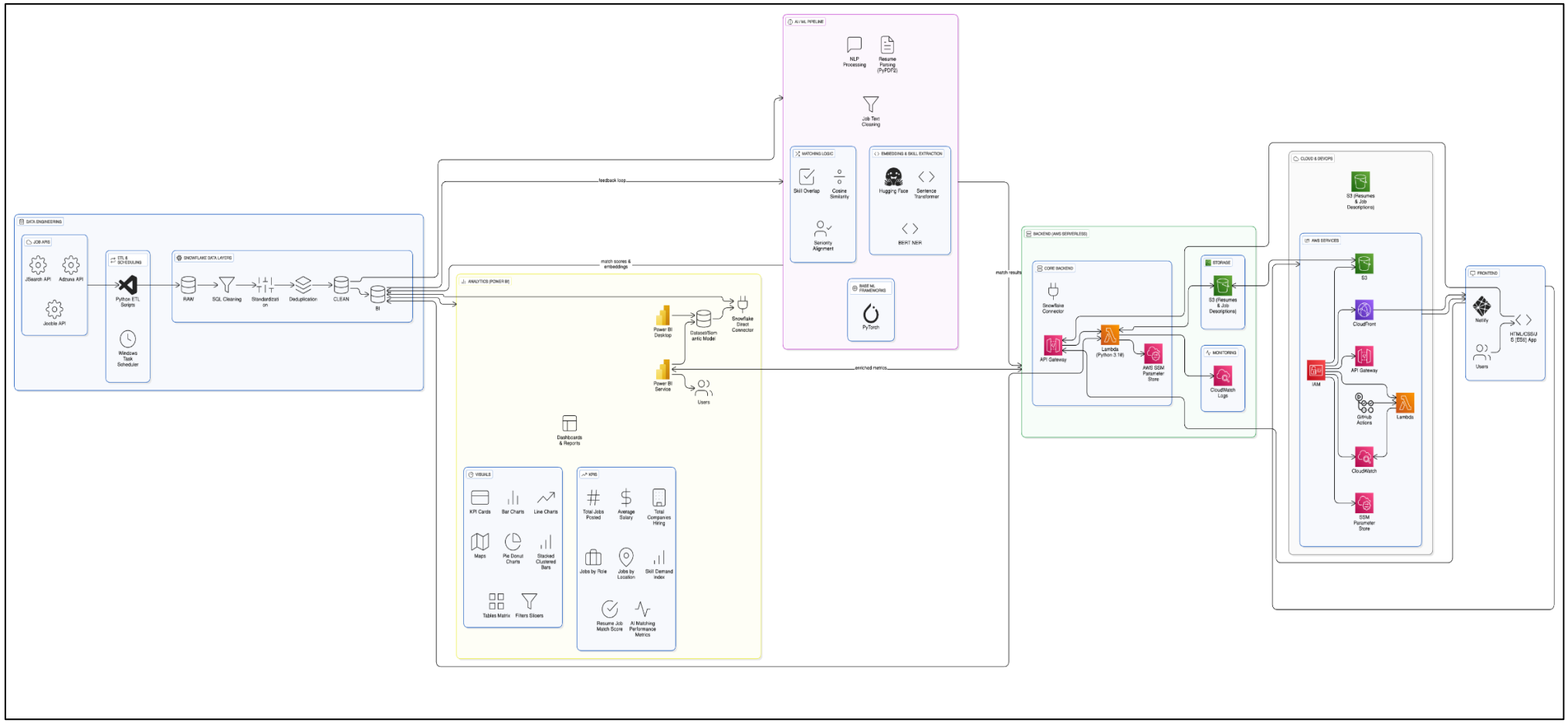
||Jai Sri Gurudev ||
BGSKH Education Trust (R.) – A unit of Sri Adichunchanagiri Shikshana Trust(R.)
BGS College of Engineering and Technology
Mahalakshmipuram, West of Chord Road, Bengaluru-560086
(Approved by AICTE, New Delhi and Affiliated to VTU, Belagavi)

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

Project Title: Remote Staffing System

System Architecture

USN	NAME
1MP22AD012	DARSHAN N A
1MP22AD058	TANISHA K S
1MP22AD001	ABHISHEK H G
1MP22AD051	SHREYA P
1MP22AD055	SUSHMITHA C
1MP22AD033	MEGHANA B


DATA ENGINEERING
JOB APIS
JSearch API
Adzuna API
Jooble API
ETL & SCHEDULING
Python ETL Scripts
Windows Task Scheduler
SNOWFLAKE DATA LAYERS
RAW
SQL Cleaning
Standardizati on
Deduplication
CLEAN
BI
AI / ML PIPELINE
NLP Processing
Resume Parsing (PyPDF2)
Job Text Cleaning
MATCHING LOGIC
Skill Overlap
Cosine Similarity
Seniority Alignment
EMBEDDING & SKILL EXTRACTION
Hugging Face
Sentence Transformer
BERT NER
BASE ML FRAMEWORKS
PyTorch
feedback loop
match scores & embeddings
ANALYTICS (POWER BI)
Power BI Desktop
Dataset/Sem antic Model
Snowflake Direct Connector
Power BI Service
Users
Dashboards & Reports
VISUALS
KPI Cards
Bar Charts
Line Charts
Maps
Pie Donut Charts
Stacked Clustered Bars
Tables Matrix
Filters Slicers
KPIS
Total Jobs Posted
Average Salary
Total Companies Hiring
Jobs by Role
Jobs by Location
Skill Demand index
Resume Job Match Score
AI Matching Performance Metrics
enriched metrics
match results
BACKEND (AWS SERVERLESS)
CORE BACKEND
Snowflake Connector
API Gateway
Lambda (Python 3.14)
AWS SSM Parameter Store
STORAGE
S3 (Resumes & Job Descriptions)
MONITORING
CloudWatch Logs
CLOUD & DEVOPS
S3 (Resumes & Job Descriptions)
AWS SERVICES
S3
CloudFront
API Gateway
IAM
GitHub Actions
Lambda
CloudWatch
SSM Parameter Store
FRONTEND
Netlify
HTML/CSS/JS (ES6) App
Users

1. **Role: Data Engineer**

Environments Used:

- Collected job data from JSearch, Adzuna, and Jooble using Python ETL scripts in VS Code
- Stored and organized all data in Snowflake using RAW, CLEAN, and BI layers
- Cleaned, standardized, and removed duplicates using SQL transformations
- Automated the entire data ingestion process using Windows Task Scheduler
- Prepared analytics-ready datasets for AI matching and Power BI dashboards

2. **Role: Power BI Analyst**

Environments Used:

- Connected Power BI to Snowflake to work with cleaned datasets
- Created interactive dashboards showing key KPIs like Total Jobs, Companies, Avg Salary, UK Jobs
- Built visuals for job roles, locations, industries, and hiring companies
- Added filters & slicers for user-friendly exploration (role, location, skills, experience)
- Converted complex data into easy-to-understand insights and trends

3. **Role: AI / ML Engineer**

Environments Used:

- Developed NLP pipelines in VS Code using Python for resume parsing, job-text cleaning, and skill extraction.
- Generated semantic embeddings using Sentence Transformer models (all-mpnet-base-v2) for job–resume similarity.
- Implemented NER-based skill extraction using Hugging Face Transformers (bert-base-NER).
- Processed resume PDFs using PyPDF2 for accurate text extraction and normalization.
- Loaded and processed cleaned job datasets from Snowflake using Snowpark and pandas.
- Designed the job–resume matching logic using cosine similarity, skill overlap, and seniority alignment.
- Tuned and validated AI/ML components to improve ranking accuracy and consistency across datasets.

4. **Role: Frontend Development**

Environment Used:

Programming Languages & Versions: HTML5**,** CSS3**,** JavaScript (ES6+)

Tools / IDEs:
Visual Studio Code (primary editor for UI development)
Chrome DevTools (for debugging, layout testing, and responsiveness checks)

Frontend Frameworks / Libraries:
Vanilla JavaScript (core scripting)
CSS Flexbox & Grid (page structure and layout)
Firebase Web SDK (for authentication, data retrieval, and real-time updates)

Database Used (Frontend Integration):
Firebase Firestore – for storing user data, application details, and dynamic content accessed directly from the frontend.

Runtime Environment:
Local browser runtime (Google Chrome)
VS Code Live Server for real-time preview
Deployment: Netlify
Operating System:
Windows 10/11 (development and testing)

5. <u>Role: Backend Development</u>
Environment Used:

->Programming Languages & Versions
Python 3.10+ (primary language for AWS Lambda functions)
->Backend Frameworks / SDKs
AWS SDK for Python (boto3) – S3, Lambda, API Gateway, SSM, IAM
Snowflake Python Connector – secure database connectivity
Requests / httpx – external API calls (if required)
JSON / Marshmallow schemas – input validation (custom)
->Tools / IDEs
Visual Studio Code (primary development environment)
AWS CLI – for cloud resource access and deployments
Snowflake Web UI – for querying, monitoring, and debugging
Postman / Thunder Client – for API testing
Cloud Services Used (AWS Backend Infrastructure)
AWS Lambda – serverless compute for all backend logic
API Gateway – REST API endpoints (JD Upload, Resume Upload, Match Retrieval)
AWS S3 – storage for uploaded files (PDF resumes, JDs)
AWS SSM Parameter Store – secure secrets management (Snowflake credentials)
CloudWatch – logs, debugging, monitoring
IAM – roles and security policies

->Database / Data Warehouse Integration
Snowflake
Stores all job descriptions, resumes, embeddings, and match results
Queried directly by backend using Snowflake Connector
Used for running matching algorithms and analytics
Runtime Environment
AWS Lambda Runtime (Python 3.10)
Serverless execution with automatic scaling

->Local testing using:
VS Code + AWS Toolkit
Local invocation scripts (SAM CLI optional)
Deployment
AWS Console (direct lambda updates)

AWS CLI (scripted deployments)

6. Role: Cloud / DevOps Development
Environment Used:

➔ **Cloud Platforms & Services:**
AWS and Firebase

➔ **Services Configured:**
AWS S3, CloudFront, API Gateway, Lambda, IAM ,
Firebase Authentication, Firestore, Storage, and Cloud Functions.

➔ **Programming & Configuration:**
JavaScript/Node.js, SQL, YAML/JSON.

➔ **Tools & IDEs Used:**
AWS Console & CLI, Firebase Console & CLI, GitHub Actions.

➔ **CI/CD & Automation:**
Automated deployments using GitHub Actions for frontend to S3
and backend to Firebase Cloud Functions.

➔ **Data Pipeline & Analytics Setup:**
Cloud Functions → Lambda

➔ **Runtime Environment:**
AWS Lambda runtime, Firebase Cloud Functions.

➔ **Monitoring Tools:**
AWS CloudWatch, Google Cloud Monitoring, API logs.

➔ **Deployment:**
AWS hosting with CDN, Firebase functions and services.

Operating System:
Windows 10/11 for development and testing; Linux-based cloud runtimes.