Requirements Specification Document

Project Name: Flux - Real-Time People Flow Monitoring Platform

Version: 1.1

Author: FLUX

Date: 2024.10.25

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this project, "Flux - Real-Time People Monitoring Platform," is to provide a comprehensive and reliable solution for monitoring people flow in various public settings, ensuring safety, and enhancing management efficiency. Flux aims to clearly define the functional and non-functional requirements necessary for the project's success, including system features, user interactions, and constraints, serving as a foundational blueprint throughout the software lifecycle.

This platform will act as the basis for ensuring all stakeholders—developers, testers, and users—have a consistent understanding of what the project entails, aligning expectations and ensuring cohesive progress. It establishes clear benchmarks to measure progress during development and deployment phases, enhancing communication between all participants.

The platform aims to support development and testing processes by offering specific criteria to validate the software's reliability, usability, and overall functionality. Ultimately, the project's purpose is to create a user-friendly, effective, and well-integrated people flow monitoring solution that meets the intended goals of enhancing public safety and improving operational efficiency.

1.2 Project Background

With the ever-increasing demands for public safety, crowd control, and operational efficiency, the need for real-time monitoring of people flow in various public settings has grown significantly. Places such as shopping malls, campuses, airports, events, and other environments often experience high foot traffic that needs to be properly monitored to prevent hazards, maintain a safe atmosphere, and facilitate smooth management. "Flux" is designed to address these concerns by providing a multi-camera monitoring solution that offers comprehensive real-time insights, intuitive data visualization, and valuable analytical data.

Flux aims to empower decision-makers with actionable information by helping them understand the dynamics of people movement, predict crowd behavior, identify bottlenecks, and optimize resource allocation. By enabling such data-driven decisions, Flux ultimately strives to enhance safety, optimize resource allocation, and create an efficient public environment for visitors and staff alike.

1.3 Reference Materials

- **Related academic papers on crowd management and real-time monitoring technologies**
- [Pytorch documentation]

- Springboot decumentation
- Vue.js Official Documentation

2. Overall Description

2.1 Objectives

The primary goal of Flux is to enable effective, real-time monitoring of people flow in various public settings, ultimately enhancing safety, management, and efficiency. Flux is designed to capture, process, and visualize people flow data by utilizing multiple cameras that can monitor multiple locations simultaneously. The system processes this data in real-time and delivers insights through an intuitive interface, making it accessible and actionable for end-users.

Flux aims to allow decision-makers, such as mall managers, campus administrators, or event organizers, to gain instant visibility into people flow. This visibility helps prevent unsafe situations due to overcrowding, enables the efficient deployment of resources (like security or cleaning personnel), and assists management in understanding traffic patterns, allowing them to make informed decisions. Ultimately, the goal is to provide an easy-to-use platform that delivers robust capabilities for crowd analysis, improving safety standards and operational efficiency.

2.1.1 Development Intention

Flux is intended to be a highly user-friendly monitoring solution that can be integrated into existing infrastructure with minimal friction. The system is designed with flexibility and ease of use at its core, making it simple to incorporate into a variety of environments, regardless of the scale of operation. Flux facilitates dynamic camera integration, allowing users to easily add, remove, or replace cameras as needed without significant system reconfiguration.

The system provides flexible options for data visualization, enabling users to view data in formats that best suit their needs, such as heat maps, trend charts, or tabular reports. Alerts are customizable, allowing users to define thresholds and conditions based on their individual safety and management requirements. Additionally, role-based user management ensures that data and functionalities are accessible only to authorized personnel, thereby providing privacy and security.

By emphasizing integration, flexibility, and user-centric design, Flux serves as a powerful yet adaptable tool for managing people flow in any public setting.

2.1.2 Application Goals and Scope

- **Real-Time Monitoring of People Flow:** Flux provides real-time monitoring of people flow in crowded environments such as shopping malls, campuses, airports, event centers, and other public spaces. The system continuously collects and analyzes data from multiple camera feeds, offering users live insights into people density, flow rate, and potential bottlenecks.

- **Statistical Analysis:** Flux delivers advanced statistical analysis of people flow, including key metrics such as entry and exit rates, crowd density at specific locations, and trends over time. Users have access to historical data presented in insightful formats, such as daily, weekly, or monthly comparisons. This data is crucial for decision-makers seeking to understand long-term trends, identify peak times, and make evidence-based improvements to crowd management strategies.
- **Features like Real-Time Alerts, Role-Based Access, and Historical Data Analysis**: Flux provides advanced features including real-time alerts, role-based user access, and historical data analysis. Alerts notify users when specific thresholds, such as overcrowding, are reached—enabling immediate action to prevent unsafe situations. Role-based access ensures that only authorized personnel can view or interact with sensitive data, maintaining data security and privacy. Historical data analysis allows users to compare data over time to understand behavior patterns and make better future decisions.
- **Application Areas:** Flux is applicable across various scenarios, including:
    - **Shopping Malls:** Monitor foot traffic, ensure efficient use of space, and improve visitor experiences.
    - **Campuses:** Help university administrators track people flow and ensure student safety, especially during large events or gatherings.
    - **Event Centers:** Prevent overcrowding and ensure safety at large-scale events.
    - **Public Safety Zones:** Monitor crowded areas such as transportation hubs, ensuring compliance with safety norms.
    - **Other Public Spaces:** Any area requiring effective people flow management to enhance both safety and operational efficiency.

2.1.3 Product Prospects

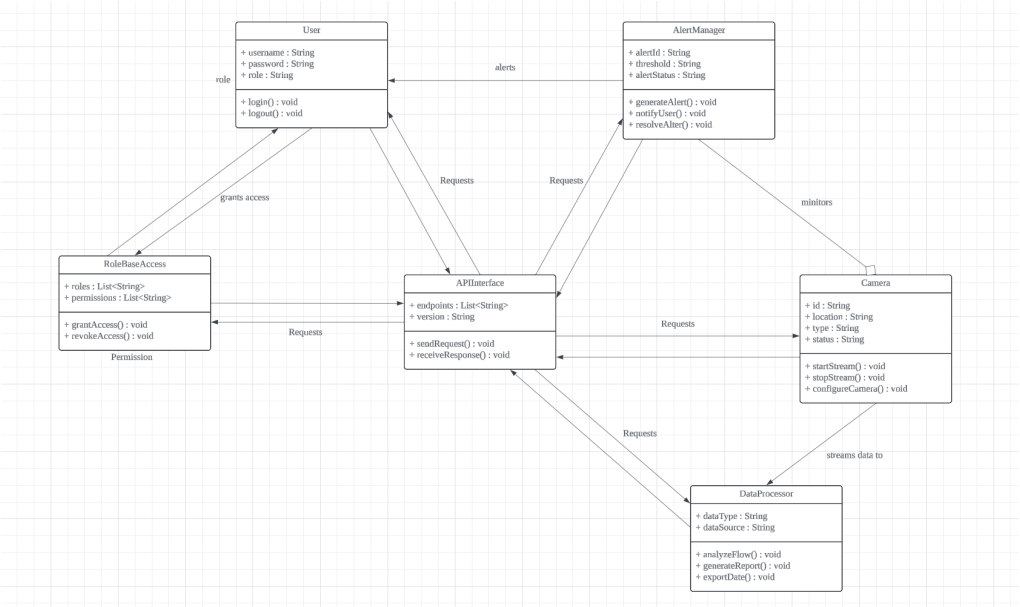
Flux comprises front-end and back-end components that work together seamlessly to provide a cohesive monitoring solution.

- **Front-End Component:** This is a web-based user interface that allows users to interact with the real-time data gathered by the system. It offers visualizations such as heatmaps, crowd flow diagrams, and tabular data that are easy to understand, enabling users to make quick, informed decisions. Users can access the interface via standard web browsers on desktops, tablets, or mobile devices. The interface is designed for ease of navigation and includes interactive elements that enhance the user experience.
- **Back-End Component:** The back-end handles the core data processes of Flux. It is responsible for collecting video data from cameras, processing it using machine learning algorithms, and generating actionable insights. The back-end manages data storage and retrieval of historical data and implements alert mechanisms. The architecture of the back-end ensures system reliability, scalability, and security, allowing the system to function effectively as new monitoring points or additional cameras are added. This scalability is crucial for expanding the system from a pilot installation to larger-scale

deployments. Furthermore, the platform is designed with modularity, meaning that components can be upgraded or extended without affecting the entire system—enabling continuous improvement and adaptability as requirements evolve.

3. Specific requirements

3.1 Class Diagram



This diagram shows the core architecture of the Flux real-time personnel monitoring platform, covering six main classes: user management, camera integration, alarm generation, data processing, permission control and API communication. Each module collaborates with each other to build an efficient and flexible real-time monitoring system.

3.1.1 User class

- The User class represents the users in the system, each with username, password, and role attributes. Users log in and log out of the system and access different system functions based on their roles. The user's role (e.g., admin, observer) determines which system modules they can use. To ensure secure and flexible access to functions within the system. Through role control, users can obtain functions that match their permissions, such as camera management and alarm monitoring. These permissions are managed by the RoleBasedAccess class.

3.1.2 RoleBasedAccess class

- The RoleBasedAccess class manages user permissions and defines the access permissions of different roles in the system. RoleBasedAccess uses the grantAccess() and revokeAccess() methods to ensure that users can access only functional modules corresponding to their roles. It is interlinked with the User class, ensuring that users can safely operate system functions based on their roles, and that the permissions of each user are managed dynamically.

3.1.3 Camera class

- The Camera class is used to represent individual cameras in a monitoring system, each with a unique ID, location, type, and status. The video stream collected by the camera is managed by startStream() and stopStream() methods, and sent to the background through the APIInterface class of the system for data interaction and real-time data stream transmission. Multiple cameras integrate and transmit data at the same time, ensuring the real-time monitoring of multiple areas.

3.1.4 DataProcessor class

- The DataProcessor class is responsible for receiving data from the Camera class and performing real-time analysis of the personnel flow data captured by those cameras. Its core features include analyzeFlow() to analyze the flow data collected by the camera in real time, generateReport() to generate historical data reports, and export the analysis results through the exportData() method. The DataProcessor class is closely connected with the Camera class, and the camera data flows directly to the data processing module for further analysis and processing.

3.1.5 AlertManager class

- The AlertManager class is responsible for generating alerts based on camera data and threshold conditions set by the system. It generates an alert when an exception is detected through the generateAlert() method and notifyUser() method, ensuring that the user receives the alert at the critical moment. Associated with the Camera and User classes, it monitors camera data and is responsible for sending alerts to users. After receiving the alert, the user can take further action through the system.

3.1.6 APIInterface class

- The APIInterface class is the communication bridge between the front and back ends of the system. It is responsible for sending the user's requests (such as viewing camera data, generating reports, etc.) to the back-end module via the sendRequest() method and returning the processing results to the front end via the receiveResponse() method. This type of interface standardizes the data transmission format and ensures the reliability and consistency of the data exchange between the front-end and back-end. At the same time, this class supports version control, which facilitates future expansion and upgrade of the system.

It can be clearly seen from this diagram that the modules of the system work together under the unified API framework, which ensures the real-time data, security and scalability of the system. At the same time, the class diagram also shows how the system improves the intelligence and practicability of monitoring through function modules such as role authority control, alarm generation, data analysis and visualization, and finally realizes efficient and intelligent public safety management.

3.2 Properties

3.2.1 Multi-Camera Integration

- **Description:** Flux allows seamless integration and management of multiple cameras to monitor different areas within a public setting.
- **Requirements:**
  - Users must be able to add, configure, and remove cameras through the user interface.
  - The system should support various camera types and protocols to ensure compatibility.
  - Cameras should stream data for both live monitoring and recording purposes.
  - The system must handle simultaneous feeds from multiple cameras without performance degradation.

3.2.2 Real-Time Monitoring and Alerts

- **Description:** Provides real-time analysis of people flow, with automated alerts for abnormal activities such as overcrowding.
- **Requirements:**
  - Real-time detection and counting of individuals within the camera's field of view.
  - Configurable alert thresholds based on crowd density, entry/exit rates, and other parameters.
  - Notifications should be sent through multiple channels, including the user interface, email, and SMS.
  - The system must ensure low latency in alert generation to enable timely responses.

3.2.3 Data Analysis and Visualization

- **Description:** Enables detailed analysis of people flow data with intuitive visual representations to aid decision-making.
- **Requirements:**
  - Provide statistical reports on people flow metrics such as entry/exit rates, density, and movement patterns.
  - Support for viewing historical data with options to filter by day, week, month, and custom date ranges.
  - Interactive visualizations including heatmaps, trend charts, and crowd flow diagrams.

- Export functionality for reports and data visualizations in common formats (e.g., PDF, CSV).

3.2.4 Role-Based Access

- **Description:** Ensures secure access to system features and data based on user roles and permissions.
- **Requirements:**
  - Define multiple user roles (e.g., Administrator, Manager, Viewer) with specific access rights.
  - Administrators can manage cameras, users, alerts, and system settings.
  - Managers can access monitoring data, generate reports, and respond to alerts.
  - Viewers can only access live monitoring feeds and view reports without modification rights.
  - Implement secure authentication mechanisms to verify user identities.

3.2.5 Search and Filtering

- **Description:** Facilitates efficient retrieval of specific data through advanced search and filtering options.
- **Requirements:**
  - Enable search functionality based on camera location, date, time, and specific events.
  - Provide filtering options based on parameters such as time range, camera ID, and detected crowd density.
  - Allow users to save custom search and filter settings for repeated use.
  - Ensure search and filtering operations are performant, even with large datasets.

3.3 Other Properties

3.3.1 An API interface defines the interaction with data

- **Description:** The purpose of API interface definition and data interaction is to standardize the communication between the front and back end and ensure the standardization of data transmission and processing
- **Requirements:**
- Standardized interfaces: API interactions between frontend and backend must follow a uniform interface standard, and all API request and response formats should be clearly defined.

- Interface configuration and versioning: Interfaces in the src/api directory should be clearly divided according to functional modules and supported for versioning to accommodate future interface updates.

3.3.2 Performance and compatibility requirements

- **Description:** The system needs to maintain high performance under high load conditions, and both front-end and back-end should be optimized to ensure page loading speed and service response time.
- **Requirements:**
- Frontend build tool and configuration: The frontend uses vite as build tool, and the configuration file vite.config.ts needs to be optimized to ensure build speed and product performance.
- Cross-browser compatibility: Front-end pages such as index.html and App.vue should be compatible with major browsers (e.g., Chrome, Firefox, Edge, etc.) to ensure normal display and interaction in different browsers.
- Database and middleware support: The configuration files in the back-end infra module need to support common database and middleware services and support future extensions and configuration changes.

3.3.3 Maintainability and Extensibility

- **Description:** The project design should be highly modular to enable code reuse and ease maintenance.
- **Requirements:**
- Code reuse and componentization: Business components in the frontend components directory should be designed in a modular way for easy reuse and maintenance. common utility classes and methods should be provided in the common directory of the backend to reduce code redundancy.
- Module extension and interface update: The system architecture should have good extensibility to allow new functional modules to be added in the future, and the front-end and back-end apis should support version control to avoid compatibility issues caused by updates.

4. Interface Prototype

The FLUXR Pedestrian Flow Detection System includes an intuitive and user-friendly interface that integrates various modules to ensure seamless operation and ease of use. Below is a detailed description of each module in the interface prototype:

4.1 Data Overview Module

The **Data Overview** module provides a comprehensive overview of pedestrian flow. Key components of the interface include:

- **Current Number of People**: This section displays the current count of individuals in the monitored area. It includes visual indicators for maximum capacity (see **Figure 1**).
- **Cumulative People Count**: Metrics for the current day, week, month, and cumulative totals are displayed prominently (see **Figure 1**).
- **Pie Chart Distribution**: A pie chart shows the distribution of individuals at different monitored areas on that day (see **Figure 1**).
- **Line Chart for Daily Flow**: The line chart below shows the pedestrian flow rate over time (see **Figure 1**).
- **Geographical Map**: The module integrates Amap to provide a visual representation of the geographic location being monitored (see **Figure 1**).

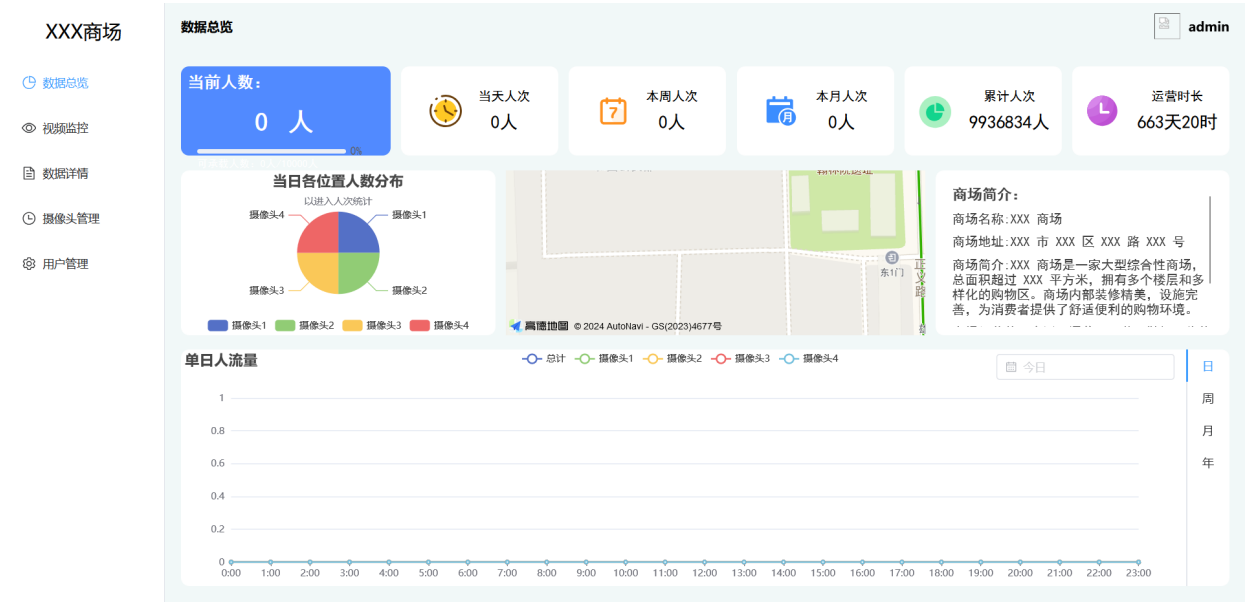


Figure 1. Data Overview Module

4.2 Video Surveillance Module

The **Video Surveillance** module shows real-time video streams from different cameras, enabling direct monitoring of pedestrian activities:

- **Camera Views**: The interface presents separate live feeds from each camera. This feature allows users to observe the flow of people in different areas at any given moment (see **Figure 2**).
- **Loading Indicators**: During stream loading, loading icons are displayed until the feed becomes available (see **Figure 2**).

Figure 2. Video Surveillance Module

4.3 Data Details Module

The **Data Details** module offers a detailed breakdown of pedestrian movement over various time periods:

- **Entry and Exit Information**: Each row displays the entry and exit counts (0 for departure, 1 for arrival) (see **Figure 3**).
- **Hourly Data**: The module provides detailed information on hourly pedestrian counts, facilitating in-depth analysis of inflow and outflow patterns (see **Figure 3**).

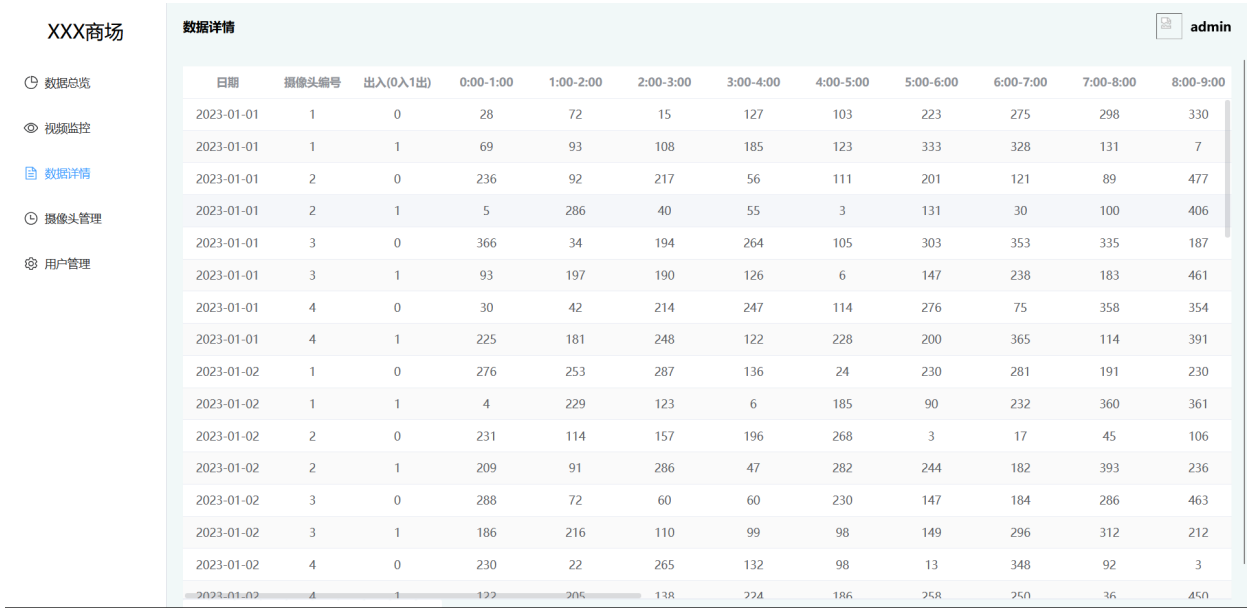


Figure 3. Data Details Module

4.4 Camera Management Module

The **Camera Management** module allows users to manage camera details:

- **Camera Information Display**: The interface shows the ID, name, IP address, and geographical coordinates (longitude and latitude) for each camera (see **Figure 4**).
- **Camera Operations**: Users have the option to modify or delete each camera's information (see **Figure 4**).
- **Add Camera Button**: A dedicated button allows easy addition of new cameras to the system (see **Figure 4**).

Figure 4. Camera Management Module

4.5 User Management Module

The **User Management** module provides the following functionalities:

- **User Information**: Displays user ID, username, password, role, phone, profile image, and additional notes (see **Figure 5**).
- **Add User Button**: Users can easily add new individuals to the system with assigned roles (see **Figure 5**).

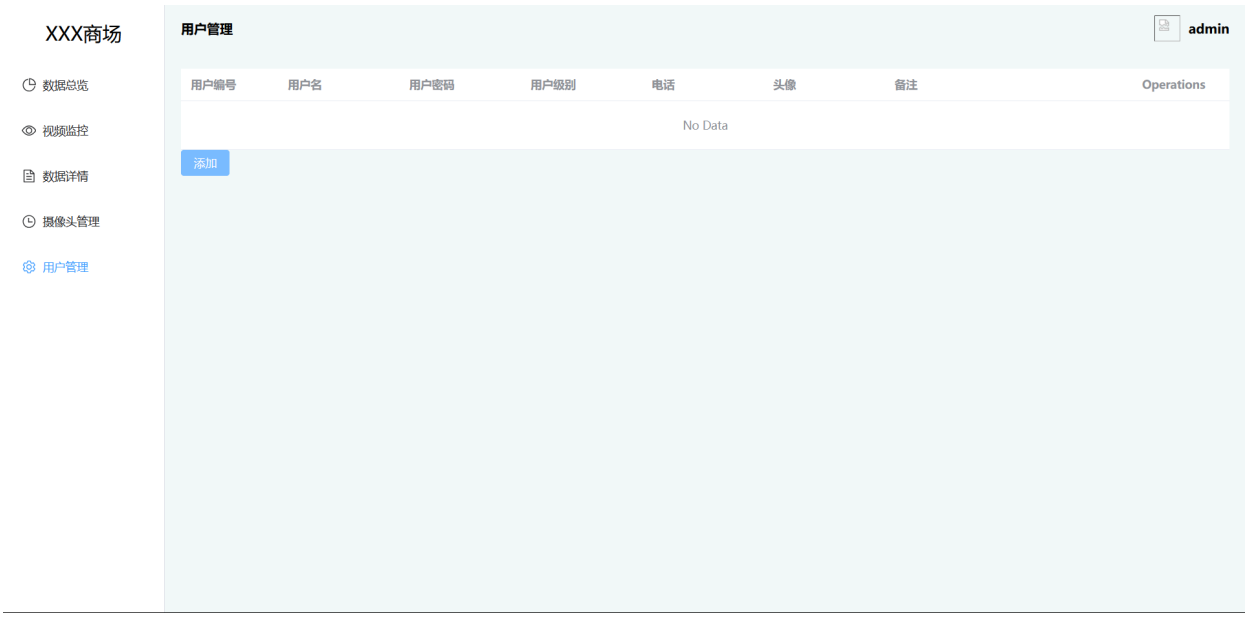


Figure 5. User Management Module

5. Function description and acceptance verification standards

5.1 Functional Modules Overview

5.1.1 Data Overview Module

The **Data Overview** module in the FLUXR Pedestrian Flow Detection System provides a comprehensive overview for monitoring and managing pedestrian flow in various areas. Users can view key metrics such as:

- Current number of individuals in the monitored area.
- Total daily, weekly, and monthly pedestrian counts.
- Cumulative count of individuals in real time.
- Operating hours of monitored areas.

To present the data visually, the system includes:

- A **pie chart** illustrating the distribution of individuals at different monitored locations throughout the day.
- A **line chart** depicting the trend of pedestrian flow over time.

Additionally, users can view the geographical location of monitored areas using **Amap**, which provides an intuitive map view. This module also displays brief information about monitored areas, enabling users to quickly grasp trends in pedestrian flow changes.

5.1.2 Video Surveillance Module

The **Video Surveillance** module allows users to monitor real-time video streams from cameras and obtain pedestrian flow data. The module includes:

- **Real-time display** of human flow dynamics.
- **Image recognition technology** to annotate individual persons in the frame, providing accurate statistics for analysis.

The module empowers managers by enabling precise monitoring and analysis of pedestrian movement and facilitating effective responses to detected changes.

5.1.3 Data Details Module

The **Data Details** module provides detailed records of pedestrian flow data for each camera, categorized by date and time. Key features include:

- Visibility into the **inflow and outflow** of pedestrian traffic for each camera.

- **Binary annotation system (0/1)** for flow direction:
  - **0** represents individuals leaving the monitored area.
  - **1** represents individuals entering the monitored area.

This simplified annotation allows users to effectively track the direction of pedestrian flow and analyze changes over time, providing deeper insight into people movement patterns.

5.1.4 Camera Management Module

The **Camera Management** module allows users to manage all camera-related information easily. Key functions include:

- **Recording detailed camera information**, such as:
  - Camera name
  - IP address
  - Geographic coordinates (longitude and latitude)
  - Note fields for additional details
- **Management features** to add, modify, or delete camera information as needed.

This module ensures all camera configurations are organized and accessible, supporting the effective operation of the surveillance system.

5.1.5 User Management Module

The **User Management** module employs **Role-Based Access Control (RBAC)** to ensure secure access to system functions and data. The module supports multiple user roles with specific permissions:

- **Administrator**: Full access to manage cameras, users, alerts, and system settings.
- **Manager**: Access to monitoring data, report generation, and alert responses.
- **Viewer**: Read-only access to real-time monitoring streams and reports.

A secure authentication mechanism verifies user identities, ensuring that only authorized users can access system resources and data.

5.2 Input and Output Format

- **Input**: The system takes in live camera streams with a resolution of **1280x720 pixels**.
- **Output**: The output includes real-time pedestrian flow data generated by the algorithm, which is displayed in a graphical format using **charts that depict changes in pedestrian flow over time**.

5.3 Interface Acceptance Criteria

- **Data Overview Module (Front-End)**: Daily, weekly, and monthly pedestrian flow charts must be displayed accurately, along with operating hours and the current day's flow count.
- **Video Surveillance Module (Front-End)**: Upon clicking the respective button, the system should accurately display real-time video streams from selected cameras.
- **Data Details Module (Front-End)**: Upon interaction, the module should display detailed real-time data including total pedestrian counts, current flow rate (in people per minute), peak flow times, and alerts for any anomalies detected in the flow patterns.
- **Camera Management Module (Front-End)**: The interface must allow users to select a specific camera stream and display the real-time video for the selected camera.
- **User Management Module (Front-End)**: Interface must allow setting user roles and managing access rights securely.

5.4 Functional Acceptance Criteria

- **Pedestrian Detection Function**: The system must use a live video stream of **1280x720** resolution to detect pedestrian flow in real time. The error rate between the detected flow and the calculated count must be within **2 individuals per second** for accuracy.
- **Alert Function**: The system must continuously calculate the **average pedestrian flow**. If the current flow rate exceeds **100%** of the average rate, an alert must be generated.
- **Front-End Functionality**: The front-end must be capable of displaying real-time pedestrian flow accurately. The acceptable error rate must be less than **2 individuals per second** to be considered accurate.