Bhanoo Pratap Singh

JobSphere Architecture Document

1. Overview

JobSphere is a microservices-based job portal that allows **Job Seekers** to find and apply for jobs, **Employers** to post and manage job listings, and **Admins** to monitor platform activities. The system is built using **Java 17, Spring Boot 3.x, AngularJS 1.8, and AWS services**.

---

2. Tech Stack

Component	**Technology Used**
Backend	Java 17, Spring Boot 3.x, Spring Cloud (Eureka, Gateway, Resilience4J)
Frontend	AngularJS 1.8, Thymeleaf, Bootstrap 5
Database	PostgreSQL (AWS RDS)
Storage	AWS S3 (Resume Storage)
Authentication	Spring Security, JWT, AWS Cognito
Notifications	AWS SES (Emails)
Deployment	Docker, Kubernetes (AWS EKS)
Monitoring & Logging	AWS CloudWatch, AWS X-Ray
CI/CD	GitHub Actions, AWS CodeDeploy
Caching	Redis or AWS ElastiCache

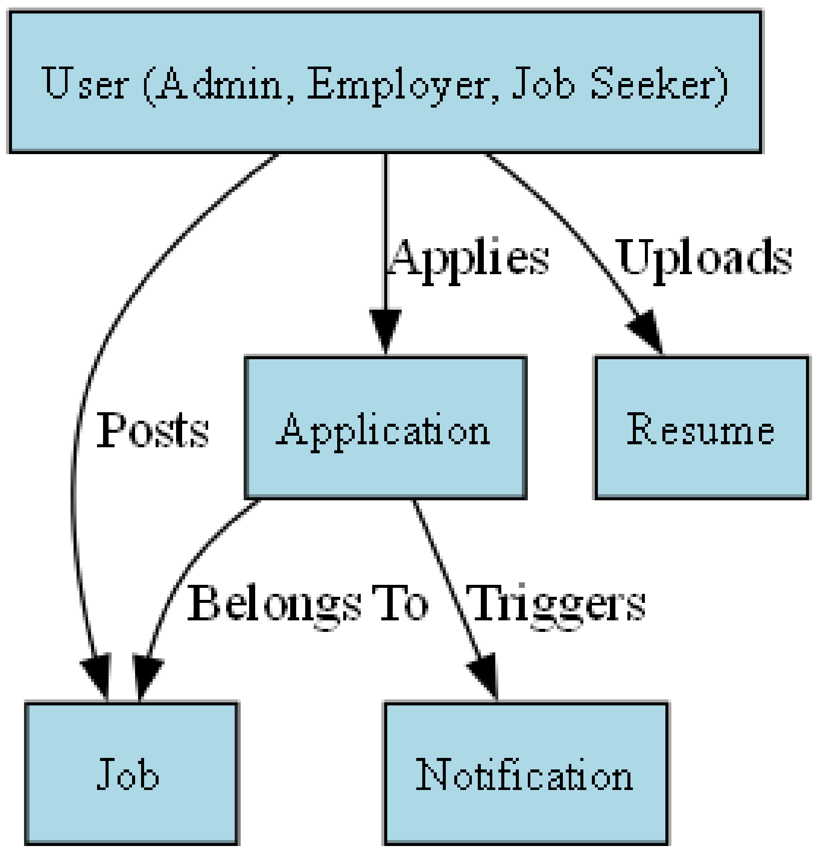
---

3. ER Diagram (Entity-Relationship Diagram)

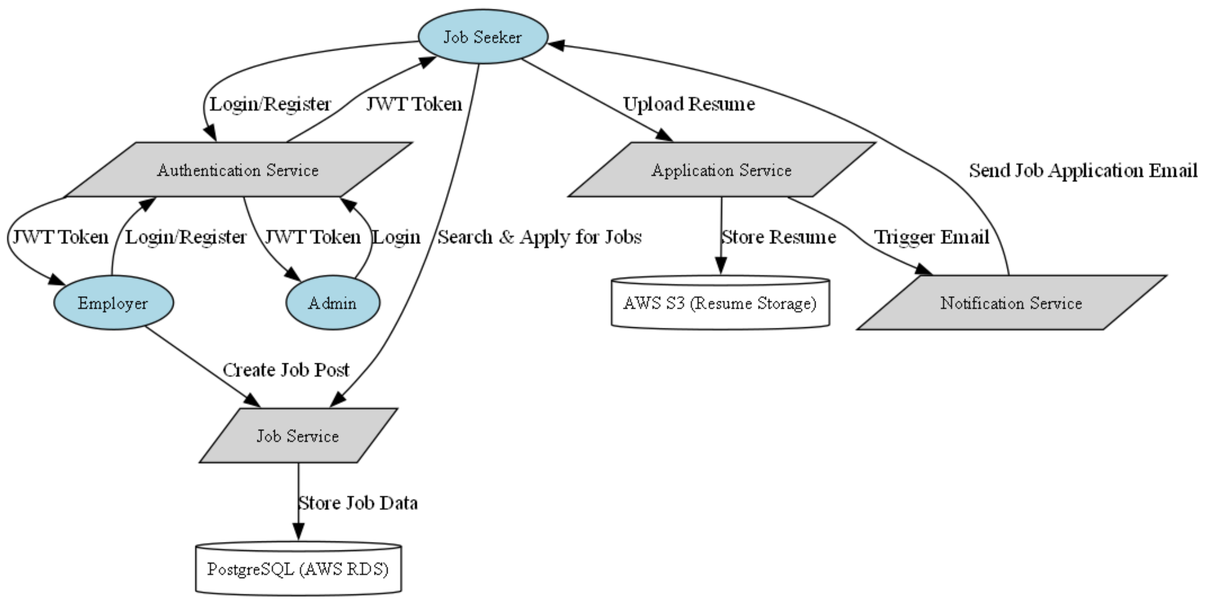
The **ER Diagram** represents how entities interact in JobSphere. The main entities are:

1. **User** (Admin, Employer, Job Seeker)
2. **Job** (Posted by Employers)
3. **Application** (Job Seeker applies for a Job)
4. **Resume** (Uploaded to AWS S3)
5. **Notification** (Email alerts for applications)

ER Diagram



Data Flow Diagram (DFD)



4. Explanation of Architecture Components

1. Authentication Service

- Handles **user registration, login, and role-based authentication** (Admin, Employer, Job Seeker).
- Uses **Spring Security 6** and **JWT** for authentication.
- Can integrate with **AWS Cognito** for user management.
- Ensures secure authentication and token validation.

2. Job Service

- Allows **employers** to create, update, delete, and manage job postings.
- Provides **job search and filtering** for job seekers.
- Stores job details in **PostgreSQL (AWS RDS)**.
- Exposes REST APIs for frontend and other services.

3. Application Service

- Allows job seekers to **apply for jobs**.
- Handles **resume uploads** and stores them in **AWS S3**.
- Links job applications with **Job and User entities**.
- Communicates with the **Notification Service** to send email confirmations.

4. Notification Service

- Sends **email notifications** when a job application is submitted.
- Uses **AWS SES (Simple Email Service)** for reliable email delivery.
- Can be extended for **push notifications or SMS** in the future.

5. API Gateway & Service Discovery

- Uses **Spring Cloud Gateway** to route requests to the correct microservice.
- Implements **Eureka Service Discovery** for dynamic service registration.
- Provides **load balancing** and **centralized authentication**.

6. Frontend (AngularJS + Thymeleaf + Bootstrap 5)

- **Landing Page**: Displays job listings and search functionality.
- **Employer Dashboard**: Employers manage job posts and view applicants.
- **Job Seeker Dashboard**: Users apply for jobs and upload resumes.
- **Admin Panel**: Allows administrators to monitor user activities.

7. AWS Deployment & Infrastructure

- **Backend Services**: Deployed on **AWS EC2 or AWS Elastic Beanstalk**.
- **Database**: Hosted on **AWS RDS (PostgreSQL)**.
- **File Storage**: **AWS S3** for storing resumes securely.
- **Frontend Hosting**: AngularJS deployed on **AWS S3 + CloudFront**.
- **DNS & Routing**: Managed via **AWS Route 53**.
- **Logging & Monitoring**: AWS **CloudWatch & X-Ray** for tracking system health.

8. Performance & Scaling

- **Containerization**: Uses **Docker** for packaging microservices.
- **Orchestration**: Deployed on **Kubernetes (AWS EKS)** for scalability.
- **Circuit Breaker**: Uses **Spring Cloud Resilience4J** to prevent failures.
- **Caching**: Implements **Redis (AWS ElastiCache)** for faster API responses.