Overview

The purpose of this document is to define the functional requirements and non-functional requirements for the development of the Server Monitoring web application. This document is the reference that helps all stakeholders understand the features of this application. Server Monitoring web application is a valuable tool that provides real-time insights by continuously monitoring CPU usage, memory, disk space, and network, which can help detect performance issues and downtime.

Requirements

1. High-level Requirements:

A web-based application that monitors servers in real-time.
Allows users to add, view, and remove servers.
Provides role-based access control.
Offers a comprehensive dashboard displaying real-time server metrics.

2. Detailed Requirements:

2.1 Functional Requirements:

2.1.1 Add Servers:

The application should provide a user-friendly form to input server details.
The backend should validate server details such as hostname, IP address, and credentials.
The system should store server details in a PostgreSQL database.
The backend should provide API endpoints to process server registration.

2.1.2 Remove Servers:

Users should be able to view a list of all registered servers.
An option/button should be provided to remove servers.
A confirmation dialog should appear when removing servers.

The backend should offer API endpoints to handle server removal requests.
Removed servers should be deleted from the database.

2.1.3 User Interface:

Users should see a dashboard displaying real-time server metrics.
The dashboard should update in real-time to reflect changes in the server list.

2.1.4 Security Considerations:

Implement role-based access control to ensure only authorized users can perform specific actions.
An audit trail of server management actions should be maintained.

2.2 Non-Functional Requirements:

Performance: The application should load within 3 seconds and updates should appear in real-time without noticeable lag.
Security: All data transactions, including server details, should be encrypted.
Usability: The UI should be intuitive and user-friendly, even for non-technical users.
Reliability: The application should have an uptime of at least 99.9%.
Scalability: The system should handle the addition of hundreds of servers without degradation in performance.

3. User Stories:

3.1 System Administrators/IT Operations Teams:

SA/IT1: As a system administrator, I want to add a new server to monitor its performance.
SA/IT2: As a system administrator, I want to view all servers' metrics on a single dashboard for efficient monitoring.
SA/IT3: As an IT operations member, I want to remove servers that are no longer in use.

3.2 Business Owners and Managers:

BO/M1: As a business owner, I want to view high-level server metrics to make informed decisions on resource allocation and budgeting.
BO/M2: As a manager, I want to ensure that all servers are operating within acceptable performance ranges to guarantee client satisfaction.

3.3 Application Developers:

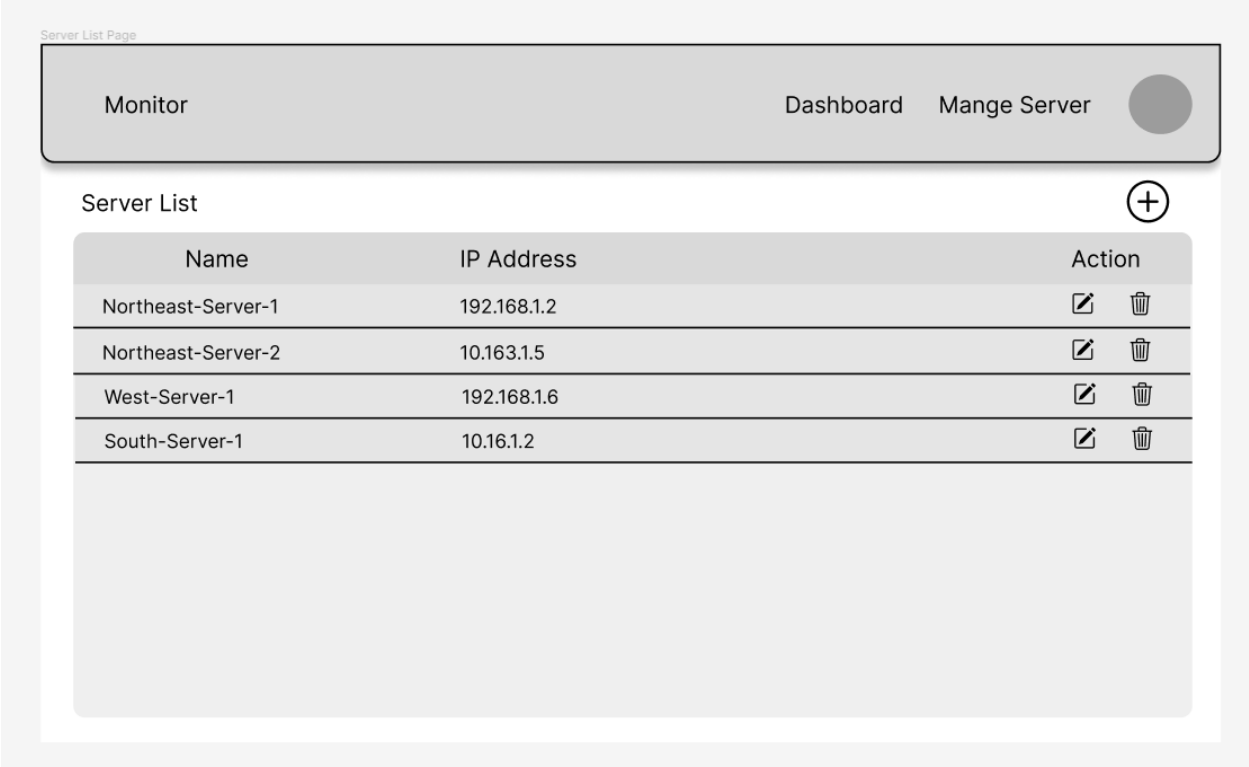
AD1: As an application developer, I want to check my application's performance metrics to optimize its functionality.
AD2: As an application developer, I want to identify any server-related bottlenecks that could be impacting application performance.

3.4 Security Teams:

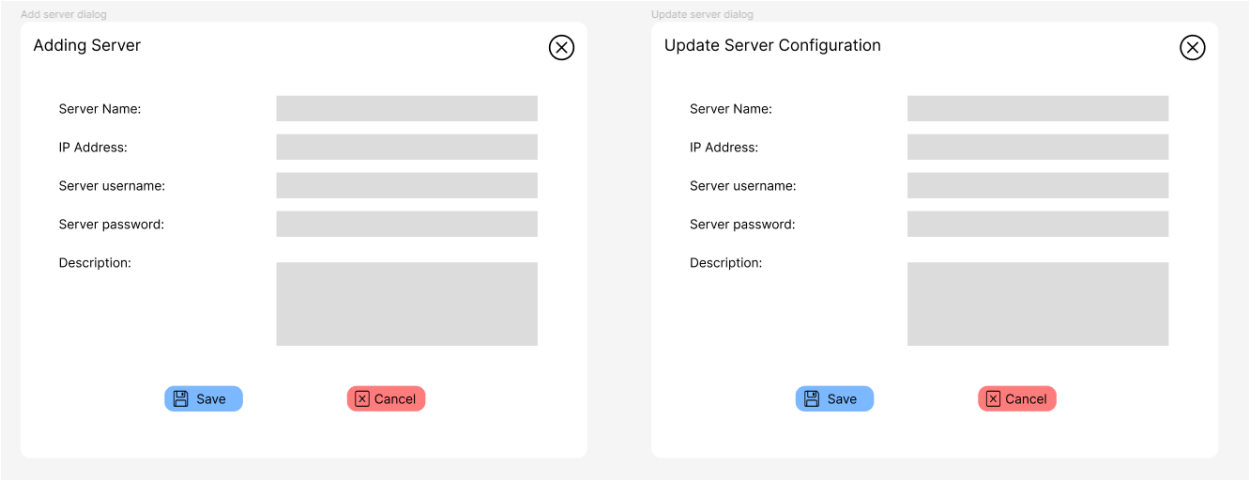
ST1: As a security team member, I want to track server activity logs to identify and respond to potential security threats.
ST2: As a security team member, I want to review the audit trails of server additions and removals to ensure only authorized actions were taken.

UI Design Interface

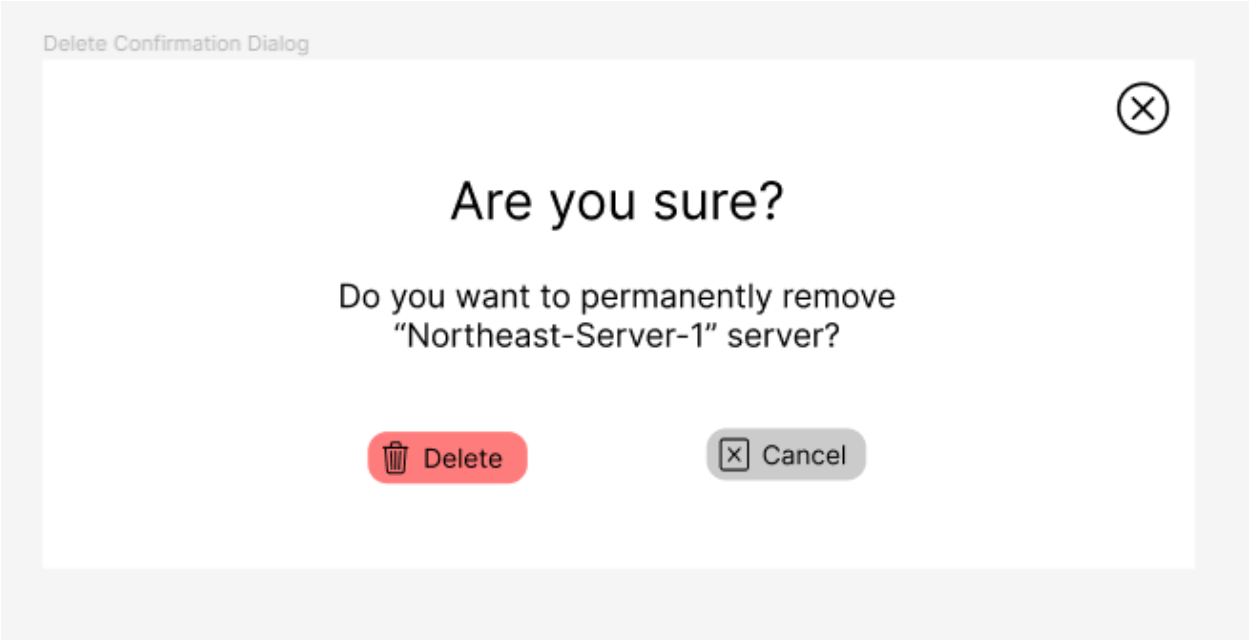
1. Server List Page



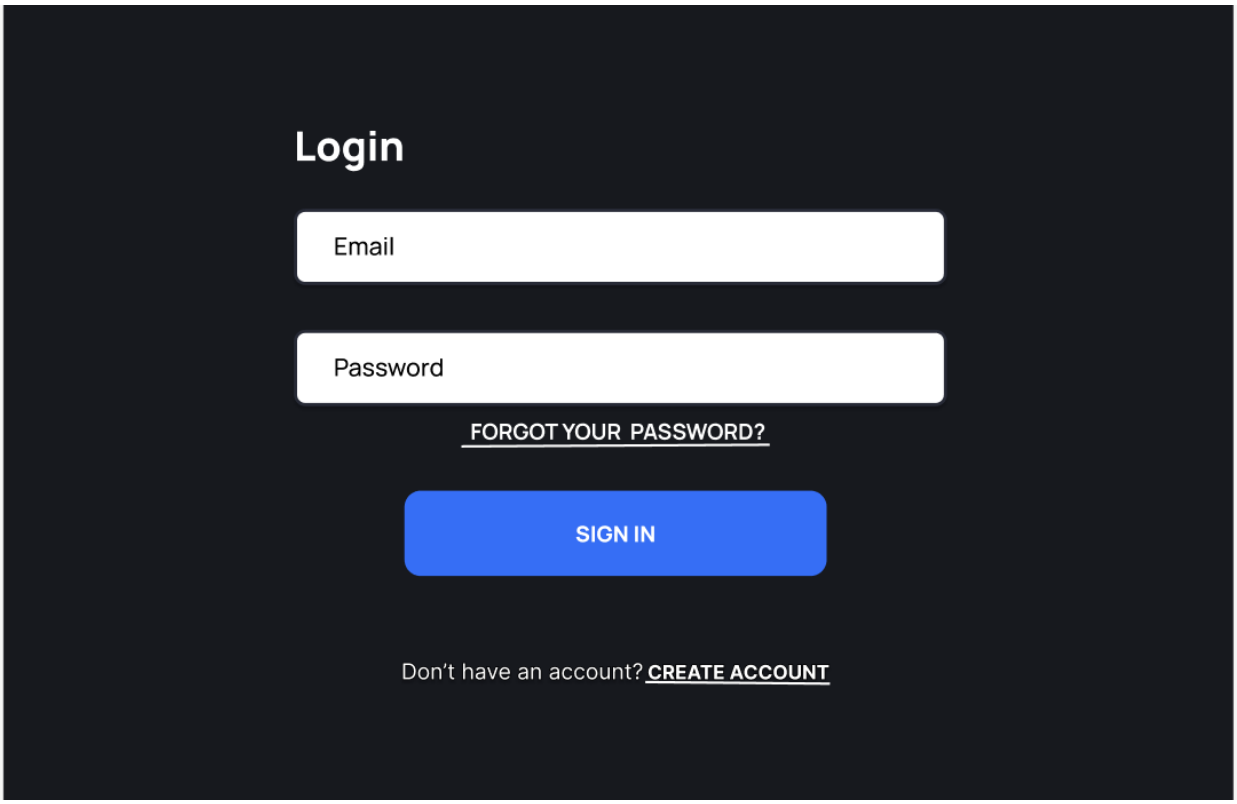
2. Add and update server configuration pages



3. Delete Confirmation page



4. Login Page



5. Create Account Page