# Software Design Document (SDD)

for

<Enhancement on pgAgent Features: Job Dependency & Role-Based Access Control (RBAC)>

## Version 1.0

**Date:** 06-03-2025
**Prepared by:** Brian Christopher, Joel Daniel Pradeep
**Institution:** TKM College of Engineering (In collaboration with IITM Pravartak)

# Table of Contents

# 1. Introduction

## 1.1 Purpose

This document describes the software design for enhancing **pgAgent** with **Job Dependency Management** and **Role-Based Access Control (RBAC)**. These enhancements will enable **efficient job scheduling** and **secure access management** within **PostgreSQL** environments.

## 1.2 Scope

The system will provide:

- **Job Dependency Management**: Sequential, conditional, and parallel execution of jobs.
- **RBAC Implementation**: Role-based permissions (Admin, Developer, Viewer) for secure job execution.

These enhancements ensure **better automation, improved security, and greater scalability** for PostgreSQL job management.

## 1.3 Definitions & Acronyms

- **pgAgent** – Job scheduler for PostgreSQL.
- **RBAC** – Role-Based Access Control.
- **Admin** – User with full control over job scheduling and user roles.
- **Developer** – User with limited control over job creation and execution.
- **Viewer** – User with read-only access to job status.

## 1.4 References


1. PostgreSQL Documentation.
2. pgAdmin and pgAgent Documentation.
3. Research Paper: "Efficient Task Scheduling in Database Systems".
4. ISO/IEC/IEEE 26514:2022 Software Documentation Standards.


---

# 2. System Overview

## 2.1 High-Level Architecture

The architecture consists of:

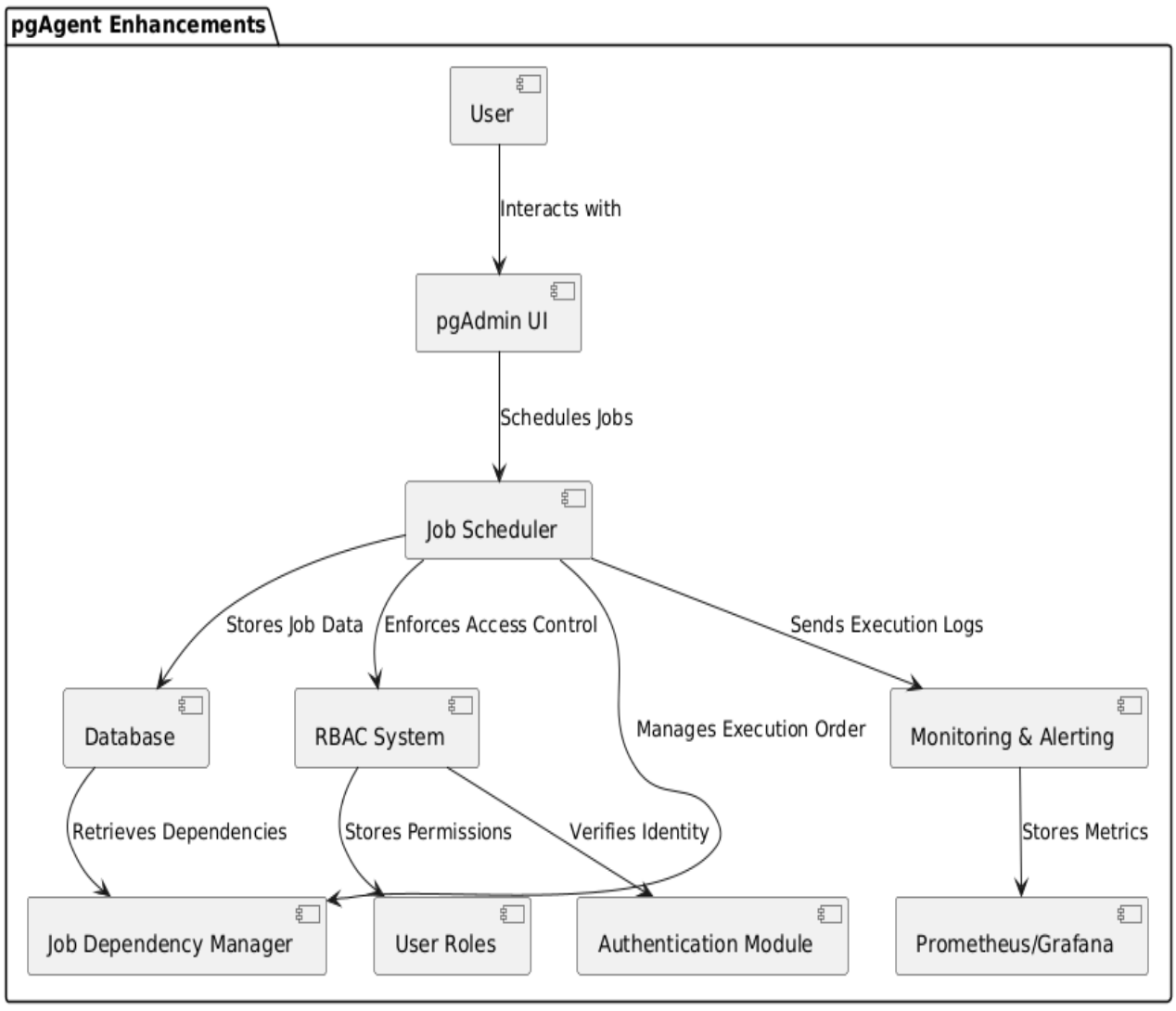
- **User Interface (UI)**: pgAdmin dashboard for job dependency visualization and RBAC management.
- **Job Execution Engine**: pgAgent handling job scheduling based on dependency configurations.
- **RBAC Module**: Manages user authentication and access control.

- **Monitoring & Alerting**: Integration with Prometheus/Grafana for job execution monitoring.

## 2.2 System Workflow

1. User schedules a job via pgAdmin.
2. pgAgent verifies job dependencies before execution.
3. RBAC enforces user roles, restricting unauthorized actions.
4. Jobs execute sequentially, conditionally, or in parallel based on dependency rules.
5. Monitoring tools track execution and alert administrators in case of failures

## 2.3 Architecture Diagram



# 3. System Components

## 3.1 Functional Components

| Component | Technology | Description |
|---|---|---|
| **Job Dependency Manager** | PostgreSQL, pgAgent | Defines job dependencies and execution order |
| **RBAC System** | PostgreSQL Roles, JWT | Implements access control mechanisms |
| **User Interface** | pgAdmin | Visualizes job dependencies & role management |
| **Monitoring Module** | Prometheus, Grafana | Tracks job execution and sends alerts |

## 3.2 Subsystems & Modules

- **Job Dependency Management**
  - Sequential, conditional, and parallel job execution.
  - Job failure handling with retry mechanisms.
- **RBAC Module**
  - Role-based permissions for job creation, execution, and deletion.
- **Monitoring & Alerting**
  - Real-time job execution monitoring and failure alerts.

---

# 4. Data Design

## 4.1 Data Structures

### Job Dependency Schema (PostgreSQL Table Format)

```
CREATE TABLE job_dependencies (
    job_id SERIAL PRIMARY KEY,
    parent_job_id INT REFERENCES jobs(job_id),
    dependency_type VARCHAR(50), -- 'sequential', 'conditional', 'parallel'
    status VARCHAR(50) -- 'pending', 'running', 'completed', 'failed'
);
```

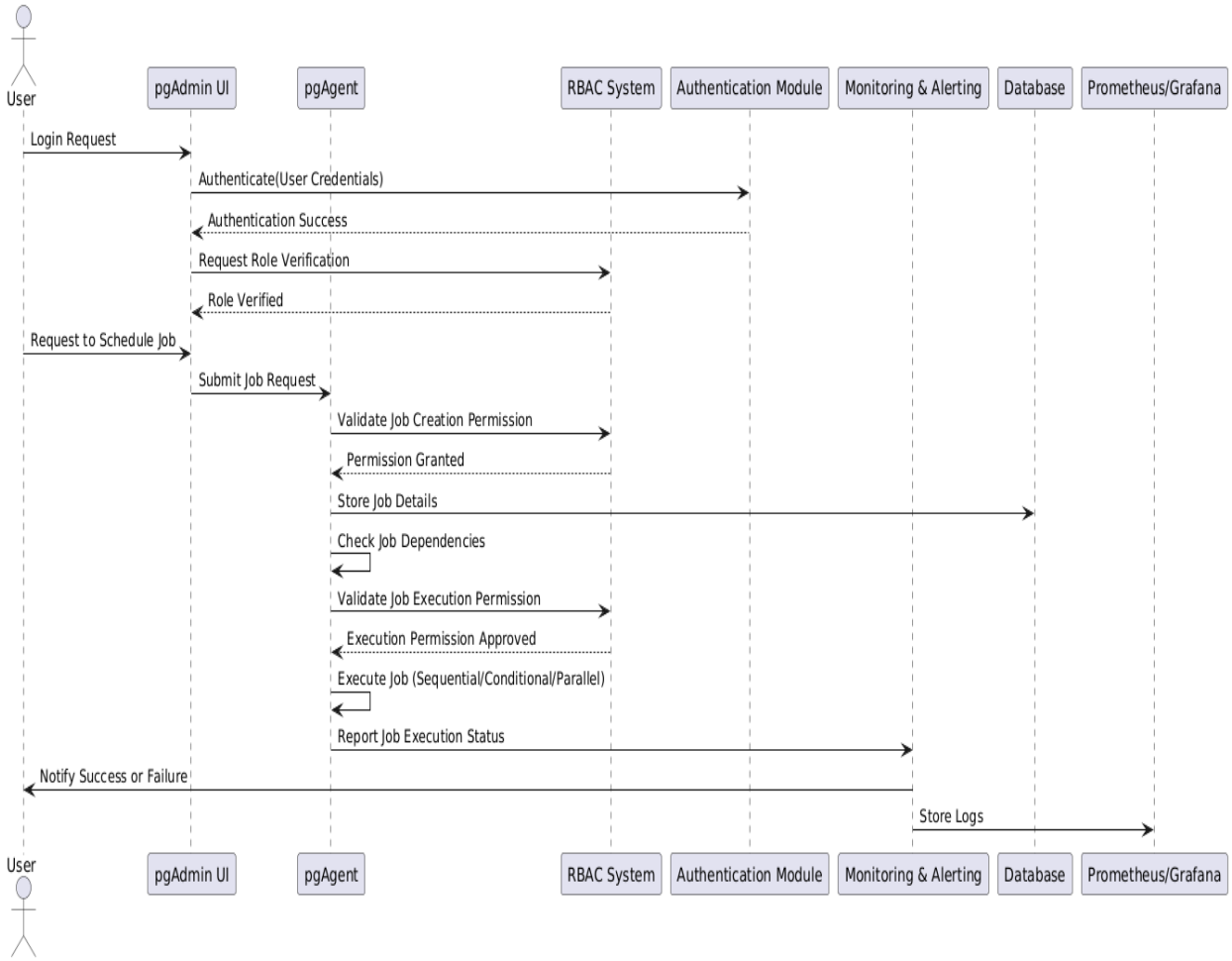
### RBAC Role Schema (PostgreSQL Table Format)

```
CREATE TABLE user_roles (
    user_id SERIAL PRIMARY KEY,
    role_name VARCHAR(50), -- 'Admin', 'Developer', 'Viewer'
    permissions TEXT
);
```

---

# 5. Workflow & Interactions

## 5.1 Sequence Diagram




## 5.2 Job Execution Flow

1. **User schedules a job** → pgAgent registers the job.
2. **Job Dependency Check** → pgAgent validates dependency conditions.
3. **Job Execution** → Jobs execute sequentially, conditionally, or in parallel.
4. **RBAC Enforcement** → Access to job creation and execution is verified.
5. **Monitoring & Alerts** → Job execution is tracked, and failures trigger alerts.

---

# 6. Security Considerations

## 6.1 Data Privacy

- Use **TLS 1.2+** for encrypted communication.
- Encrypt sensitive job execution logs using **AES-256**.

## 6.2 Access Control

- **Strict role-based access enforcement** to prevent unauthorized actions.
- **JWT-based authentication** for secure API access.

## 6.3 Audit Logging

- Maintain **detailed logs of job execution and user activities** for security audits.

---

# 7. Testing Strategy

## 7.1 Unit Testing

| Test Type | Tools Used |
|---|---|
| Job Dependency Execution | PyTest, pgTAP |
| RBAC Access Control | JUnit, pgAdmin Tests |
| API Security | OWASP ZAP |

## 7.2 Integration Testing

| Component | Tool Used |
|---|---|
| End-to-End Job Execution | Docker, Kubernetes |
| RBAC Policy Enforcement | Selenium Tests |

## 7.3 Performance Testing

- **Latency**: Ensure job scheduling completes within **<1s** delay.
- **Throughput**: Support **100+ concurrent job executions**.

---

# 8. Deployment Strategy

## 8.1 Deployment Plan

| Component | Deployment Environment |
|---|---|
| Job Scheduler | PostgreSQL (pgAgent) |
| RBAC System | PostgreSQL Roles & API |
| Monitoring | Prometheus & Grafana |

## 8.2 Version Control & CI/CD

- **GitHub/GitLab** for version control.
- **Docker + Kubernetes** for scalable deployment.
- **Jenkins/GitHub Actions** for CI/CD automation.

---

# 9. Future Enhancements

1. **AI-Driven Job Execution Optimization** – Predict job failures using ML.
2. **Cloud Integration** – Extend pgAgent support to **AWS RDS & Google Cloud SQL**.
3. **Automated Job Recovery** – Implement **self-healing mechanisms** for failed jobs.

---

# 10. Compliance & Standards

- **ISO/IEC/IEEE 26514:2022** – Software Documentation Standards.
- **OWASP Security Guidelines** – Ensuring API & Role-based security best practices.

---

# 11. Conclusion

This **Software Design Document (SDD)** outlines a scalable and secure solution for **enhancing pgAgent** with **Job Dependency Management and Role-Based Access Control (RBAC)**. By integrating **robust scheduling, access control, and monitoring mechanisms**, this project significantly **improves PostgreSQL job management**, ensuring **efficiency, security, and scalability**.

---