Software Requirements Specification

for

Enhancement on pgAgent Features

(In collaboration with IITM Pravartak)

Version 1.0

Prepared by
Brian Christopher
Joel Daniel Pradeep

TKM COLLEGE OF ENGINEERING

30 January 2025

Table of Contents

1. Introduction

1.1 Purpose

*This document outlines the functional and non-functional requirements for enhancing PgAgent with **Job Dependency and Audit Logging** features. The primary objectives are:*

- ***Job Dependency:** Enable the scheduling of jobs based on the completion status of other jobs.*
- ***Audit Logging:** Implement a logging mechanism to track job-related actions such as creation, modification, deletion, and execution.*

*These enhancements aim to improve job execution workflows, provide **transparent auditing for administrative and security purposes**, and optimize job scheduling efficiency in PostgreSQL environments.*

1.2 Project Scope

The system extends PgAgent functionalities by implementing:

- ***Job Dependency:** Supporting sequential, conditional, and parallel job dependencies.*
- ***Audit Logging:** Logging all job-related actions with timestamps, user details, and operation types to enable tracking and compliance monitoring.*

*This enhancement will facilitate **efficient database task automation, improve system observability, and ensure accountability in job execution workflows**.*

.

1.3 References

1. ***PostgreSQL Official Documentation***

   - *Comprehensive resource for understanding PostgreSQL's features, including job management and logging mechanisms.*
   - *Covers details on extensions like PgAgent and best practices for database auditing.*

2. ***PostgreSQL Logging & Auditing Best Practices***

   - *Provides guidelines on structured logging, error handling, and compliance-driven audit logging in PostgreSQL.*
   - *Discusses use cases for logging job-related operations and security events.*

3. ***Apache Airflow Documentation***

   - *Airflow is a workflow orchestration platform offering robust support for defining job dependencies using Directed Acyclic Graphs (DAGs).*

- *A relevant reference for implementing job dependency management effectively.*

4. SQL Server Agent Documentation

- *SQL Server Agent supports job scheduling and dependency management via configurable steps and triggers.*
- *Includes logging mechanisms that provide insights into job execution history and failures.*

*5. **PostgreSQL pgAgent Documentation***

- *Documentation for pgAgent, the job scheduling agent for PostgreSQL.*
- *Details existing job management features and provides a foundation for enhancements.*

.

2. System Overview

2.1 System Perspective

*The enhancement integrates with existing PgAgent functionalities, extending its capability to support complex job dependencies and **audit logging for job-related activities**.*

2.2 Core Functionalities

- ***Job Dependency:***
  - *Define dependencies between jobs.*
  - *Set conditions such as "execute only if the parent job succeeds."*
  - *Support parallel execution of dependent jobs.*
- ***Audit Logging:***
  - *Log job-related actions such as creation, modification, deletion, and execution.*
  -
  - *Store timestamps, user details, and operation types for tracking changes.*
  -
  - *Provide query mechanisms to filter logs based on job name, operation type, and timestamp..*

2.3User Classes and Characteristics

- ***Database Administrators:** Define job dependencies and oversee audit logs for accountability.*
- ***Developers:** Implement job workflows based on dependency conditions and review logs for debugging.*
- ***Operators:** Monitor job execution, track modifications, and manage job failures using audit logs.*

2.4 Design and Implementation Constraints

- *Must adhere to **PostgreSQL security and logging guidelines**.*
- *Ensure minimal performance impact due to dependency checking and logging mechanisms.*
- *Provide backward compatibility with existing PgAgent deployments.*
- *Implement **secure and tamper-proof audit logging** to prevent unauthorized modifications.*

3.External Interface Requirements

3.1 Hardware Interfaces

- *Supports multi-core CPU environments for parallel job execution and logging processes.*
- *Requires sufficient memory for handling multiple dependent jobs and storing audit logs efficiently.*

3.2 Software Interfaces

- *The system will interact with the following components:*
- *PostgreSQL 14+ for job execution and storing audit logs.*
- *pgAdmin 4 for job monitoring and viewing audit logs.*
- *pgAgent for job scheduling.*
- *Logging and Monitoring Systems for storing and querying audit logs securely.*

3.3 Communication Interfaces

Secure Network Protocols: *TLS 1.2+ for encrypted communication.*

REST APIs: *For external system integration with job dependency management and audit log retrieval***.**

Alerts and Notifications: *Email and webhook notifications for job execution failures and detected anomalies in audit logs.*

4.Functional Requirements

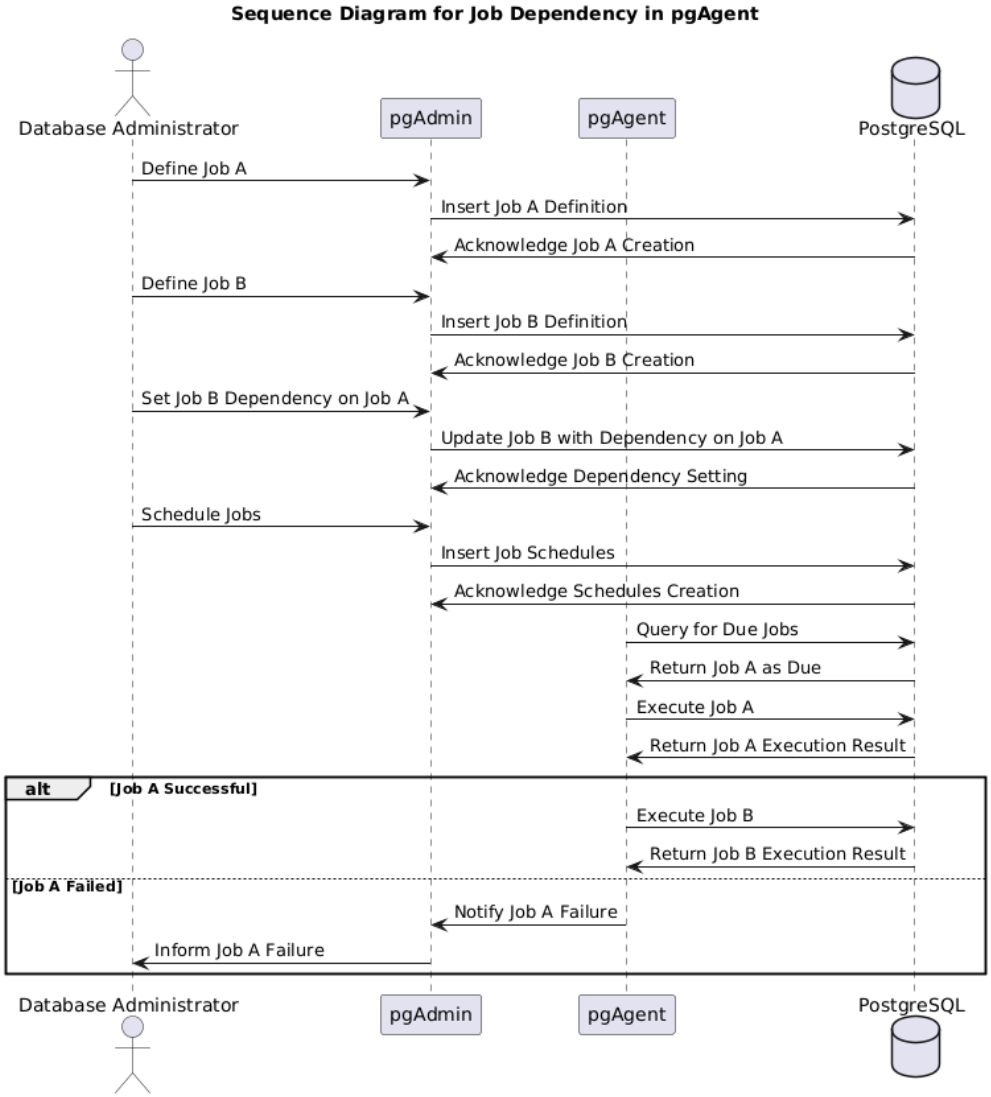
4.1Job Dependency

Description:
Allows users to configure job execution based on dependencies.

Priority: *High*

Functional Requirements:

- *FR-1: Support sequential and conditional job execution.*
- *FR-2: Allow parallel execution of dependent jobs.*
- *FR-3: Provide visual representation of job dependencies in pgAdmin*



4.2 Audit Logging

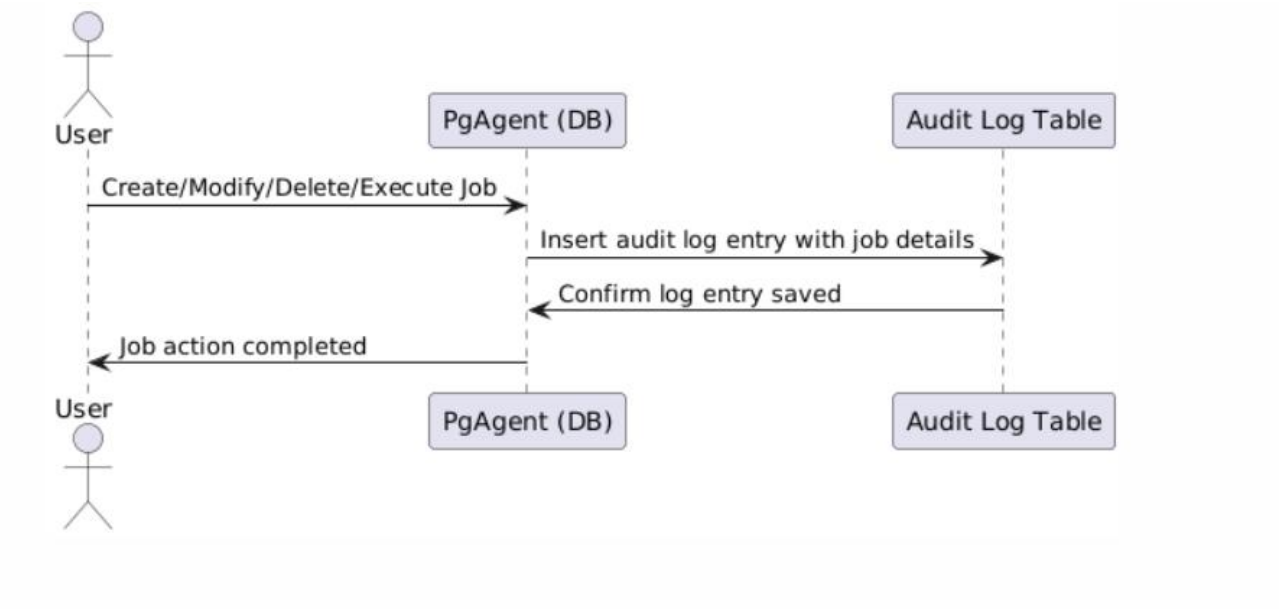
Description:
Implements audit logging for tracking job-related actions and ensuring accountability.

Priority: *Critical*

Functional Requirements:

- *FR-4: Log job-related actions, including creation, modification, deletion, and execution.*
- *FR-5: Ensure secure storage of audit logs with access restricted to authorized users.*
- *FR-6: Provide filtering and search capabilities for querying logs based on job name, operation type, and timestamp*

Activity Diagram



5.Nonfunctional Requirements

5.1 ***Performance***

- *Ensure minimal latency in checking job dependencies and* ***recording audit logs****.*
- *Optimize queries and indexing for efficient* ***log retrieval and job scheduling****.*
- *Benchmark system performance under high job execution loads.*

5.2 ***Scalability***

- *Support an increasing number of job dependencies without significant performance degradation.*
- *Ensure audit logging scales efficiently for enterprise-level deployments.*

5.3 ***Security***

- *Use AES-256 encryption for stored job details* ***and audit logs****.*
- ***Restrict log access*** *to authorized users to prevent unauthorized modifications.*
- *Implement* ***detailed auditing*** *of user activities, including job creation, modification, deletion, and execution.*

5.4 ***Availability***

Ensure a minimum uptime of ***99.9%*** *for job execution, monitoring, and* ***audit logging services****.*

6 System Architecture and Design

6.1 Architectural Overview

The system adopts a modular architecture that integrates seamlessly with PostgreSQL and pgAgent. The architectural components include:

- ***Job Dependency Manager**: Handles dependency rules and execution order.*
- ***Audit Logging Module**: Records job-related actions, including creation, modification, deletion, and execution.*
- ***Database Layer**: Stores job details, dependency information, and **audit logs**.*
- ***Interface Layer**: Provides APIs and pgAdmin UI integration for **audit log visualization and job dependency management**.*

6.2 System Flow

1. *Job Dependency:*
    - *Define job dependencies in pgAdmin.*
    - *Store dependency rules in the database.*
    - *Execute dependent jobs sequentially, conditionally, or in parallel.*
2. *Audit Logging:*
    - *Record user actions related to job creation, modification, deletion, and execution.*
    - *Store logs securely with timestamps and user details.*
    - *Provide filtering options to retrieve specific logs based on job name, operation type, or time period..*

7 Implementation Plan

7.1Phases of Development

The project will be developed in the following phases:

Phase 1: Requirements Gathering and Analysis *(Duration: 2 weeks)*

Phase 2: Job-Dependency *(Duration: 3 weeks)*

Phase 3: Audit Logging *(Duration: 2 weeks)*

Phase 4: System Testing and Optimization *(Duration: 2 weeks)*

7.2 Implementation Steps

- **Step 1: Requirement Analysis**

Analyze the scheduling and access control requirements by collaborating with stakeholders. Identify use cases for job dependencies and audit logging.

- **Step 2: System Design**

Design the system architecture, database schema for job dependencies and roles, and workflows for job execution and audit tracking.

- **Step 3: Job Dependency Development**

Implement functionalities to define job dependencies, set conditions, and allow parallel or sequential execution of jobs.

- **Step 4: Audit Logging Development**

Implement the audit logging module, ensuring all job-related actions are recorded.Design log storage and retrieval mechanisms for efficient access.

- **Step 5: Integration**

Integrate audit logging with pgAdmin for log visualization and job tracking.Ensure compatibility with external monitoring systems like Prometheus and Grafana.

- **Step 6: Testing**

Perform functional testing for job dependency workflows and audit logging correctness.Conduct performance testing under high job loads and security testing for log integrity.

- **Step 7: Deployment**

Deploy the solution in a production-like environment, conduct live testing, and ensure the system meets all functional and nonfunctional requirements.

---

8 Test Plan

8.1Testing Objectives

- ***Verify job dependency execution workflows.***
- ***Validate audit logging for tracking job-related actions.***
- ***Ensure system stability and performance under high job loads.***

8.2Testing Strategy

- ***Functional Testing: Ensure job dependencies (sequential, conditional, and parallel) and audit logging features work as intended..***
- ***Performance Testing: Test the system's efficiency under heavy job scheduling and log generation.***
- ***Security Testing: Validate access controls for log viewing and ensure log integrity.***
- ***Usability Testing: Ensure the pgAdmin UI for audit log and job dependency visualization is user-friendly.***

8.3Test Cases

- ***TC-1: Verify that jobs are executed in the correct order based on defined dependencies.***
- ***TC-2: Check if a dependent job executes only when the parent job succeeds.***
- ***TC-3: Validate parallel execution of dependent jobs without conflicts.***
- ***TC-4: Verify that audit logs capture job creation, modification, deletion, and execution actions.***
- ***TC-5: Test filtering and searching functionality for audit logs.***
- ***TC-6: Check the system's response when a parent job in a dependency chain fails.***
- ***TC-7: Test system behavior under a high number of concurrent job executions.***
- ***TC-8: Validate log encryption and access control mechanisms.***
- ***TC-9: Test proper notifications for job failures or access violations.***
- ***TC-10: Check the encryption of job details in the database and secure communication channels.***

9 Conclusion and Future Scope

9.1Summary

*The proposed enhancements to pgAgent, specifically the implementation of **Job Dependency and Audit Logging**, aim to significantly improve the usability, efficiency, and security of PostgreSQL job scheduling. By introducing advanced dependency management, users can schedule jobs based on sequential, conditional, or parallel execution requirements, optimizing workflows for complex database tasks. **Audit Logging ensures that all job-related actions are recorded, providing transparency, accountability, and compliance tracking.** These features, combined with seamless integration into existing pgAgent systems, provide a robust solution for database administrators, developers, and operators to manage and monitor jobs effectively.*

9.2Future Enhancements

*While the current implementation addresses the core needs of **job monitoring and auditing**, several areas for future development could further enhance the system's functionality:*

1. ***Enhanced Visualization**: Incorporate advanced graphical representations for job dependencies and audit logs, such as **timeline-based log views**.*
2. ***Granular Log Retention Policies**: Implement **configurable log retention settings** to allow administrators to control storage usage.*
3. ***Real-Time Alerts**: Introduce **event-driven log monitoring** to trigger alerts for critical job failures or unauthorized modifications.*
4. ***Improved Scalability**: Optimize the system further to handle **millions of logs efficiently** in enterprise environments.*