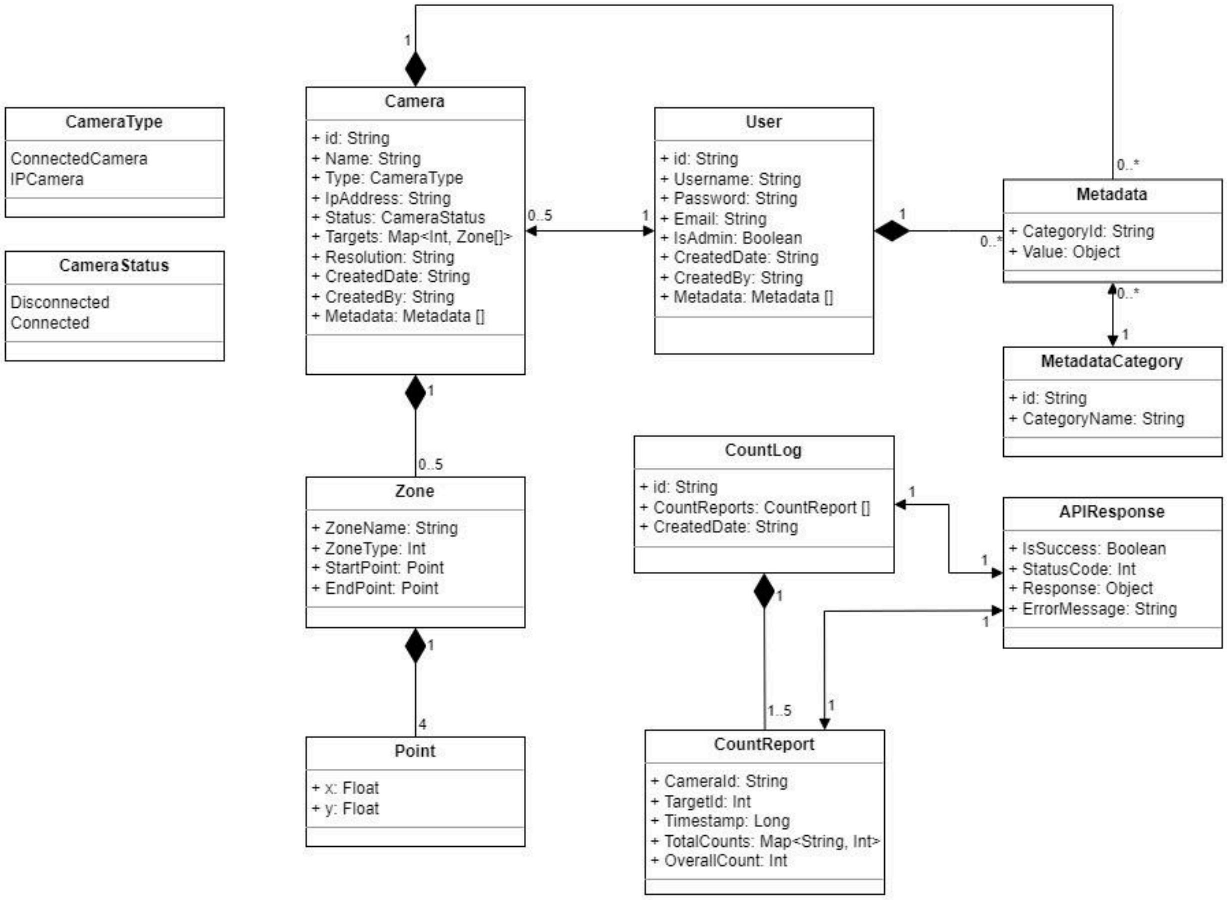
The architecture is segmented into three primary layers. The User Interface Layer, also called presentation layer, is tailored for simplicity and ease of use, enabling clients to seamlessly add cameras, configure object classifications, and define specific zones for counting within camera feeds. This layer is developed using desktop application technologies, ensuring a user-friendly experience.

In the Application Logic Layer, the core functionalities of Insightio are housed. This includes the execution of counting and tracking algorithms. It processes the configurations set by the user, managing the intricate interaction between the user interface and the data storage. By handling camera feed processing and object tracking tasks locally, the system ensures prompt response times and reduces dependency on continuous internet connectivity or external servers.

The Data Storage Layer is integral for the secure storage and management of user data, also responsible for interaction with the application layer. This includes client- specific settings such as camera configurations and the statistical data derived from object counting and tracking. The emphasis is on data integrity and security within the local operational environment of the client's desktop application. Insightio mainly stores tracking data, user data, camera configuration, zone definitions and logs.

While the current architecture primarily focuses on local data handling and processing, Insightio envisions the future integration of a central server. This server would primarily be responsible for controlling the licenses of each desktop application. Additionally, while not currently implemented, there's potential for incorporating cloud-based solutions in the future, particularly for data backups and enhanced data accessibility,catering to the evolving needs of the business.

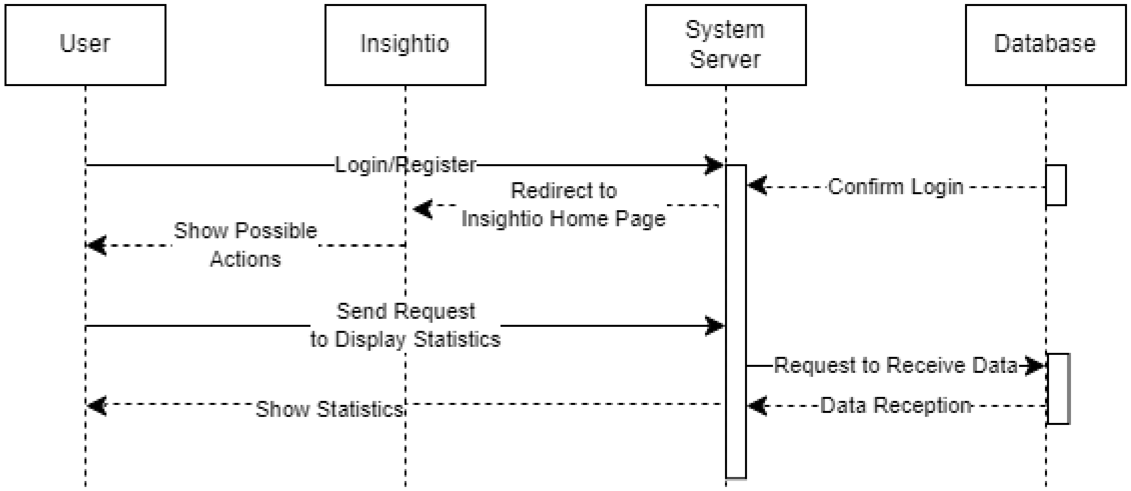
3.2. Decomposition Description



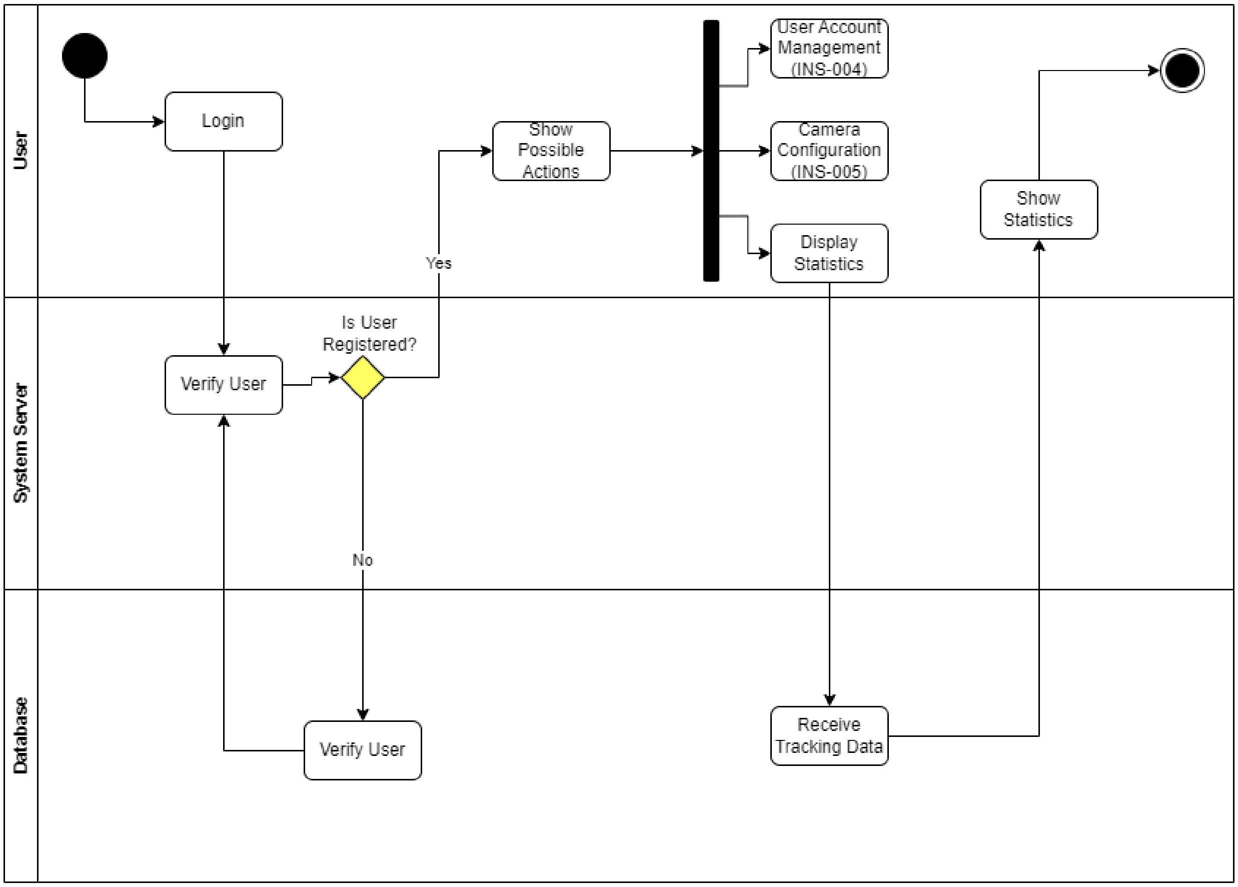
4. System Modeling (Activity & Sequence Diagrams)

INS-UC-001: Dashboard Integration

Sequence Diagram

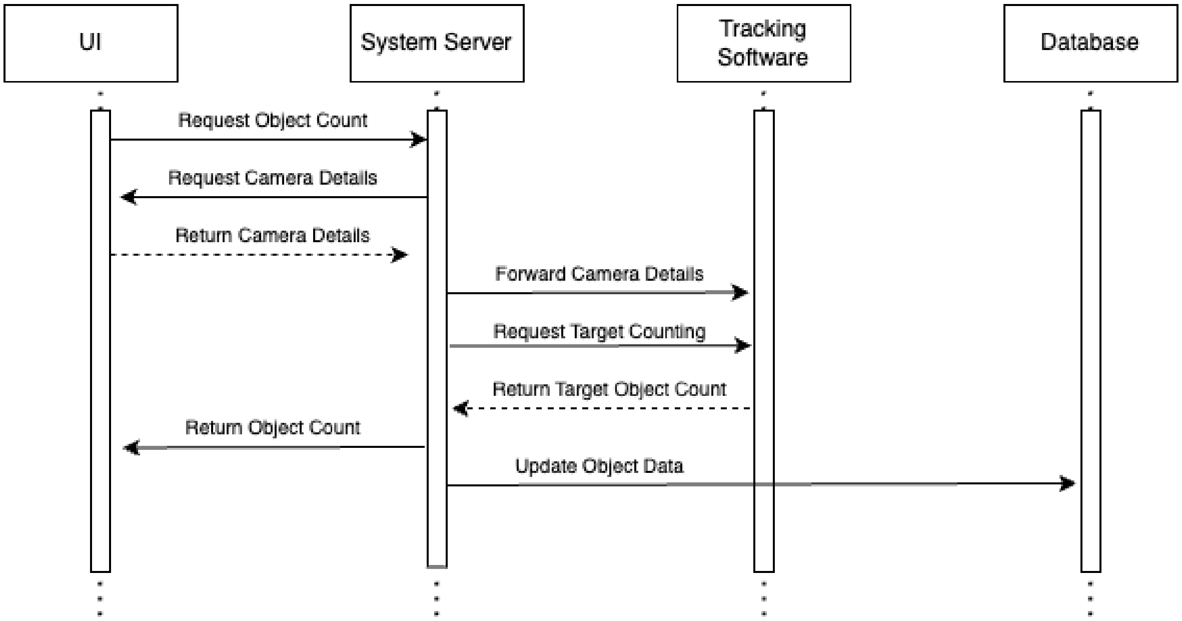


Activity Diagram

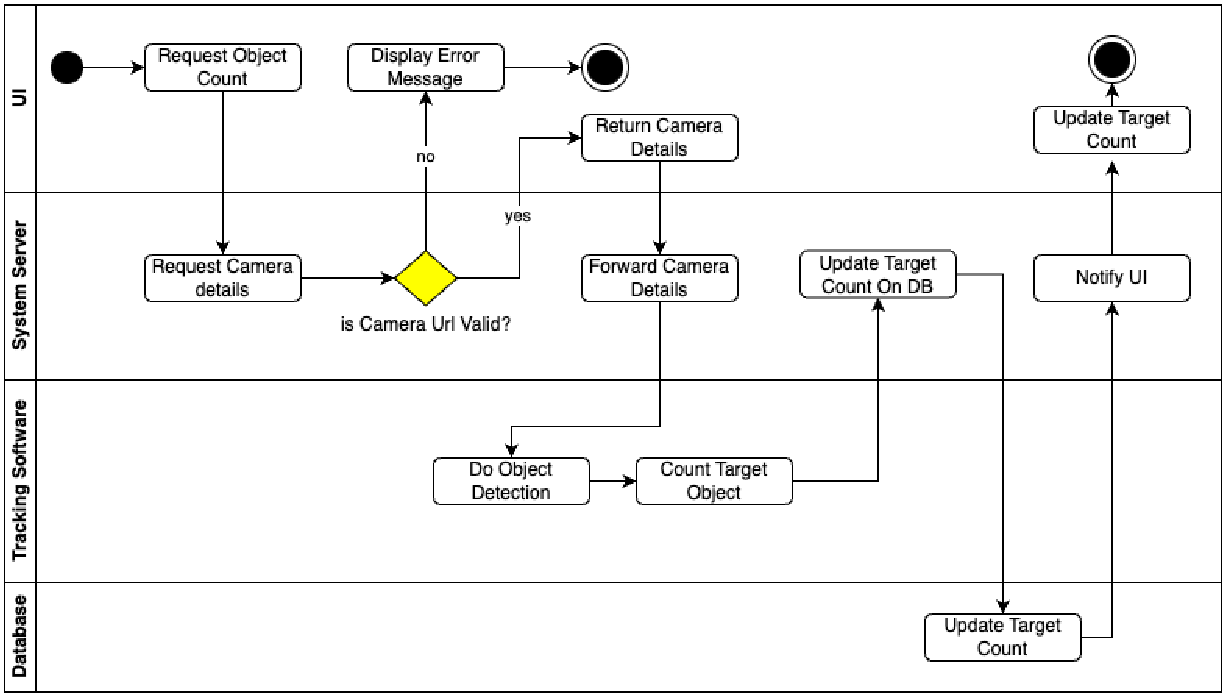


INS-UC-002: Client-Server Communication

Sequence Diagram

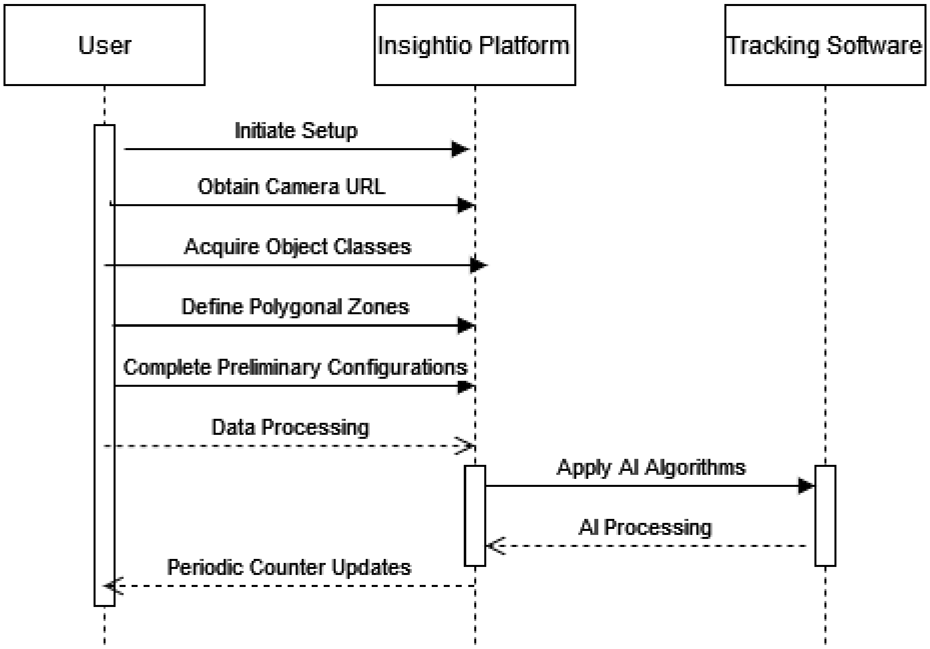


Activity Diagram

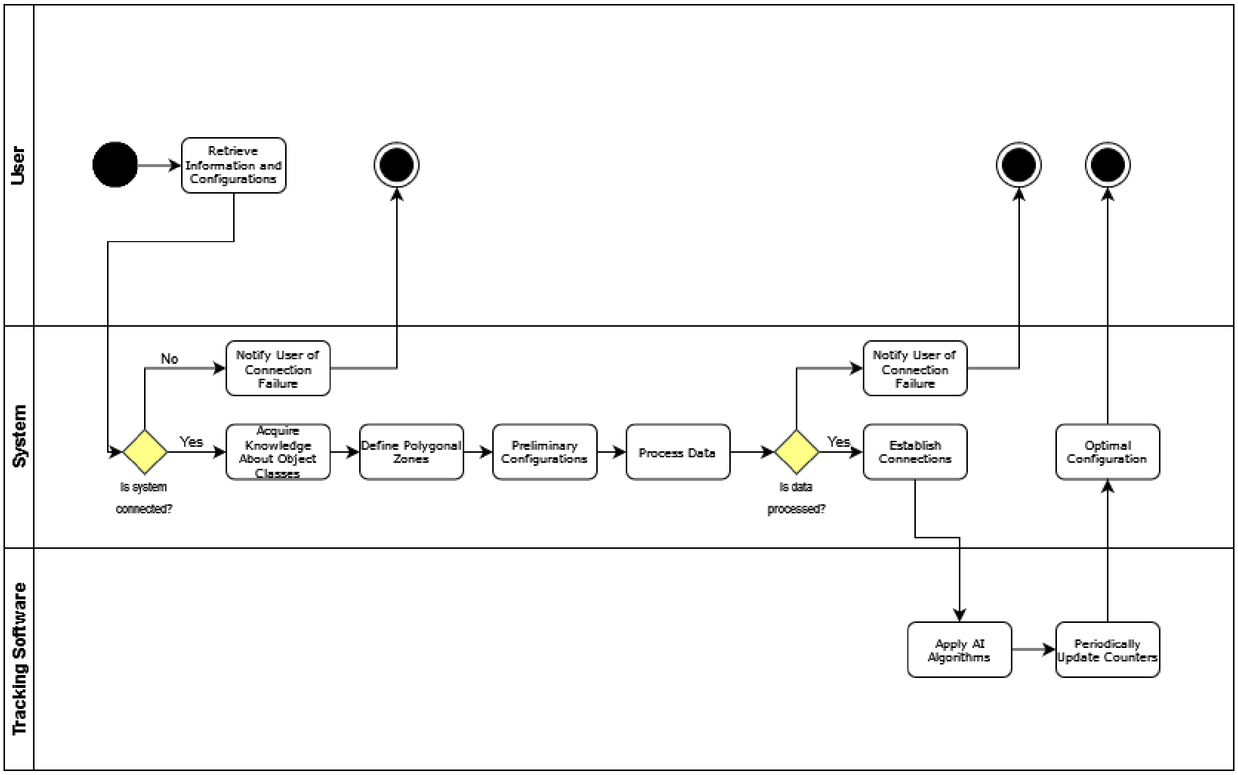


INS-UC-003: AI Driven Object Tracking

Sequence Diagram

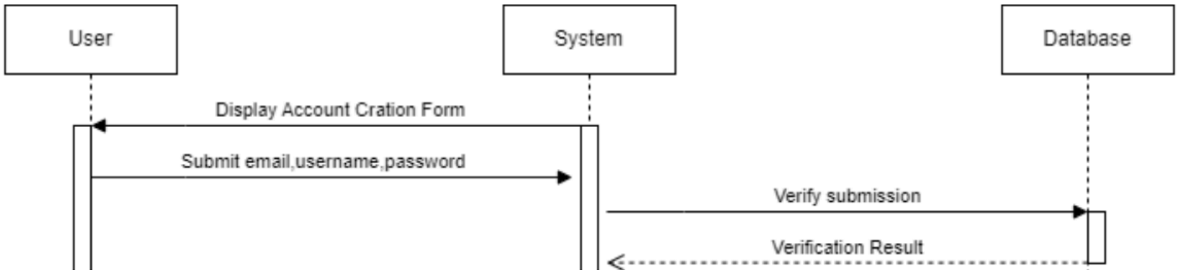


Activity Diagram

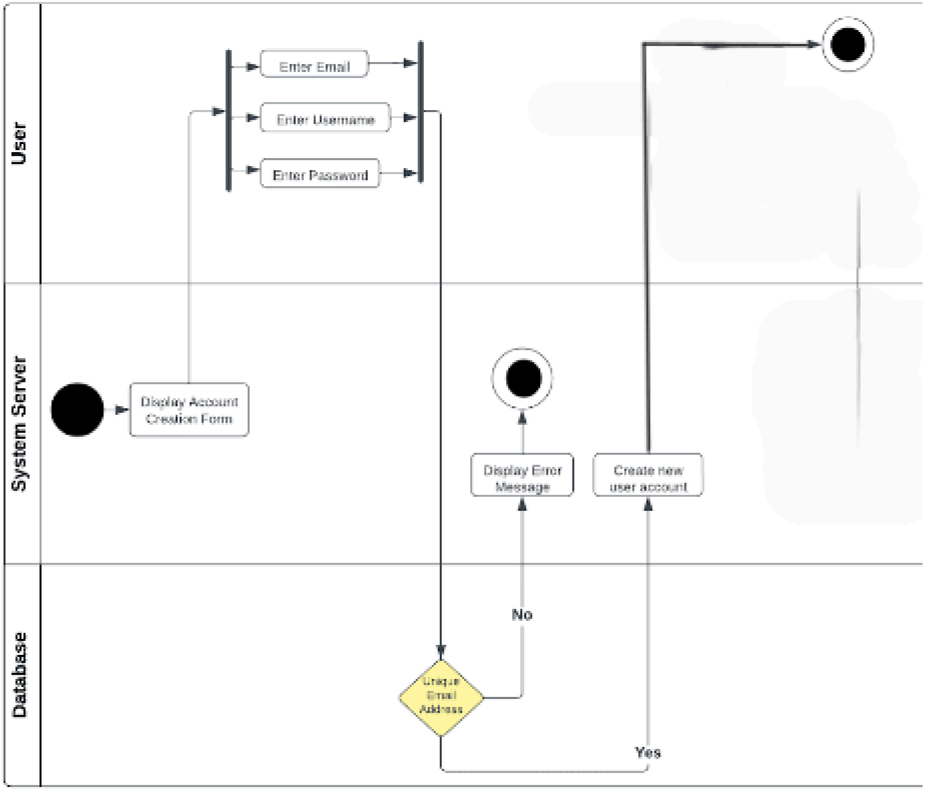

User
System
Tracking Software
Retrieve Information and Configurations
No
Notify User of Connection Failure
Yes
Is system connected?
Acquire Knowledge About Object Classes
Define Polygonal Zones
Preliminary Configurations
Process Data
Is data processed?
Yes
Notify User of Connection Failure
Establish Connections
Optimal Configuration
Apply AI Algorithms
Periodically Update Counters

INS-UC-004: User Account Creation

Sequence Diagram

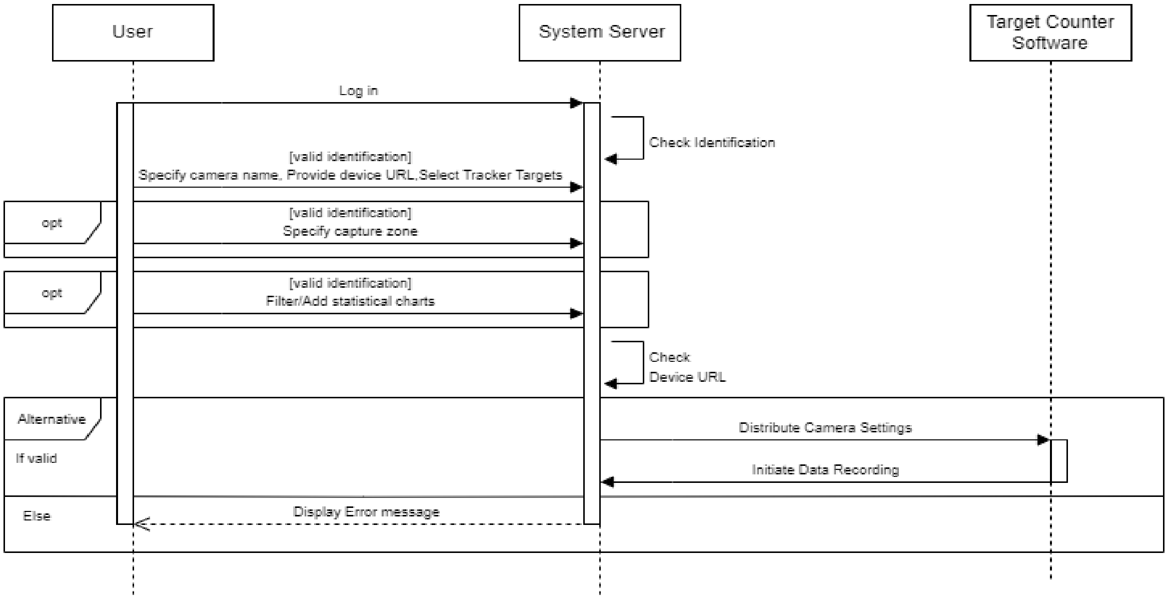


Activity Diagram

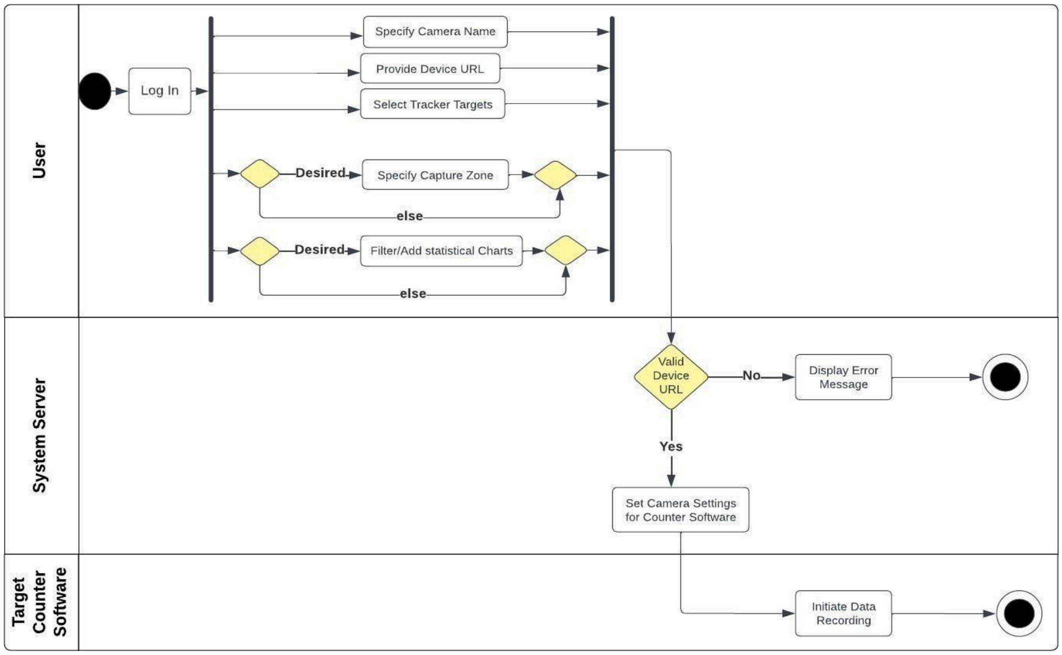


INS-UC-005: Camera Configuration

Sequence Diagram

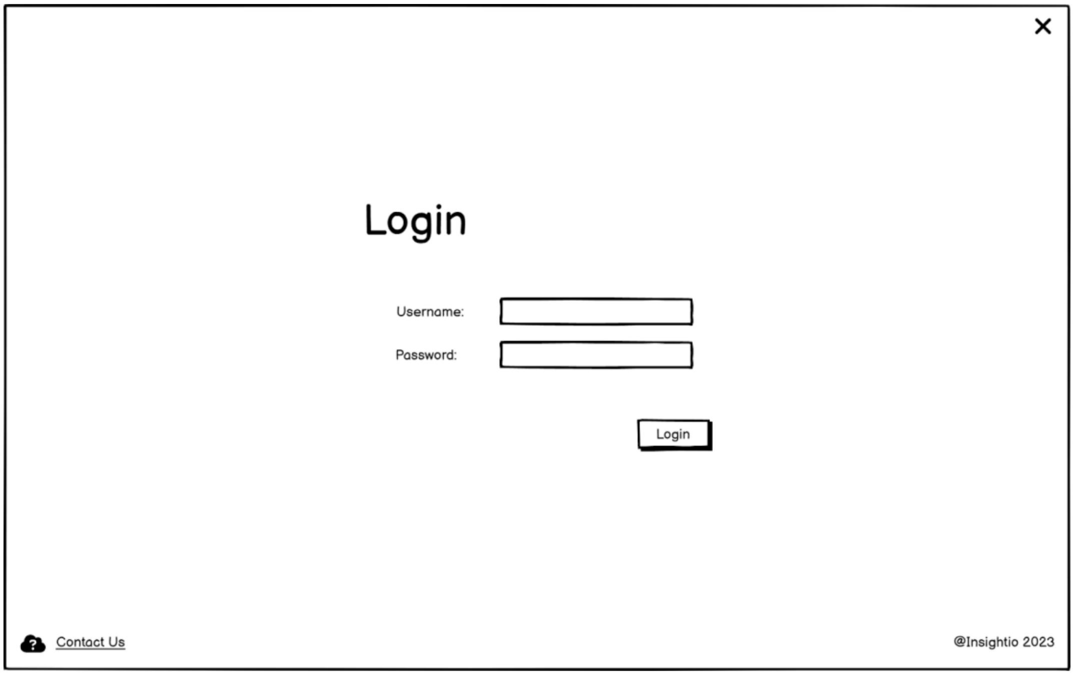


Activity Diagram



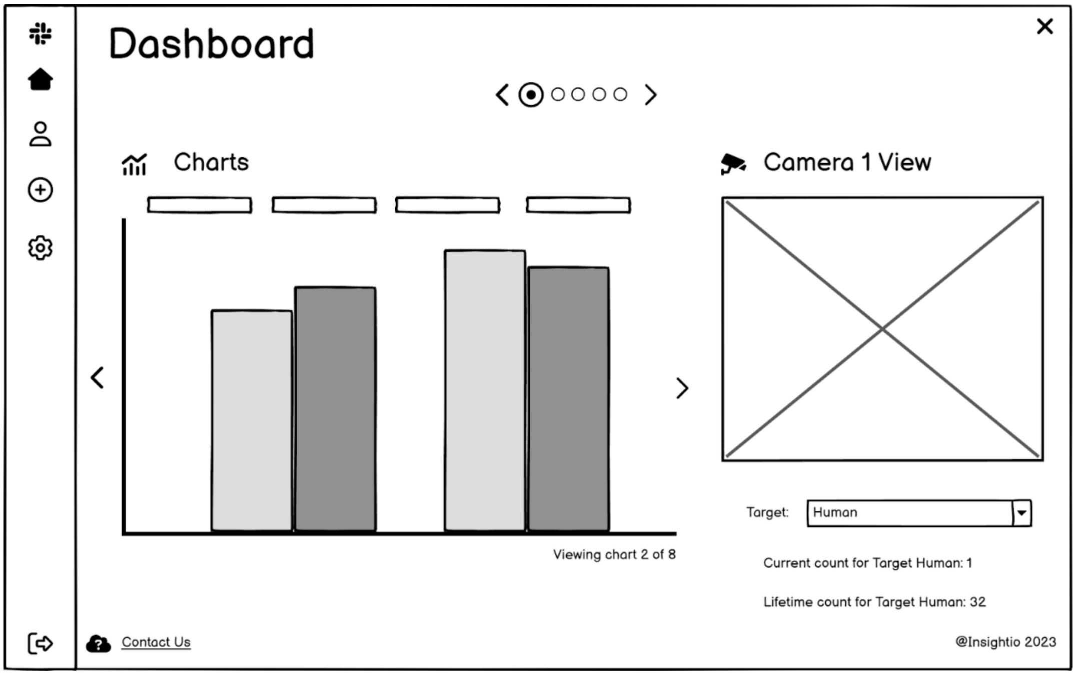
5. Interface Design

5.1. Login



5.2. Dashboard

5.2.1. Example 1



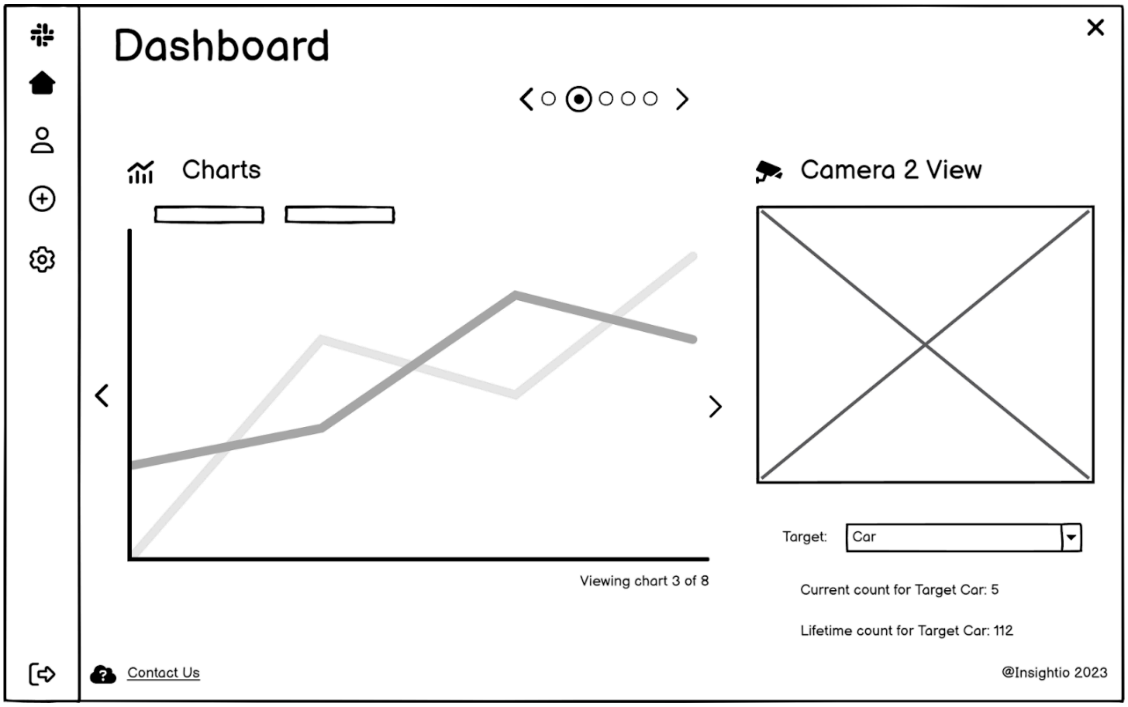
Viewing chart 2 of 8

Target: Human

Current count for Target Human: 1

Lifetime count for Target Human: 32

5.2.2. Example 2



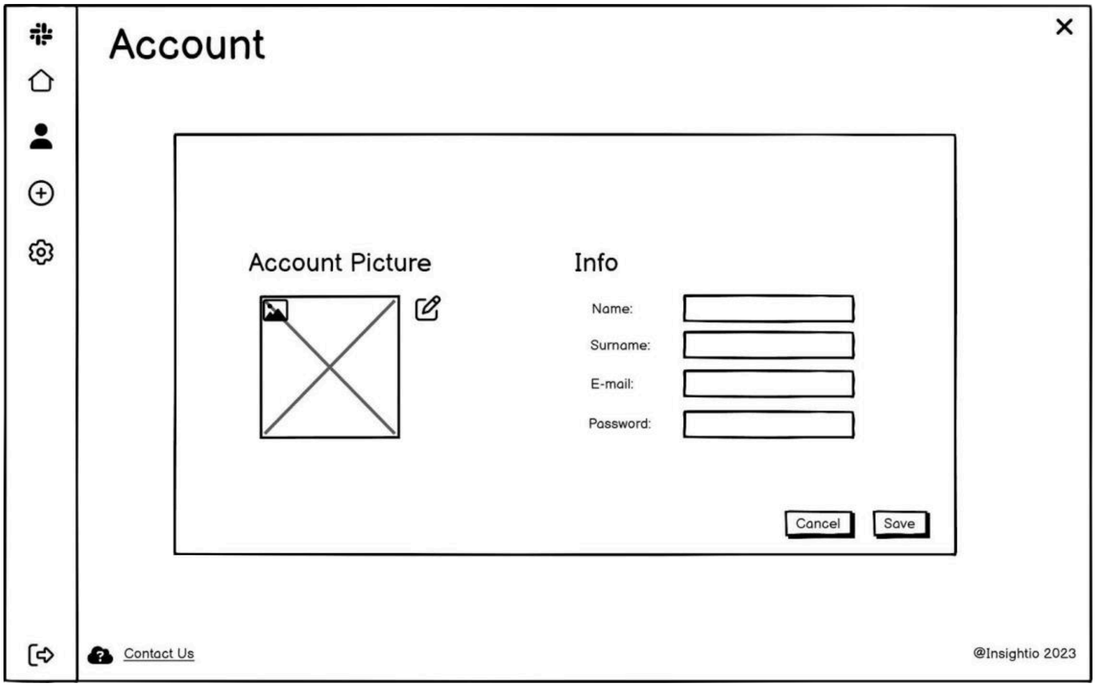
Viewing chart 3 of 8

Target: Car

Current count for Target Car: 5

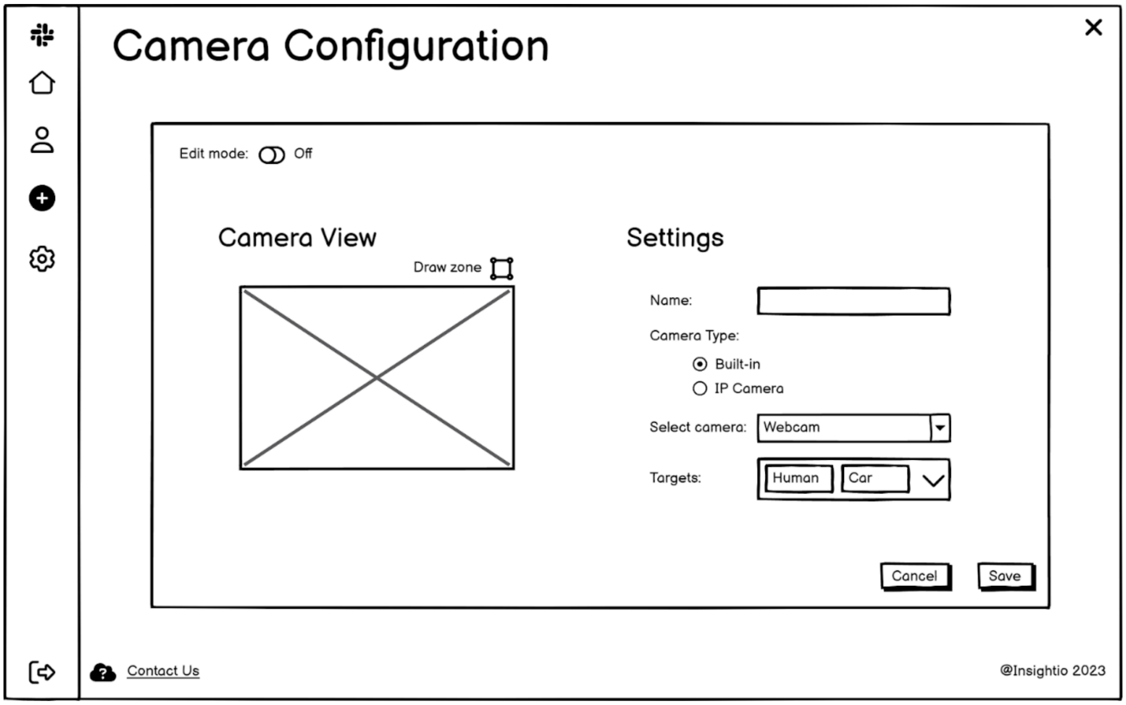
Lifetime count for Target Car: 112

5.3. Account



5.4. Camera Configuration

5.4.1. Edit mode off



Camera Configuration

Edit mode: Off

Camera View

Settings

Name:

Camera Type:

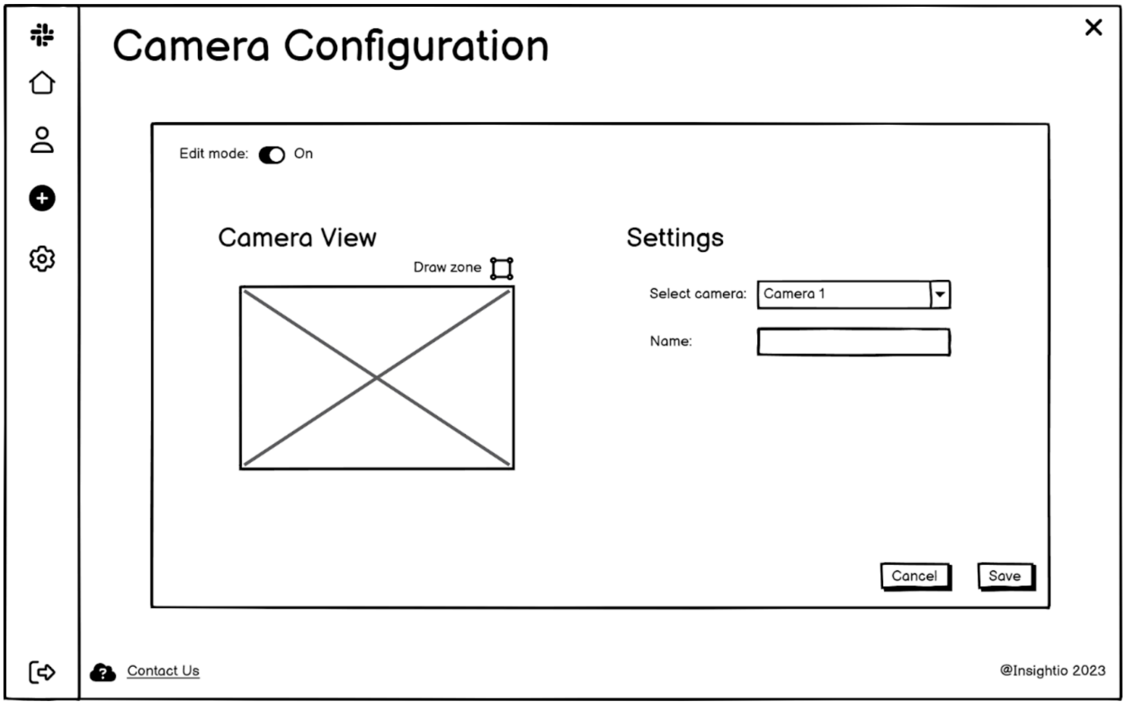
◉ Built-in

○ IP Camera

Select camera: Webcam

Targets: Human Car

5.4.2. Edit mode on



Camera Configuration

Edit mode: On

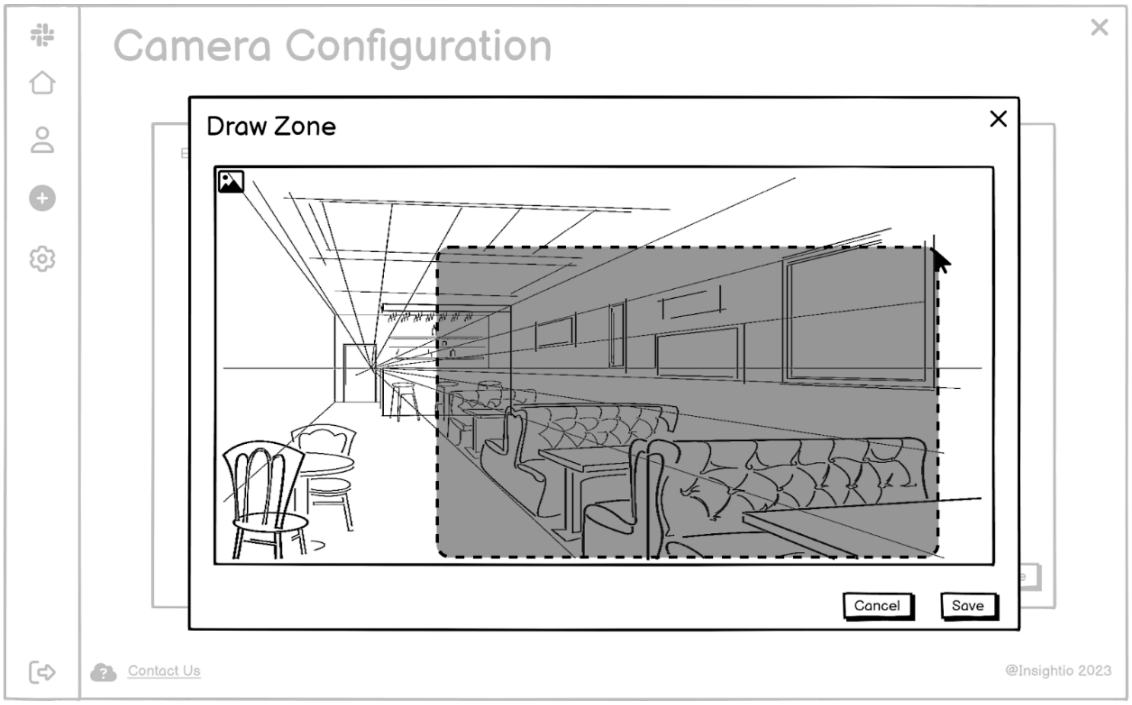
Camera View

Settings

Select camera: Camera 1

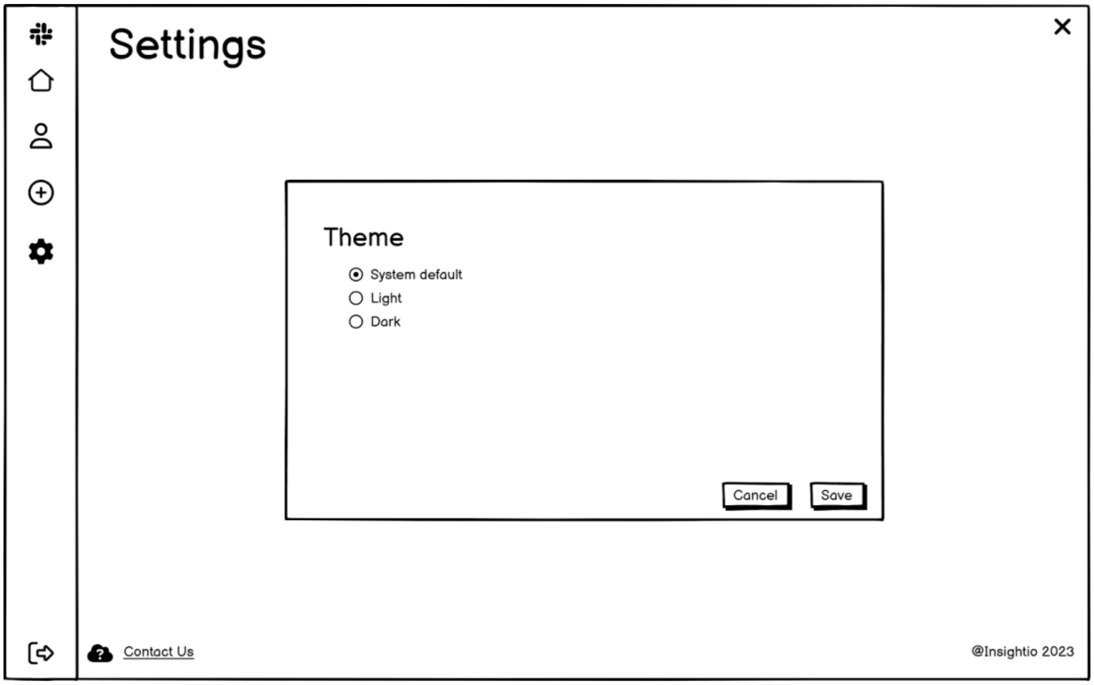
Name:

5.4.3. “Draw Zone” modal



5.5. Settings and Logout Modal

5.5.1. Settings

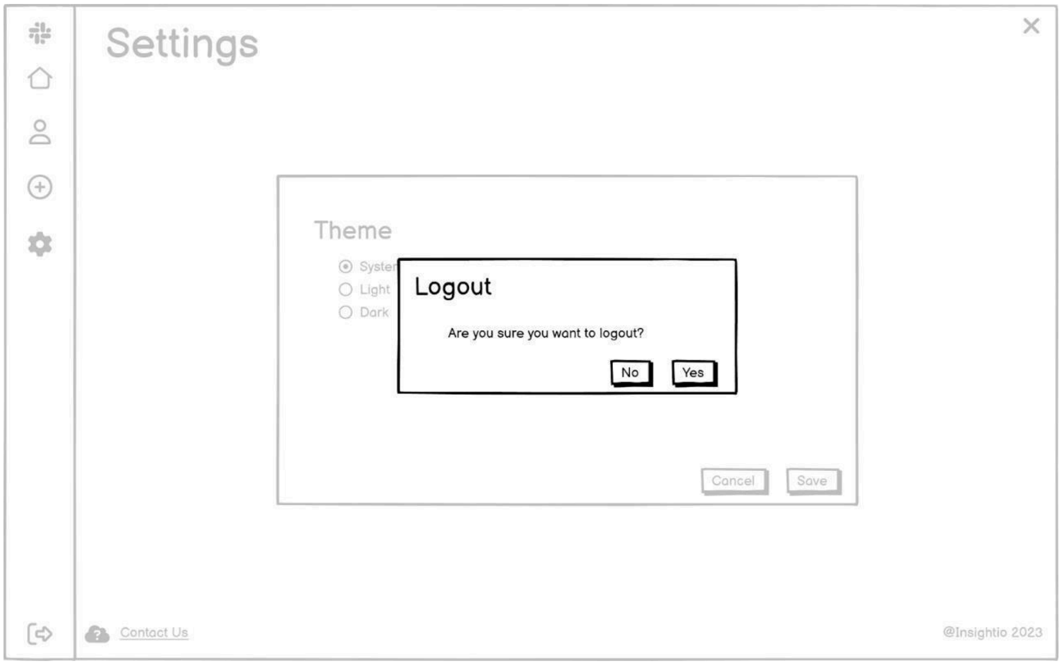


Settings

Theme

- ● System default
- ○ Light
- ○ Dark

5.5.2. Logout Modal



Settings

Theme

◉ Syster
○ Light
○ Dark

Logout

Are you sure you want to logout?

6. Test Plan

6.1. Introduction

6.1.1. Version Control

Version No.	Description of Changes	Date
0.0.1	Pre-Alpha	May, 9, 2024
0.1.0	Alpha	May, 26, 2024

6.1.2. Overview

The Insightio platform provides crowd/object counting and tracking solutions for businesses.The use case of the system had been explained in the SRS document. The characteristics of the Insightio will be tested on the system in various settings.

6.1.3. Scope

This document contains the test design specifications, the test cases in accordance with the test plan, and the test plan for the use cases. As a result, the parts that follow will describe the test requirements and how the test cases will be implemented.

6.1.4. Terminology

Acronym	Definition
SRS	Software Requirement Specification
SDD	Software Design Document
UI	User Interface
CS	Client-Server
AI	Artificial Intelligence
SP	Security and Performance
CUDA	Compute Unified Device Architecture

6.2. Features To Be Tested

In this section, we will explain our test plan and provide general information of the test cases for the distinctive aspects of our project.

6.2.1. UI Interactions of the Application (UI)

Interface Specifications Testing entails assessing the Insightio platform's user interface (UI) and interaction components to make sure they are simple to use, intuitive, and compliant with requirements. The purpose of this testing is to confirm that all interface elements work as intended and offer a smooth user experience. Features such as account creation, camera setup, and access to statistical data are checked for accuracy and usability.

6.2.2. Client-Server Communication (CS)

Client-Server Communication Testing evaluates the effectiveness and dependability of data transfer between the client and server of the Insightio platform, with a special emphasis on camera setup and user authentication procedures. Through accurate transmission of requests and responses over the network, secure user authentication, and smooth camera configuration, these tests guarantee consistent and safe platform operations.

6.2.3. AI Driven Object Tracking (AI)

Using cutting-edge artificial intelligence algorithms, AI-Driven Object Tracking Testing assesses the platform's ability to precisely identify and track designated objects within predetermined zones. In order to ensure that the system provides high-performance object recognition and statistical reporting that meets user expectations, this testing confirms the accuracy of zone definition, the dependability of object counting and tracking under various conditions, and the overall accuracy of the data.

6.2.4. Security and Performance

The purpose of Security and Performance Testing is to evaluate how well the Insightio platform safeguards user data and performs under different load scenarios. This entails testing the system's response times to guarantee quick data processing, the strength of password hashing algorithms for safe authentication, and RAM optimization to keep performance levels high without sacrificing security or speed.

6.3. Features Not To Be Tested

In this section, we described features that will not be tested.

6.3.1. External System Integrations

Due to the unpredictability of third-party service availability and configurations, as well as the presumption that these services carry out their own thorough testing, testing integrations with external systems—such as third-party APIs or services—falls outside our immediate purview. Instead of concentrating on the external systems per se, we continue to pay attention to the essential features of the application and the interfaces that connect it to them.

6.3.2. Computers Without CUDA

For the purpose of carrying out the intricate parallel computations required for AI and machine learning tasks, our application critically depends on CUDA. As these systems essentially lack the hardware capability needed to run our application, testing on computers without CUDA hardware is prohibited and is therefore irrelevant to our target user base and hardware requirements.

6.3.3. Real-Time Multi-User Interaction

Although our platform facilitates user-to-user interactions, the complex scenarios that entail synchronous, real-time actions and data transfers between users are not covered in the preliminary testing stages. This choice was made with an emphasis on core functionality and stability under typical usage conditions due to the difficulty of accurately simulating real-world interaction patterns and the substantial resources required.

6.4. Item Pass/Fail Criteria

A test case must satisfy all success requirements without resulting in any failure conditions in order to pass. A review and possible rework of the tested feature or function is required if there are no pass criteria or if there are any failure criteria.

6.4.1. Exit Criteria

- 100% of the test cases are executed
- 75% of the test cases passed
- All High Priority test cases passed

6.5. References

1. CENG408_Group6_SRS, Dec 08, 2023. Available: https://github.com/CankayaUniversity/ceng-407-408-2023-2024-Insightio/wiki/SRS
2. CENG408_Group6_SDD, Jan 22, 2023. Available: https://github.com/CankayaUniversity/ceng-407-408-2023-2024-Insightio/wiki/SDD

6.6. Test Design Specifications

6.6.1. UI Interactions of the Application

6.6.1.1. Subfeatures to be Tested

6.6.1.1.1. Account Edit (UI.AE)

Enables users to edit their account by entering personal information like a username, email address, and password; includes data integrity and security validations.

6.6.1.1.2. Camera Setup (UI.CS)

Enables users the ability to add, set up, and control camera devices on the platform. This includes creating monitoring zones and object tracking parameters.

6.6.1.1.3. Statistical Data (UI.SD)

Based on the object tracking data collected, it provides users with statistical reports and visual analytics that provide insights through customizable time frames, graphs, and charts.

6.6.1.2 Test Cases

TC ID	Priority	Scenario Description
UI.AE.01	M	Enter a valid user account picture, name, surname, email and password.
UI.CS.01	H	Select camera then enter a camera name and type. Finally, to draw zones select the target
UI.SD.01	M	Displaying statistics with charts based on the configured cameras and tracking settings.

6.6.2 Client-Server Communication

6.6.2.1. Subfeatures to be Tested

6.6.2.1.1 Camera Configuration (CS.CC)

Makes it easier for the client interface and the server to exchange camera settings and parameters, guaranteeing that cameras are set up correctly for object tracking in accordance with user requirements.

6.6.2.1.2 Authenticate Users (CS.AU)

Oversees the safe exchange of login credentials between the client and server, enabling platform access while guarding against illegal use and reset password with having "Reset Password" option.

6.6.2.1.3 Account Operations (CS.AO)

Enables users to add/edit their accounts to the server by entering personal information like a username, email address, account picture and password; includes data integrity and security validations.

6.6.2.2. Test Cases

TC ID	Priority	Scenario Description
CS.CC.01	H	Add a camera which is not already in db.
CS.CC.02	H	Add a camera which already exists.
CS.CC.03	M	Edit a camera.
CS.AU.01	H	Enter invalid credentials for login.
CS.AU.02	H	Enter valid credentials for login.
CS.AU.03	M	Reset password with "Forgot Password" option.
CS.AO.01	M	Add Account Picture.

6.6.3 AI Driven Object Tracking (AI)

6.6.3.1. Subfeatures to be Tested

6.6.3.1.1 Zone Definition (AI.ZD)

Improves the accuracy of data collected by enabling users to designate particular regions inside the camera's field of view for targeted object tracking and counting.

6.6.3.1.2 Object Counting and Tracking (AI.OCT)

Uses artificial intelligence (AI) algorithms to precisely identify, count, and track specific objects throughout the defined zones, enabling real-time analytics and monitoring.

6.6.3.1.3 Data Accuracy (AI.DA)

Guarantees the object counting and tracking process's dependability by continuously improving the algorithms used in data reporting and analysis to achieve high accuracy. Guarantees the object counting and tracking process's dependability by continuously improving the algorithms used in data reporting and analysis to achieve high accuracy.

6.6.3.1.4 Real-Time Object Recognition (AI.RTOR)

Utilizes real-time object recognition techniques to identify and classify objects accurately and efficiently.

6.6.3.2. Test Cases

TC ID	Priority	Scenario Description
AI.ZD.01	H	Verify that users can accurately define polygonal zones on camera images for targeted object tracking.
AI.OCT.01	H	Ensure that the system accurately identifies, counts, and tracks specific objects throughout the defined zones using AI algorithms.
AI.OCT.02	M	Verify that the system maintains accurate object tracking even in low-light conditions or with partially obscured objects.
AI.DA.01	H	Validate that the object counting and tracking process consistently achieves high accuracy through continuous algorithm improvements.
AI.RTOR.0.1	H	Verify that the system can accurately recognize and classify objects in real-time, maintaining high accuracy.

6.6.4 Security and Performance

6.6.4.1. Subfeatures to be Tested

6.6.4.1.1 Response Time (SP.RT)

Monitors the speed at which the system responds to queries and produces results, with the goal of minimizing latency to guarantee a seamless and effective user experience.

6.6.4.1.2 RAM Usage (SP.RAM)

Keeps track of and adjusts how much RAM the application uses while it is running to make sure system resources are used effectively without sacrificing performance.

6.6.4.1.3 Password Hashing (SP.PH)

Protects user credentials against illegal access and data breaches by using secure cryptography techniques to transform user passwords into hashed formats.

6.6.4.1.4 Access Control (SP.AC)

Implements access control mechanisms to restrict unauthorized access to sensitive features and data within the platform.

6.6.4.1.5 Audit Logging (SP.AL)

Logs all user actions and system events to provide a comprehensive audit trail for security monitoring and forensic analysis.

6.6.4.2. Test Cases

TC ID	Priority	Scenario Description
SP.RT.01	H	Test the system's response time to queries and ensure it meets performance benchmarks for seamless user experience.
SP.RT.02	M	Verify that the application effectively manages RAM usage to optimize system resources without compromising performance.
SP.PH.01	H	Ensure that user passwords are securely hashed to protect against unauthorized access and data breaches.
SP.AC.01	H	Verify that access controls are enforced consistently across all system features, preventing unauthorized access to sensitive data.
SP.AL.01	M	Validate that audit logs capture all user actions and system events accurately.

6.7. Detailed Test Cases

6.7.1. UI.AE.01

Purpose	Edit User Account
Requirements	Login into application successfully.
Priority	Medium
Estimated Time Needed	2 Minutes
Dependency	Login into application test cases should pass.
Setup	Application must be provided.
Procedure	- Go to the Account page.
	- Enter an account picture by choosing a file from the computer.
	- Enter a valid username, surname, email and password.
	- Click on the “Save” button.
	- Observe that the edit is successful and the account information has changed.
Cleanup	Close the application

6.7.2. UI.CS.01

Purpose	Make Camera Configuration
Requirements	Login into application successfully.
Priority	High
Estimated Time Needed	3 Minutes
Dependency	Selecting targets must be done before drawing the zones.
Setup	Application and at least one camera must be provided.
Procedure	- Go to Camera Configuration page.
	- Enter a camera name.
	- Choose Camera Type.
	- Select a Camera.
	- Select Targets.
	- Draw Zones.
	- Observe that the configuration is successful and the camera setup successfully made.
Cleanup	Close the application

6.7.3. UI.SD.01

TC_ID	UI.SD.01
Purpose	Display Statistics
Requirements	Login into application successfully.
Priority	Medium
Estimated Time Needed	1 Minute
Dependency	Selecting targets and time frame must be done before displaying statistics.
Setup	Application and at least one camera view must be provided.
Procedure	- Go to the Dashboard page.
	- Pick a camera view (if you have more than one camera that has been set).
	- Select target.
	- Select time frame to display specific data.
	- Select Chart type to display.
	- Observe that the display statistics are successful and accurate.
Cleanup	Close the application

6.7.4. CS.CC.01

TC_ID	CS.CC.01
Purpose	Add a new camera to the application
Requirements	Login into application successfully
Priority	High
Estimated Time Needed	5 Minutes
Dependency	Camera name, type, and selected targets must be provided
Setup	Application and at least one camera must be provided
Procedure	- Go to the Camera Configuration page.
	- Enter a camera name.
	- Select a camera type.
	- Select a camera.
	- Select Targets.
	- Draw Zones.
	- Observe that the settings for the new camera transmitted to the server successfully.
Cleanup	Close the application

6.7.5. CS.CC.02

TC_ID	CS.CC.02
Purpose	Try to add a new camera configuration with the same camera from the device to the application
Requirements	Login into application successfully
Priority	High
Estimated Time Needed	5 Minutes
Dependency	Camera name, type, and selected targets must be provided
Setup	Application and at least one camera that is already saved in the system must be provided
Procedure	- Go to the Camera Configuration page.
	- Enter a camera name.
	- Select a camera type.
	- Select a camera that is already selected and saved to the database.
	- Select Targets.
	- Draw Zones.
	- Observe that the application is not allowing for the new camera configuration with the same camera from the device.
Cleanup	Close the application

6.7.6. CS.CC.03

TC_ID	**CS.CC.03**
Purpose	Edit camera settings in application
Requirements	Login into application successfully
Priority	Medium
Estimated Time Needed	3 Minutes
Dependency	Existing Camera Configuration must be provided
Setup	Application and at least one camera configuration must be provided
Procedure	- Go to the Camera Configuration page.
	- Select Edit Option from the panel.
	- Select a camera configuration that already exists in the device.
	- Enter a camera name.
	- Select Targets.
	- Draw Zones.
	- Observe that the edited settings for the camera are transmitted to the server successfully.
Cleanup	Close the application

6.7.7. CS.AU.01

TC_ID	CS.AU.01
Purpose	Check if invalid credentials may be logged in to the application
Requirements	-
Priority	High
Estimated Time Needed	2 Minutes
Dependency	-
Setup	Application must be provided
Procedure	- Go to the Login page.
	- Enter invalid username.
	- Enter invalid password.
	- Click the login button.
	- Observe that logging in to the application with invalid credentials is not allowed by the server.
Cleanup	Close the application

6.7.8. CS.AU.02

TC_ID	CS.AU.02
Purpose	Logged in to the application with valid credentials
Requirements	User should be created
Priority	High
Estimated Time Needed	2 Minutes
Dependency	User must exist in the database
Setup	Application must be provided
Procedure	- Go to the Login page.
	- Enter valid username.
	- Enter valid password.
	- Click the login button.
	- Observe that logging in to the application with valid credentials is done.
Cleanup	Close the application

6.7.9. CS.AU.03

TC_ID	CS.AU.03
Purpose	Reset Password
Requirements	User should be created
Priority	Medium
Estimated Time Needed	1 Minutes
Dependency	User must exist in the database
Setup	Application must be provided
Procedure	- Go to the Login page.
	- Click the "Reset Password" button.
	- Observe that resetting of password is done.
Cleanup	Close the application

6.7.10. CS.AO.01

TC_ID	CS.AO.01
Purpose	Add Account Picture
Requirements	User should be created
Priority	Medium
Estimated Time Needed	1 Minutes
Dependency	Users must be logged into the system.
Setup	Application must be provided
Procedure	- Go to the Account page.
	- Click the "Account Picture" button.
	- Observe that saving picture to database is successful.
Cleanup	Close the application

6.7.11. AI.ZD.01

TC_ID	AI.ZD.01
Purpose	Verify that users can accurately define polygonal zones on camera images for targeted object tracking
Requirements	Login into application successfully
Priority	High
Estimated Time Needed	10 minutes
Dependency	User must have access to camera configuration settings
Setup	User must be logged into the application with access to camera configuration settings
Procedure	- Navigate to the Camera Configuration page.
	- Enter the camera name.
	- Select a camera type.
	- Draw polygonal zones on camera images to define tracking areas.
	- Observe that the system allows users to accurately define polygonal zones without encountering any errors.
Cleanup	Close the application

6.7.12. AI.OCT.01

TC_ID	AI.OCT.01
Purpose	Ensure that the system accurately identifies, counts, and tracks specific objects throughout the defined zones using AI algorithms
Requirements	Login into application successfully
Priority	High
Estimated Time Needed	15 minutes
Dependency	User must have access to object tracking settings
Setup	User must be logged into the application with access to object tracking settings
Procedure	- Navigate to the Object Tracking page.
	- Select the desired object classes for tracking.
	- Draw polygonal zones on camera images to define tracking areas.
	- Observe that the system accurately identifies and tracks specific objects within the defined zones without errors.
Cleanup	Close the application

6.7.13. AI.OCT.02

TC_ID	AI.OCT.02
Purpose	Verify that the system maintains accurate object tracking even in low-light conditions or with partially obscured objects
Requirements	Login into application successfully
Priority	Medium
Estimated Time Needed	15 minutes
Dependency	User must have access to object tracking settings
Setup	User must be logged into the application with access to object tracking settings
Procedure	- Navigate to the Object Tracking page.
	- Select the desired object classes for tracking.
	- Draw polygonal zones on camera images to define tracking areas.
	- Observe that the system accurately maintains object tracking even in low-light conditions or with partially obscured objects.
Cleanup	Close the application

6.7.14. AI.DA.01

TC_ID	**AI.DA.01**
Purpose	Validate that the object counting and tracking process consistently achieves high accuracy through continuous algorithm improvements
Requirements	Login into application successfully
Priority	High
Estimated Time Needed	20 minutes
Dependency	User must have access to object tracking settings
Setup	User must be logged into the application with access to object tracking settings
Procedure	- Navigate to the Object Tracking page.
	- Select the desired object classes for tracking.
	- Draw polygonal zones on camera images to define tracking areas.
	- Observe that the system consistently achieves high accuracy in object counting and tracking without errors.
Cleanup	Close the application

6.7.15. AI.TOR.01

TC_ID	AI.RTOR.01
Purpose	Verify that the system can accurately recognize and classify objects in real-time, maintaining high accuracy.
Requirements	Login into application successfully.
Priority	High
Estimated Time Needed	10 minutes
Dependency	User must have access to real-time object recognition settings.
Setup	User must be logged into the application with access to object tracking settings.
Procedure	
	- Navigate to the Object Tracking page.
	- Ensure the camera feed is active and streaming.
	- Observe that the system accurately recognizes and classifies objects in real-time without errors.
Cleanup	Close the application

6.7.16. SP.RT.01

TC_ID	**SP.RT.01**
Purpose	Test the system's response time to queries and ensure it meets performance benchmarks for a seamless user experience
Requirements	Successful login into the application
Priority	High
Estimated Time Needed	10 minutes
Dependency	User must have access to the system's query interface
Setup	User must be logged into the application with access to the query interface
Procedure	- Execute a query within the application.
	- Observe the system's response time and ensure it meets performance benchmarks without delays.
Cleanup	Close the application

6.7.17. SP.RT.02

TC_ID	SP.RT.02
Purpose	Verify that the application effectively manages RAM usage to optimize system resources without compromising performance
Requirements	Successful login into the application
Priority	Medium
Estimated Time Needed	15 minutes
Dependency	User must have access to system monitoring tools
Setup	User must be logged into the application with access to system monitoring tools
Procedure	- Monitor RAM usage while the application is running.
	- Observe that the application effectively manages RAM usage without excessive consumption or performance degradation.
Cleanup	Close the application

6.7.18. SP.PH.01

TC_ID	**SP.PH.01**
Purpose	Ensure that user passwords are securely hashed to protect against unauthorized access and data breaches
Requirements	Successful login into the application
Priority	High
Estimated Time Needed	5 minutes
Dependency	User must have access to the password management interface
Setup	User must be logged into the application with access to the password management interface
Procedure	- Change the user password through the application interface.
	- Verify that the system securely hashes the password and stores it in the database.
Cleanup	Close the application

6.7.19. SP.AC.01

TC_ID	SP.AC.01
Purpose	Verify that access controls are enforced consistently across all system features, preventing unauthorized access to sensitive data
Requirements	Successful login into the application
Priority	High
Estimated Time Needed	15 minutes
Dependency	User must have access to sensitive data within the application
Setup	User must be logged into the application with access to sensitive data
Procedure	- Attempt to access sensitive data without proper authorization.
	- Verify that the system enforces access controls and prevents unauthorized access to sensitive data.
Cleanup	Close the application

6.7.20. SP.AL.01

TC_ID	**SP.AL.01**
Purpose	Validate that audit logs capture all user actions and system events accurately
Requirements	Successful login into the application
Priority	Medium
Estimated Time Needed	15 minutes
Dependency	User must have access to audit logs
Setup	User must be logged into the application with access to audit logs
Procedure	- Perform various actions within the application, such as accessing data or modifying settings.
	- Verify that all user actions and system events are accurately recorded in the audit logs.
Cleanup	Close the application

7. Results

TC ID	Result	Priority	Explanation
UI.AE.01	Pass	M	User inputs taken successfully via system.
UI.CS.01	Pass	H	User can set up the camera successfully.
UI.SD.01	Pass	M	Statistics based on the tracking data displayed successfully.
CS.CC.01	Pass	H	New camera configuration transmitted server successfully.
CS.CC.02	Pass	H	Trying for new Camera Configuration with the same camera from the device is not permitted.
CS.CC.03	Pass	M	Edited camera settings transmitted server successfully.
CS.AU.01	Pass	H	Logging in to the application with invalid credentials is not allowed.
CS.AU.02	Pass	H	Logged in to the application with valid credentials successfully.
CS.AU.03	Fail	M	Resetting password is unsuccessful.
CS.AO.01	Fail	M	Adding pictures to the database is unsuccessful.
AI.ZD.01	Pass	H	Users were able to accurately define polygonal zones on camera images.
AI.OCT.01	Pass	H	The system accurately identified, counted, and tracked specific objects.
AI.OCT.02	Pass	M	The system successfully maintained accurate object tracking even in challenging conditions.
AI.DA.01	Pass	H	The object tracking process consistently achieved high accuracy.
AI.RTOR.01	Pass	H	The system accurately recognized and classified objects in real-time, maintaining high accuracy levels during testing.
SP.RT.01	Pass	H	The system responded promptly to queries, meeting performance benchmarks.
SP.RT.02	Pass	M	The application effectively managed RAM usage.

SP.PH.01	Pass	H	User passwords were securely hashed.
SP.AC.01	Pass	H	Access controls were consistently enforced across all system features.
SP.AL.01	Pass	H	Audit logs accurately captured all user actions and system events.

8. User Manual

8.1. Scope of Document

This manual serves as a comprehensive guide for effectively utilizing our software. From setup procedures to detailed explanations of each page's functionalities, users will find clear instructions for configuring cameras, setting up tracking parameters, and analyzing tracking data. With this resource, users can swiftly familiarize themselves with Insightio's capabilities, optimizing surveillance operations and leveraging valuable insights.

8.1.1. Release Dates

Date	Version	Release Name
09.05.2024	v0.0.1	Pre-alpha Release
26.05.2024	v0.1.0	Alpha Release

8.1.2 Glossary

Acronym	Definition
API	Application Programming Interface
CPU	Central Processing Unit
GPU	Graphics Processing Unit
UI	User Interface
IP	Internet Protocol
VS Code	Visual Studio Code
URL	Uniform Resource Locator
CUDA	Compute Unified Device Architecture

8.2. Introduction

The Insightio platform provides crowd/object counting and tracking solutions for businesses.The use case of the system had been explained in the SRS document. The characteristics of the Insightio will be described in this document.

8.2.1. Requirements

8.2.1.1. Hardware Requirements

Insightio operates optimally with hardware specifications such as an NVIDIA GPU supporting CUDA technology, coupled with at least 4GB of RAM for efficient data processing. Additionally, a multi-core CPU with a minimum of 2.0 GHz clock speed ensures smooth execution of computational tasks. By meeting these hardware requirements, users can expect seamless analysis of surveillance data and enhanced performance when utilizing Insightio for their monitoring needs.

8.2.1.2. Software Requirements

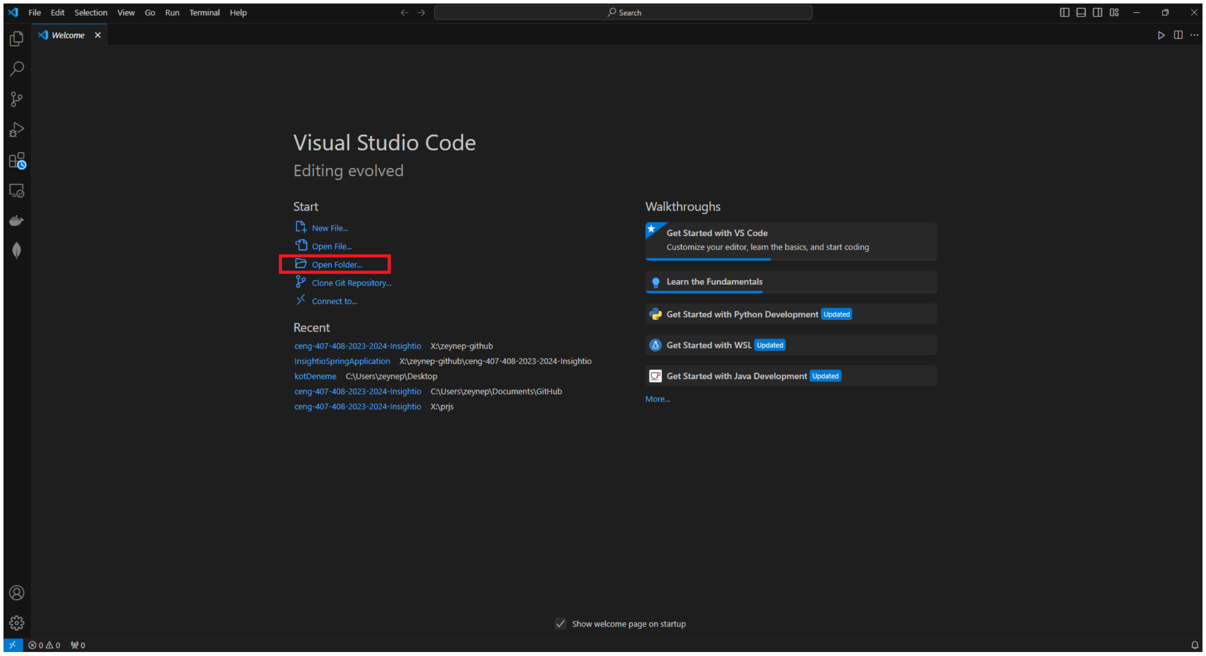
Insightio is only compatible with Windows 10 and WIndows 11 operating systems. This choice is deliberate, as older operating systems are no longer supported by the software packages utilized alongside Insightio.

8.3. Preparation for Installing the Application

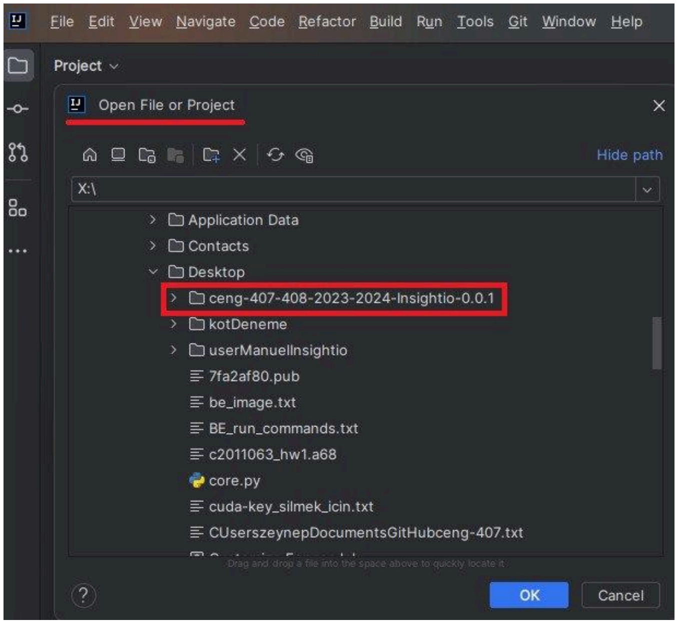
- Kotlin and Electron Interpreters are required. IntelliJ IDEA for Kotlin and Visual Studio Code for Electron are recommended.
- MongoDB and Postman required for creating the initial user and configuring the camera settings through API interactions.
- Python 3.11 is required. Please refer to README.md in the InsightioTracker directory for installation instructions and additional resources.
- Setting up CUDA

8.4. Getting Started

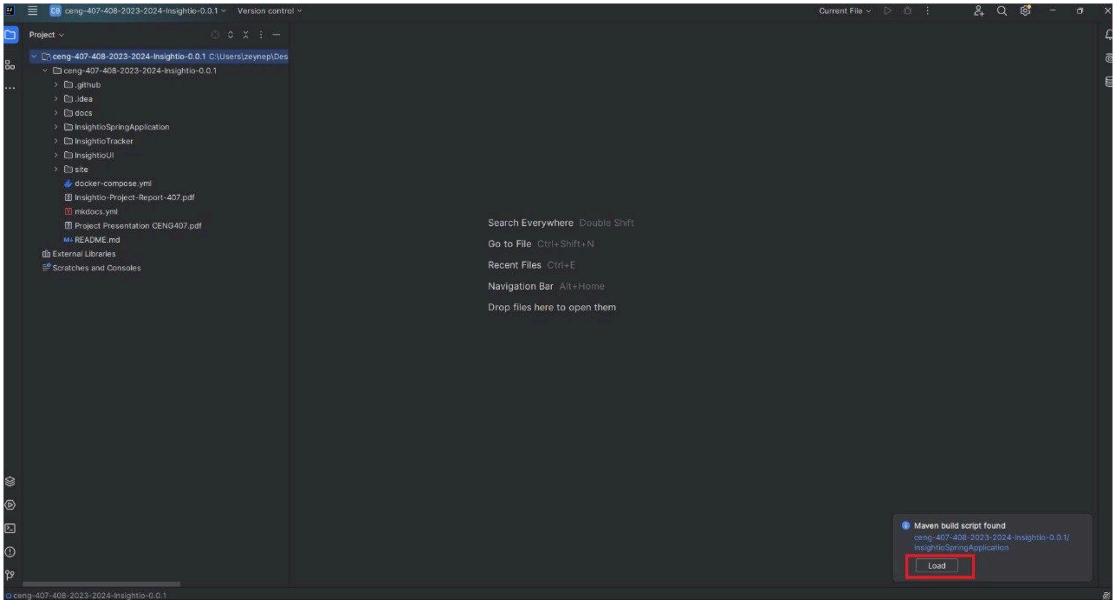
- Download the Insightio at the GitHub page. The zip file is the source code and base documents of Insightio.
- For setting up InsightioTracker and InsightioUI, it is recommended to use Visual Studio Code.


File Edit Selection View Go Run Terminal Help
Search
Welcome
Visual Studio Code
Editing evolved
Start
New File...
Open File...
Open Folder...
Clone Git Repository...
Connect to...
Recent
ceng-407-408-2023-2024-Insightio X:\zeynep-github
InsightioSpringApplication X:\zeynep-github\ceng-407-408-2023-2024-Insightio
kotDeneme C:\Users\zeynep\Desktop
ceng-407-408-2023-2024-Insightio C:\Users\zeynep\Documents\GitHub
ceng-407-408-2023-2024-Insightio X:\prjs
Walkthroughs
Get Started with VS Code
Customize your editor, learn the basics, and start coding
Learn the Fundamentals
Get Started with Python Development Updated
Get Started with WSL Updated
Get Started with Java Development Updated
More...
Show welcome page on startup

- Follow the instructions in the InsightioTracker's README.md to set up the virtual environment, install dependencies, and configure PyTorch.
- Setting up UI
- Open IntelliJ and run the server.

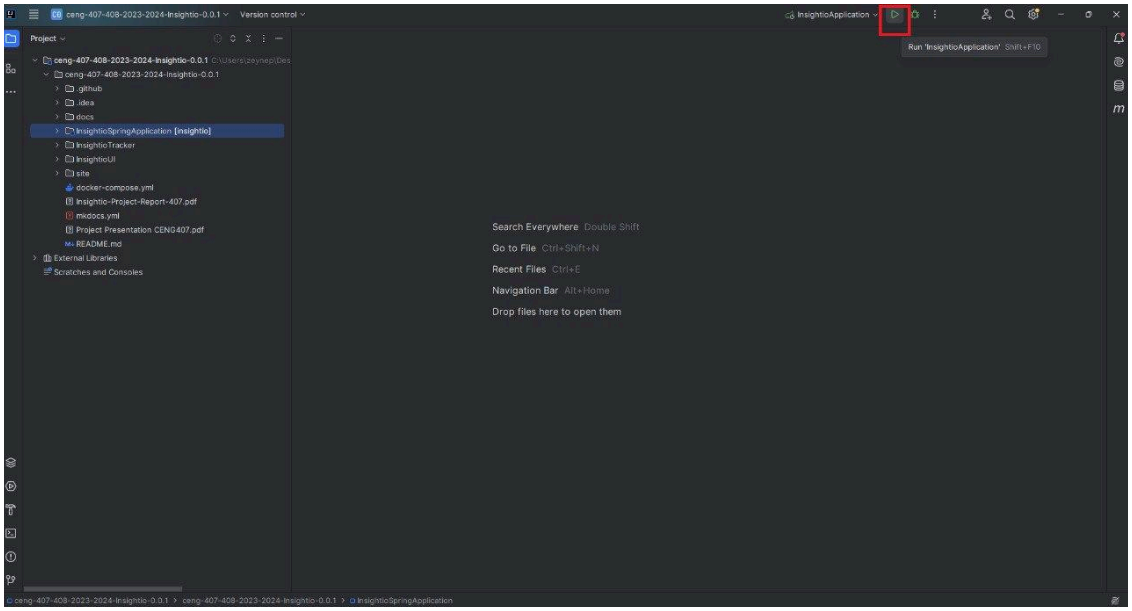

File Edit View Navigate Code Refactor Build Run Tools Git Window Help
Project
Open File or Project
Hide path
X:\
Application Data
Contacts
Desktop
ceng-407-408-2023-2024-Insightio-0.0.1
kotDeneme
userManuelInsightio
7fa2af80.pub
be_image.txt
BE_run_commands.txt
c2011063_hw1.a68
core.py
cuda-key_silmek_icin.txt
CUserszeynepDocumentsGitHubceng-407.txt
Drag and drop a file into the space above to quickly locate it
?
OK
Cancel

- Load Maven dependencies from IntelliJ.

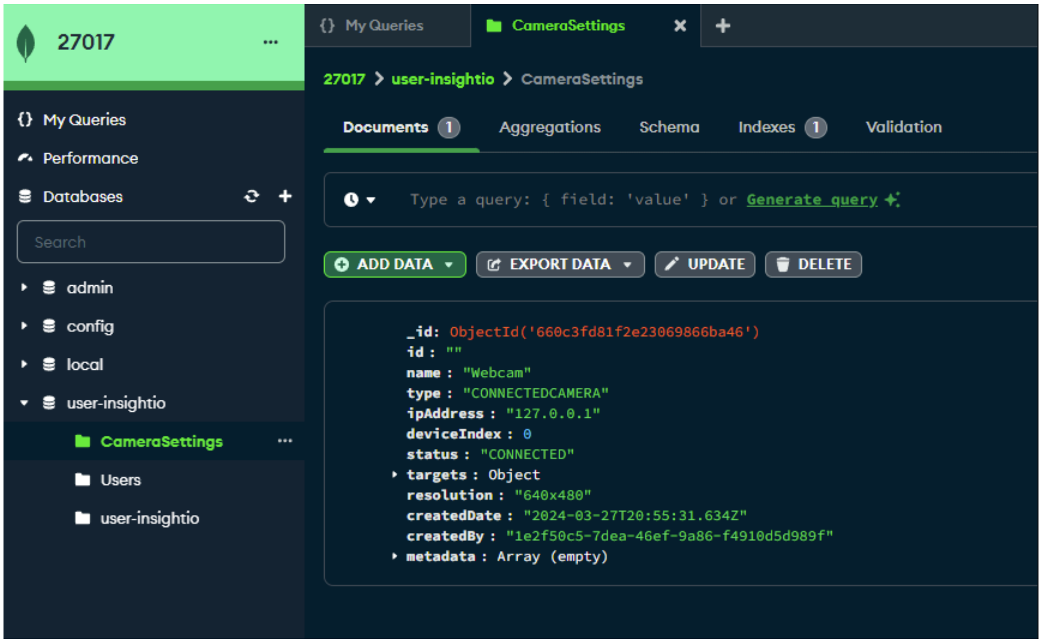

ceng-407-408-2023-2024-Insightio-0.0.1
Version control
Current File
Project
ceng-407-408-2023-2024-Insightio-0.0.1 C:\Users\zeynep\Des
ceng-407-408-2023-2024-Insightio-0.0.1
.github
.idea
docs
InsightioSpringApplication
InsightioTracker
InsightioUI
site
docker-compose.yml
Insightio-Project-Report-407.pdf
mkdocs.yml
Project Presentation CENG407.pdf
README.md
External Libraries
Scratches and Consoles
Search Everywhere Double Shift
Go to File Ctrl+Shift+N
Recent Files Ctrl+E
Navigation Bar Alt+Home
Drop files here to open them
Load

Maven build script found
ceng-407-408-2023-2024-Insightio-0.0.1/InsightioSpringApplication

- Run Server.

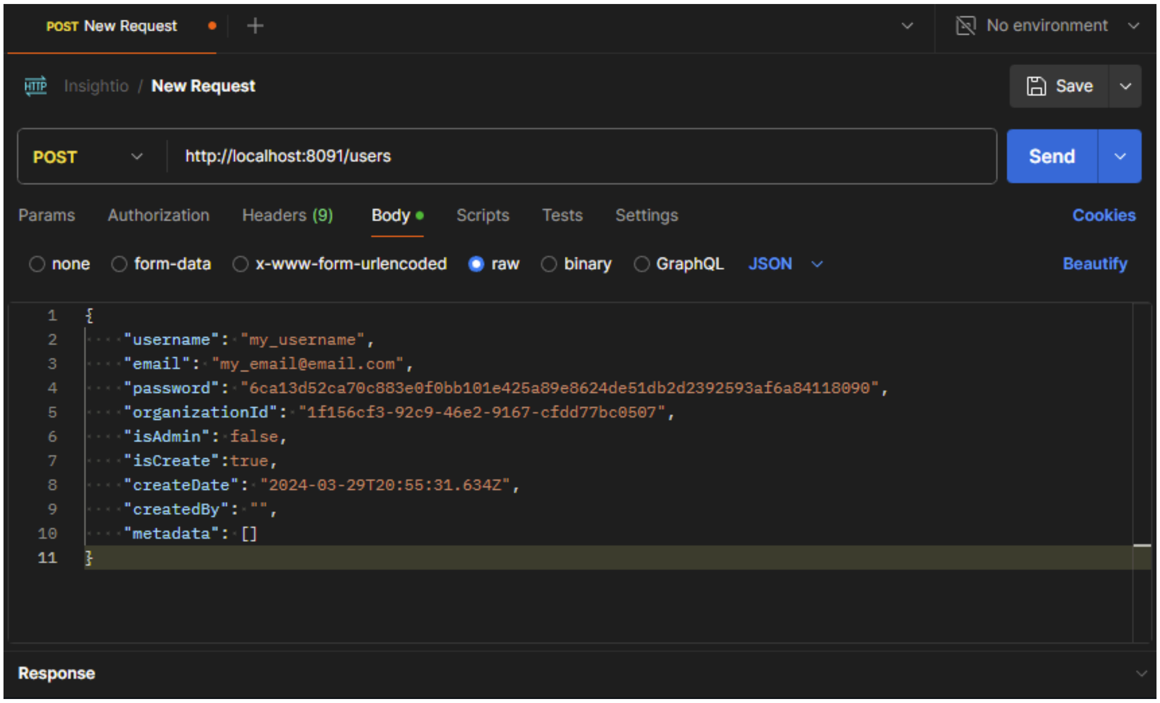

ceng-407-408-2023-2024-Insightio-0.0.1
Version control
InsightioApplication
Run 'InsightioApplication' Shift+F10
Project
ceng-407-408-2023-2024-Insightio-0.0.1 C:\Users\zeynep\Des
ceng-407-408-2023-2024-Insightio-0.0.1
.github
.idea
docs
InsightioSpringApplication [insightio]
InsightioTracker
InsightioUI
site
docker-compose.yml
Insightio-Project-Report-407.pdf
mkdocs.yml
Project Presentation CENG407.pdf
README.md
External Libraries
Scratches and Consoles
Search Everywhere Double Shift
Go to File Ctrl+Shift+N
Recent Files Ctrl+E
Navigation Bar Alt+Home
Drop files here to open them
ceng-407-408-2023-2024-Insightio-0.0.1 > ceng-407-408-2023-2024-Insightio-0.0.1 > InsightioSpringApplication

- Add MongoDB Configuration.
  - Connect to localhost:27017 in MongoDB Compass.
  - Create a database called "user-insightio".
  - Create 3 separate folders called: "CameraSettings", "Users", and "user-insightio".
  - Create an object as given below without the ID.



```
_id: ObjectId('660c3fd81f2e23069866ba46')
id : ""
name : "Webcam"
type : "CONNECTEDCAMERA"
ipAddress : "127.0.0.1"
deviceIndex : 0
status : "CONNECTED"
targets : Object
resolution : "640x480"
createdDate : "2024-03-27T20:55:31.634Z"
createdBy : "1e2f50c5-7dea-46ef-9a86-f4910d5d989f"
metadata : Array (empty)
```

- To enter your login information using Postman, proceed as follows:
  - Open Postman and navigate to the 'New Request' page.
  - Select 'POST' as the method and use the URL: http://localhost:8091/users.
  - In the 'Body' tab, select 'raw' and choose 'JSON' from the dropdown menu.
  - Input the login credentials as demonstrated.



POST http://localhost:8091/users

```
{
    "username": "my_username",
    "email": "my_email@email.com",
    "password": "6ca13d52ca70c883e0f0bb101e425a89e8624de51db2d2392593af6a84118090",
    "organizationId": "1f156cf3-92c9-46e2-9167-cfdd77bc0507",
    "isAdmin": false,
    "isCreate":true,
    "createDate": "2024-03-29T20:55:31.634Z",
    "createdBy": "",
    "metadata": []
}
```

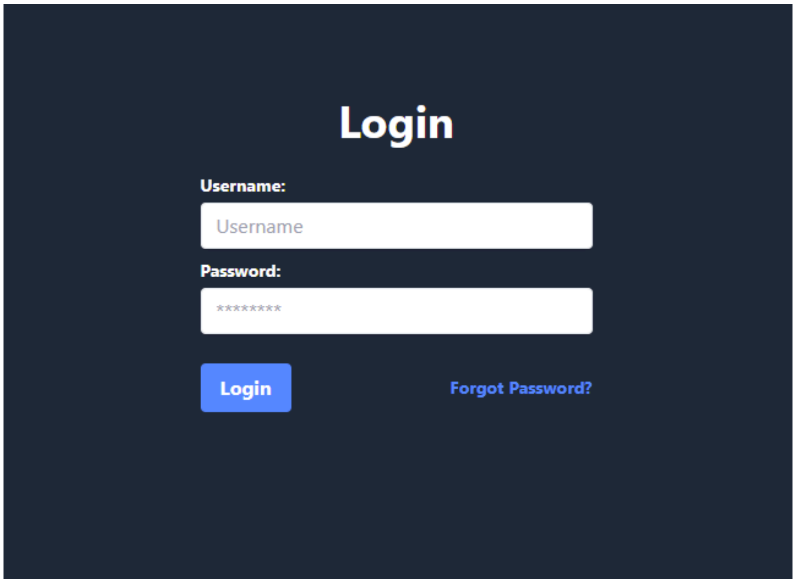
- To run UI, it is recommended to use VS Code. Execute the "yarn dev" command.

8.5. Using Insightio

Simple steps to help using the Insightio platform effectively.

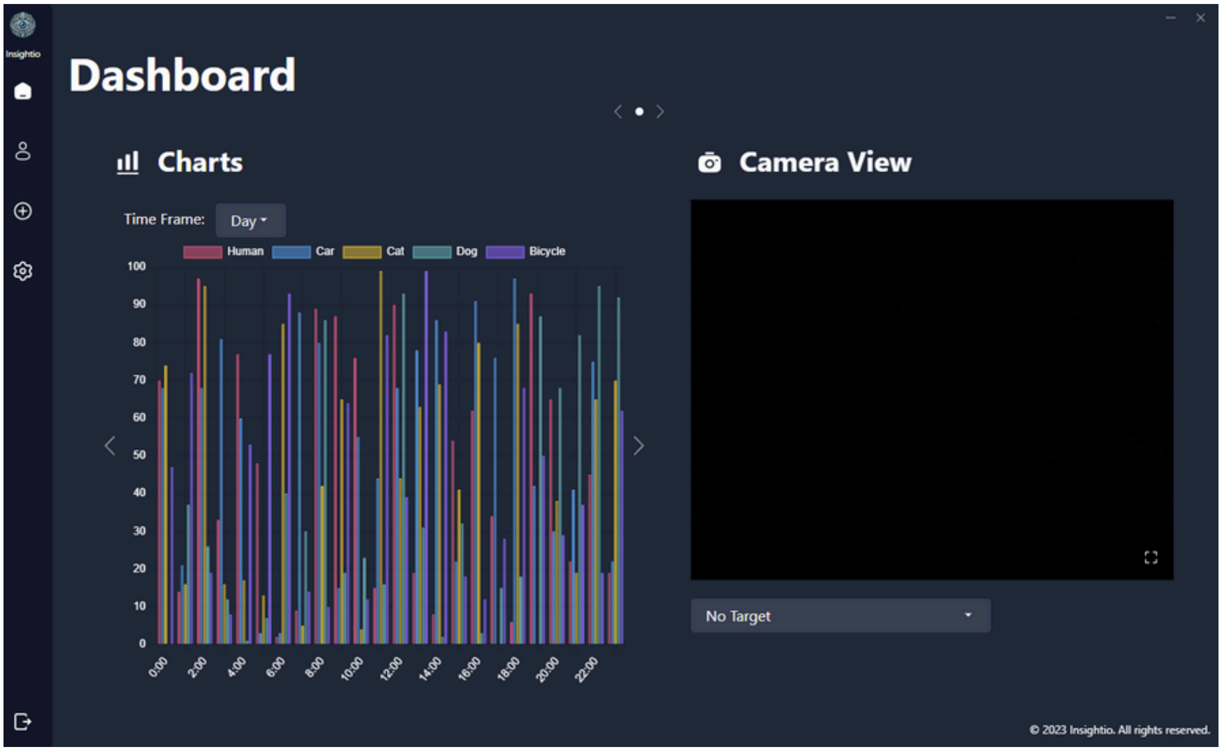
8.5.1 Login

Enter your username and password, upon successful authentication, you will gain access to the application.



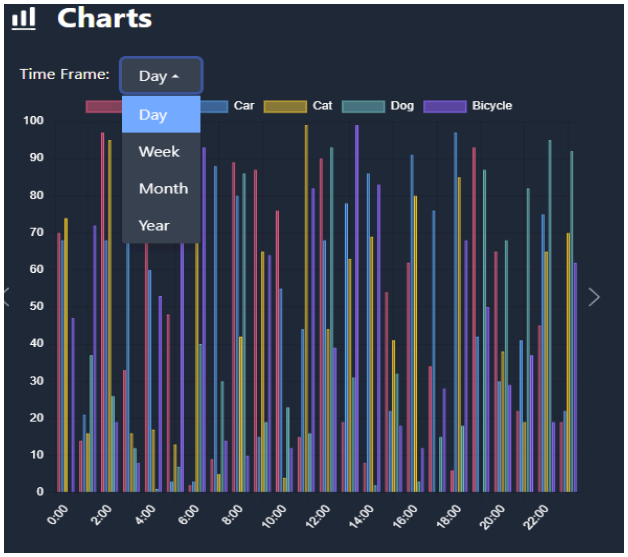
8.5.2 Main Dashboard

The main page of the application serves as a central hub for various functionalities, including camera selection, target selection, and chart viewing.



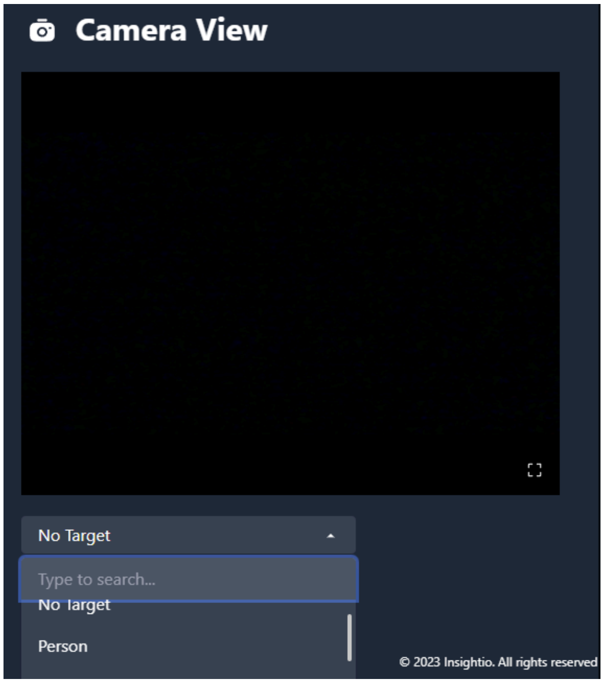
8.5.3 Select Time Frame

You can utilize the drop-down menu to choose a time frame for the chart.



8.5.4 Select Target

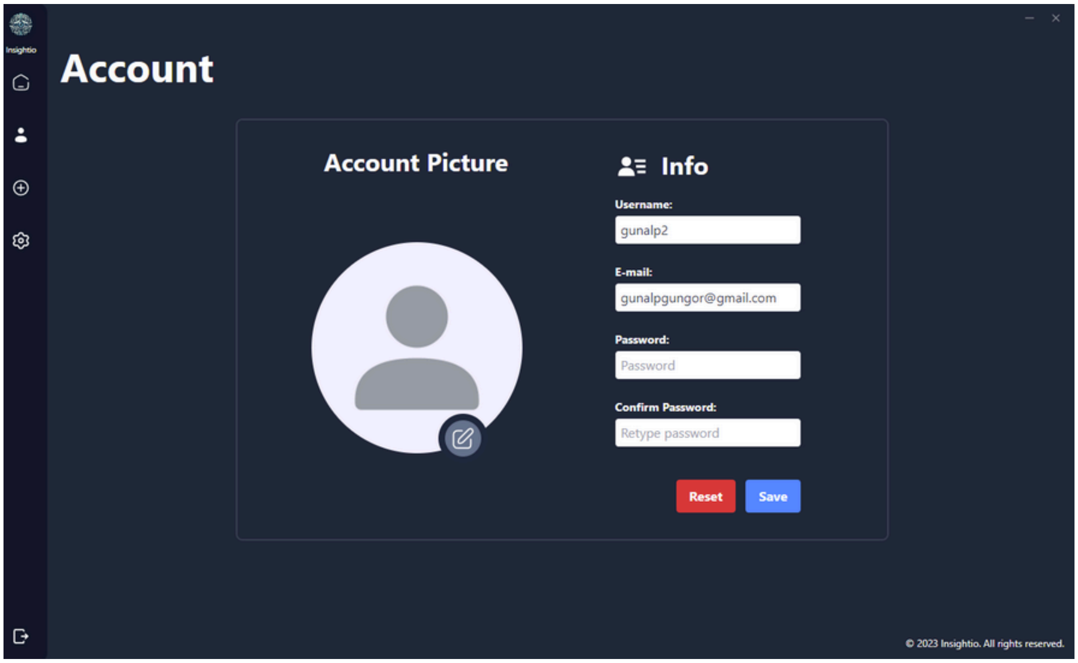
Select target objects to count.



No Target

8.5.5 Account Page

Users can update their account information using this page.



Username: gunalp2

E-mail: gunalpgungor@gmail.com

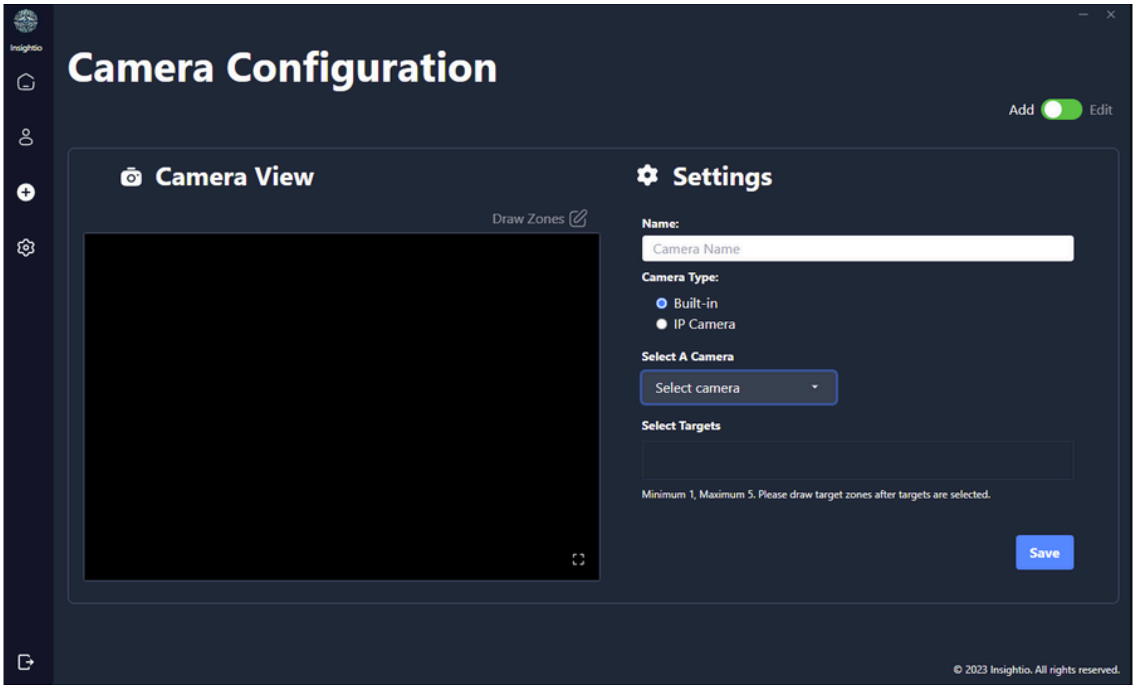
Password: Password

Confirm Password: Retype password

8.5.6 Configure Camera settings

Within the camera configuration page, users have the capability to manage cameras, whether built-in or IP, and customize tracking parameters. This includes:

- Adding or Editing Cameras
- Target Selection for Counting
- Drawing Zones for Targets



Camera Configuration

Camera View

Settings

Name: Camera Name

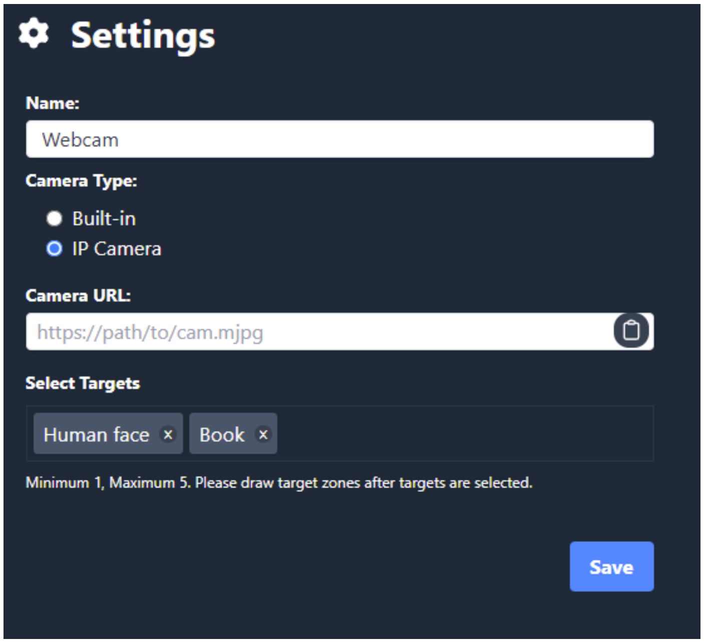
Camera Type:
● Built-in
○ IP Camera

Select A Camera: Select camera

Select Targets

Minimum 1, Maximum 5. Please draw target zones after targets are selected.

© 2023 Insightio. All rights reserved.



Settings

Name: Webcam

Camera Type:
○ Built-in
● IP Camera

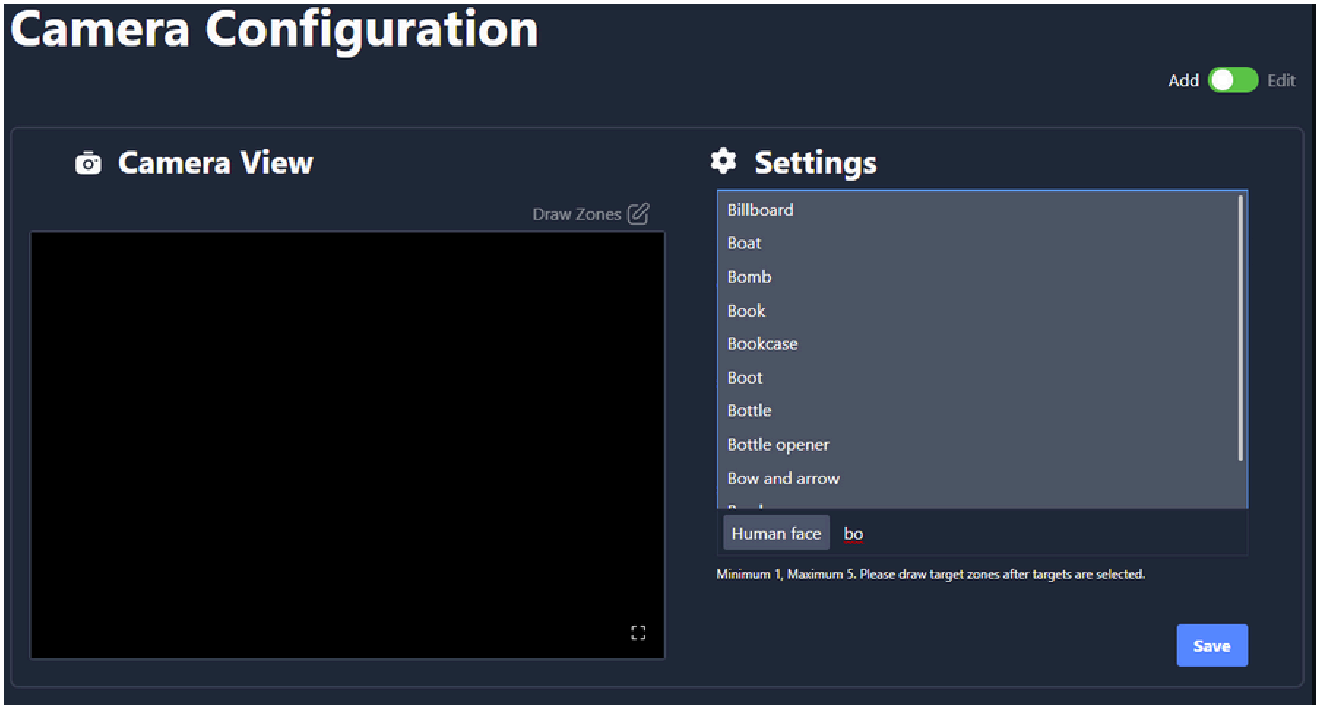
Camera URL: https://path/to/cam.mjpg

Select Targets: Human face × Book ×

Minimum 1, Maximum 5. Please draw target zones after targets are selected.

8.5.6.1 Choose Targets

Users can designate specific targets they wish to count within camera feeds.



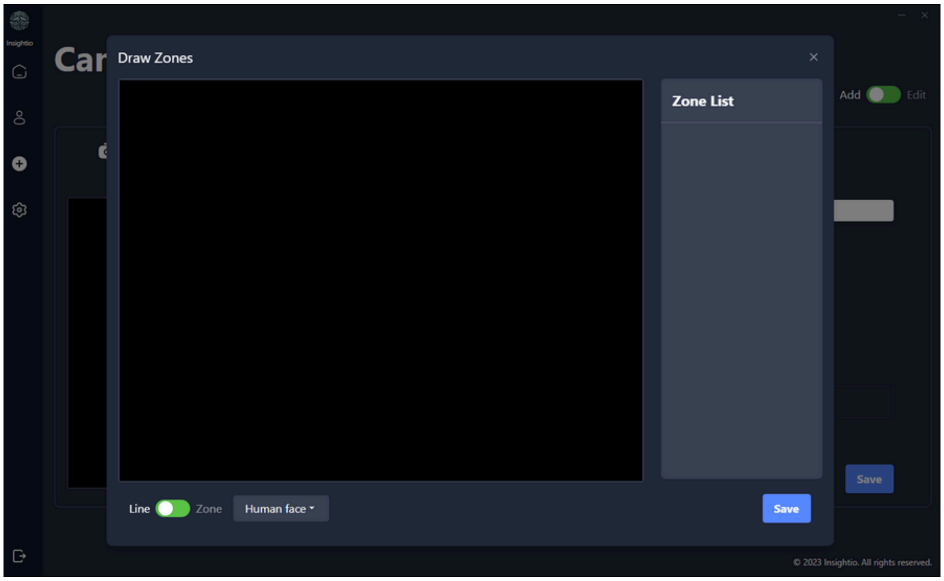
Camera Configuration

Human face bo

Minimum 1, Maximum 5. Please draw target zones after targets are selected.

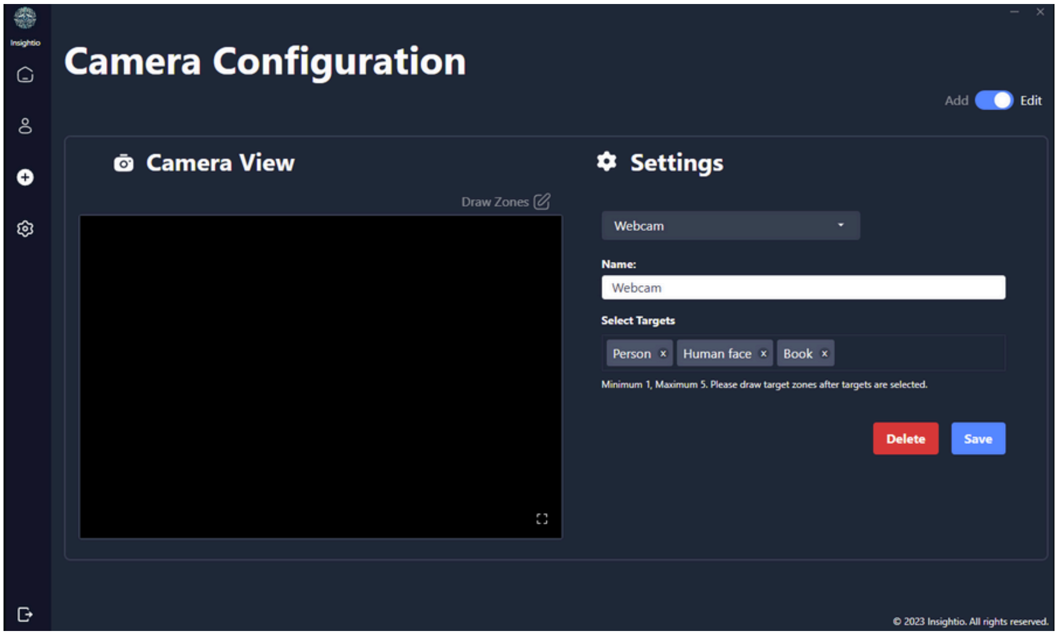
8.5.6.2 Choose Draw Zones

Users can define zones within camera feeds for target tracking purposes. These zones can be delineated as either lines or rectangles.



8.5.6.3 Add/Edit Camera

Users can add new cameras to the system or modify existing cameras whether they are built-in or IP cameras.



Camera Configuration

Add / Edit

Camera View

Settings

Webcam

Name: Webcam

Select Targets

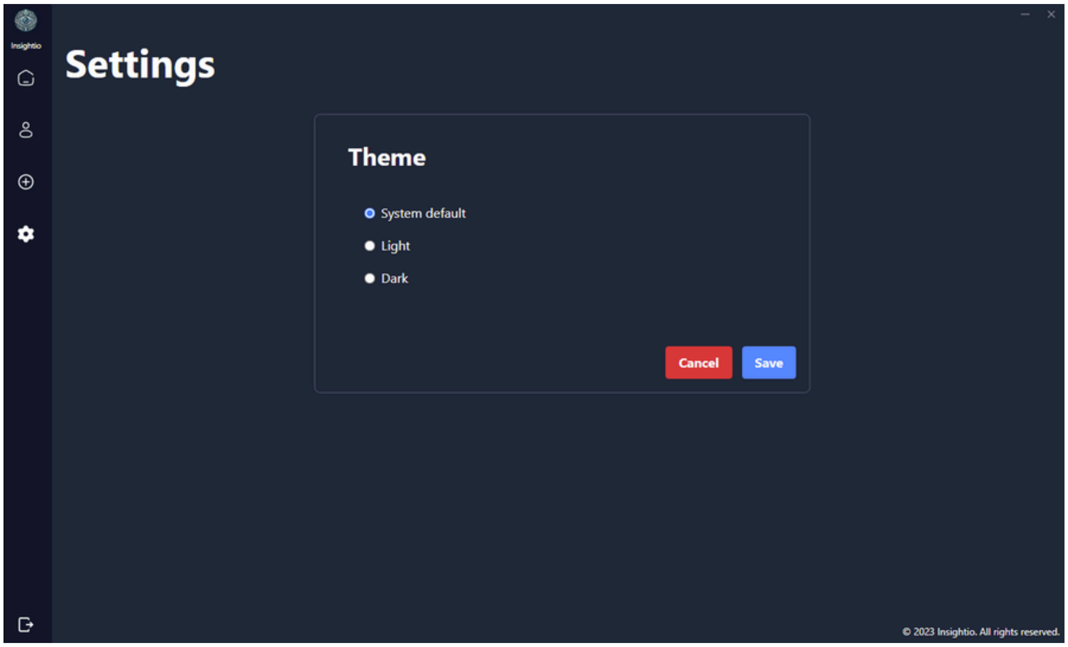
Person × Human face × Book ×

Minimum 1, Maximum 5. Please draw target zones after targets are selected.

© 2023 Insightio. All rights reserved.

8.5.7. Settings

Users can change the system theme as Light, Dark or System Default.



Settings

Theme

● System default

● Light

● Dark

© 2023 Insightio. All rights reserved.

9. Conclusion

This document provides a step-by-step overview of the overall structure of the Insightio project, offering a comprehensive insight into the platform. The literature review highlights the prominent features of the platform based on current information. The Software Requirements Specification (SRS) clearly defines the purpose and requirements of the project, while the Software Design Document (SDD) elaborates on the technical details of the platform based on these requirements.Additionally, the Test Plan and Test Results provide a detailed evaluation of the system’s performance and functionality, and the User Manual offers guidance for end-users.

In conclusion, the Insightio document serves as a comprehensive resource for the project team, developers, and stakeholders to understand the general design and functionality of the platform. The literature review goes beyond existing information, emphasizing the innovative features of the platform. The SRS and SDD sections establish a clear foundation for successful project management by defining the project's goals and technical requirements.The Test Plan and Test Results sections ensure that the platform meets its performance and functionality criteria, while the User Manual ensures a smooth user experience.

This document guides the development of the Insightio platform, providing the project team with both a functional and technical perspective, contributing to a successful implementation process.

10. References

1. Bhangale, U., Patil, S., Vishwanath, V., Thakker, P., Bansode, A., & Navandhar, D. (2020). Near Real-time Crowd Counting using Deep Learning Approach. Procedia Computer Science, 171, 770–779. https://doi.org/10.1016/j.procs.2020.04.084
2. Crowd Models for Emergency Evacuation: A review Targeting Human-Centered Sensing. (2013, January 1). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/abstract/document/6479853
3. Chakraborty, A. (2023). How People Counting Systems Enhance Customer Experience In Retail Stores. LinkedIn. https://www.linkedin.com/pulse/how-people-counting-systems-enhance- customer-retail-chakraborty/
4. Huang, C., & Chung, C. (2004). A Real-Time Model-Based human motion tracking and analysisfor Human-Computer Interface systems. EURASIP Journal on Advances in Signal Processing, 2004(11). https://doi.org/10.1155/s1110865704401206
5. Hussain, M. (2023). YOLO-v1 to YOLO-v8, the Rise of YOLO and Its Complementary Nature toward Digital Manufacturing and Industrial Defect Detection. Machines, 11(7), 677. https://doi.org/10.3390/machines11070677
6. People Counting System Market Size, Share & Trends Analysis Report By End-use (BFSI, Corporate), By Offering (Hardware, Software), By Mounting Platform (Ceiling, Wall), By Type, By Technology, And Segment Forecasts, 2022 - 2030. (n.d.). https://www.grandviewresearch.com/industry-analysis/people-counting-system-market-report
7. Raghavachari, C., Aparna, V., Chithira, S., & Balasubramanian, V. (2015). A Comparative Study of Vision Based Human Detection Techniques in People Counting Applications. Procedia Computer Science, 58, 461–469. https://doi.org/10.1016/j.procs.2015.08.06

8. Rahman, M. A. (2017). Computer Vision Based Human Detection. International Journal of Engineering and Information Systems (IJEAIS), 1(5), 62-85. https://hal.science/hal 01571292/document
9. Reis J. (2014). *Image descriptors for counting people with uncalibrated cameras* [Master Thesis,University of Porto]. U. Porto. https://paginas.fe.up.pt/~ee08266/thesis.html
10. Sam, D. B., Peri, S. V., Sundararaman, M. N., Kamath, A., & Babu, R. V. (2020). Locate, size and count: Accurately resolving people in dense crowds via detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1. https://doi.org/10.1109/tpami.2020.2974830
11. Sjöberg, A. & Hyberg, J. (2023). *Investigation regarding the Performance of YOLOv8 in Pedestrian Detection* [Bachelor's Thesis, KTH Royal Institute Of Technology]. Diva Portal. https://kth.diva-portal.org/smash/get/diva2:1778368/FULLTEXT01.pdf