# Job Pulse: Market Insight Analytics

AZURE CAPSTONE PROJECT

- Shiva Kumar Jujare
*Big Data Engineer*

# Project Overview

JPMIA is a platform which enables users to perform following analysis

- Current job trends in the market in different regions and industries
- Top companies hiring, and companies' growth trends
- Comparison of benefits across industries and its impact to attract talent
- Emerging Industries with growing job opportunities
- In-demand and trending skills across sectors

This gives users an outlook about future job opportunities, emerging job trends to watch out for and give recommendations to job seekers based on market insights

# Datasets Overview

- Company Details
  - Companies – This describes about companies in the market
  - Company Industries – This provides industries each company is involved in
  - Company Specialties – This provides each company's specialization
  - Employee Counts – provides employee and followers growth of a company recorded at different times
- Referential
  - Skills – This is a master data where all the skills are recorded and acts as a look up table
  - Industries – This is a master data where all the industries are recorded and acts as a look up table

# Datasets Overview

- Job Details
  - Benefits – List of benefits provided by each job
  - Job Industries – This provides industries each job is related to
  - Job Skills – List of required skills for each job
  - Salaries – Details of each job's salary like max, min and med and compensation type etc

- Postings: This is the list of all jobs, and their details, available in the market and divided into below datasets
  - Full Time Job Postings
  - Contract Job Postings
  - Internship Job Postings
  - Other Job Postings

# Azure Services Used

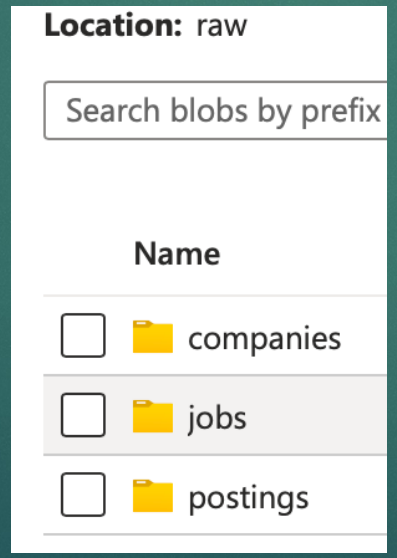
Following Azure services are used to perform data engineering activities on this project datasets

- Storage Account – ADLS Gen 2 storage account to receiver raw files and store processed files and external tables
- Data Factory – To perform data ingestion and execute Databricks notebooks
- Databricks – To clean, transform and create external tables by mounting storage account
- Synapse Analytics – To create external and managed tables and perform the data analysis on the tables
- Service Principal – To control access to services like Storage account
- Key Vault – To store secrets like database passwords, SAS token, connection strings, service principal id and secrets and use them in Data Factory to avoid hardcoding of secrets

# Project Structure

**Storage Account:**

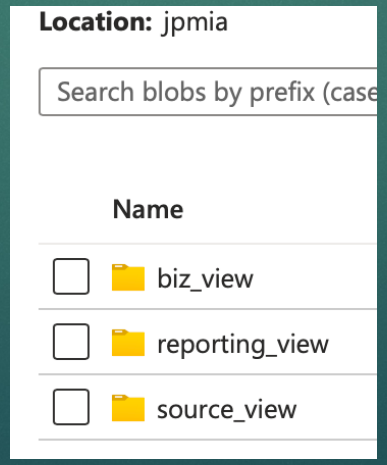
- Raw Container: This is another application folder which has 3 sub folders companies, jobs and postings where we get total 12 files, 4 in each folder. Below is the screenshot of raw container



# Project Structure

**Storage Account:**

- jpmia Container – This folder has 3 sub folders source_view, biz_view and usage view.
  - Source View – contains the files copied from raw container folders
  - Biz View – contains delta tables created by processing source view files. This acts as a serving layer for other applications as well which use few of the delta tables. Here we keep cleaned and formatted data
  - Reporting View – contains delta tables created by processing delta tables from biz view. Here we aggregate the data from biz view tables and create Dimensions and Fact delta tables



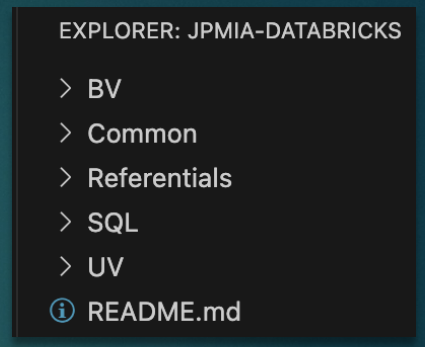
# Project Structure

- Data Factory:
  - Create linked services for Key Vault, Data lake Gen 2, Azure Databricks and Azure Synapse
  - create required datasets and dataflows and pipelines to loop and copy the files dynamically from raw container folders to source view folder in jpmia container.
  - you can create one dataset pointing to raw container and 3 dataflows for each sub folder(companies, jobs and postings) and dynamically provide the file names by looping through each folder
  - create a pipeline to run a notebook which checks if we received all files, if we have all then calls another notebook which loads delta tables in biz view. If we are missing even 1 file, then fail the pipeline
  - create a pipeline to run a notebook which reads the data from biz view delta tables, aggregates and transforms data and merges with reporting view delta tables
  - create a pipeline which calls sql procedure of Synapse dedicated sql pool to refresh synapse tables

For more information you can check the readme and source code in ADF Git Link

# Project Structure

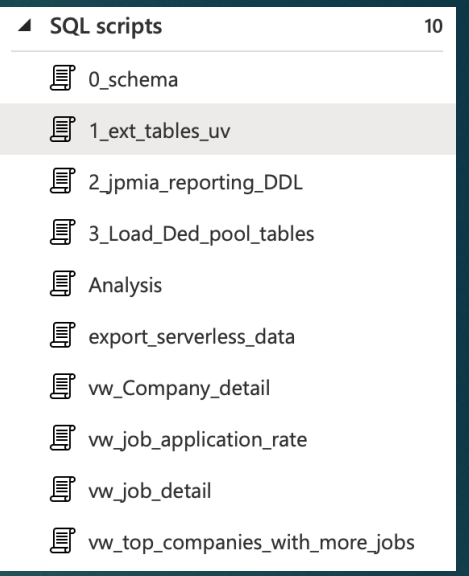
- Azure Databricks:
  - Create a service principal in app registrations and grant it access in storage account and key vault. Store the service principal tenant id, client id and secret value in key vault
  - create databricks backed secret scope using above service principal keys stored in valut. we will use scopes to mount storage account
  - create a workspace with different folders ,as show on the image, to store notebooks for each data layer.
  - Here I have created below folders
    - SQL - To keep SQL DDL scripts used to create tables in BV and UV
    - Common - to store generic notebooks that can be re used in BV and UV refresh process
    - BV and UV(reporting view) - to store notebooks used to refresh respective tables
    - Referentials – to store notebooks used to refresh referential tables in sql database




For more information on the workspace structure please go through README.md file in the datbricks Git Link

# Project Structure

- Azure Synapse:
  - Create a dedicated SQL pool database "jpmia-synapse-analytics"
  - Create external tables pointing to usage view delta tables, by creating
    - master key encryption by password
    - Databse scoped credential with identity = 'SHARED ACCESS SIGNATURE' to access uv folder where delta tables are crated
    - External data source
    - External file format as parquet
  - Create views by joining multiple external tables
  - Create managed tables CompanyDetail, JobApplicationRate, JobDetail, TopCompaniesWithMoreJobs
  - Create stored procedure which truncates and loads the data into these reporting tables




For more information on SQL scripts and tables please go through README.md file in the below repo Snyapse Link

# Rules of Data Cleaning and Transformation for BV

- Companies:
  - company size - keep N/A where it is blank
  - state - keep N/A where it is 0 or blank
  - country - keep N/A where it is 0
  - city - keep N/A where it is 0
  - zip code - keep N/A where it is 0 or blank
  - address - Keep N/A where it is 0, blank or single character
  - add ingest date as today's date
- Company Industry
  - fill blanks with N/A in all columns
  - add ingest date as today's date
- Company specialties
  - fill blanks with N/A in all columns
  - add ingest date as today's date

# Rules of Data Cleaning and Transformation for BV

- Employee counts
  - Add new column date_time_recorded column by converting time recorded in timestamp
  - Add new column date by converting time recorded column into date type
  - add ingest_date as today's date
- Benefits
  - add ingest_date as today's date
- Job Industries
  - add ingest_date as today's date
- Job Skills
  - add ingest_date as today's date
- Salaries
  - Calculate median salary where min and max salary are given but med salary is blank

# Rules of Data Cleaning and Transformation for BV

- Postings
  - Remove columns that are redundant- description, add sal id and remove max, med, min, pay period, work_type, currency, compensation_type
  - Company_id - keep 0 where it Is blank
  - Applies - 0 where it is blank
  - remote allowed - 0 where it is blank
  - views - 0 where it is blank
  - formatted_expereince_level - N/A where it is blank

# Rules of Transformation for UV

- DimCompanies
  - Add a new column formatted_company_size. if company_size inn bv_companies table is between 0 and 2 then keep “small”, if between 3 and 5 then keep “medium”, if above 6 then “large” if blank “N/A”
  - use merge command to update if there is a data change in any column and insert in case of new records
- DimCompanyIndustries
  - Aggregate the industry values of each company in one row delimited with comma
  - use merge command to update if there is a data change in any column and insert in case of new records
- DimCompanySpecialities
  - Aggregate the speciality values of each company in one row delimited with comma
  - use merge command to update if there is a data change in any column and insert in case of new records
- DimJobBenefits
  - Aggregate the benefits values of each job and inferred type in one row delimited with comma
  - use merge command to update if there is a data change in any column and insert in case of new records

# Rules of Transformation for UV

- DimJobIndustries
  - Add a new column industry_name by looking up referential table dbo.Industries, present in sql database referential, based on industry_id
  - use merge command to update if there is a data change in any column and insert in case of new records
- DimJobSkills
  - Add a new column skill_name by looking up referential table dbo.skills, present in sql database referential, based on skill_abbr
  - use merge command to update if there is a data change in any column and insert in case of new records
- FactEmployeeCounts
  - Calculate total_employees_as_of_now and total_followers_as_of_now by taking recent employee_count and follower_count from employee_counts table in BV
  - use merge command to update if there is a data change in any column and insert in case of new records
- Views
  - Create view DimSalaries pointing to job_salaries in BV
  - Create view FactJobPostings pointing to job_postings in BV

# Project Workflow

- Execute pl_srv_main which executes pipelines - pl_srv_companies, pl_srv_jobs and pl_srv_job_postings which will copy all the files from sub folders in raw container to source view folder in jpmia container
- Execute pl_BV_Refresh pipeline which validates the count of files in source_view and executes pl_dbws_main pipeline to refresh delta tables in BV by running databricks notebook
- Execute pl_UV_Refresh pipeline which refreshes delta tables in usage view layer by running databricks notebook
- Execute pl_Load_Report_Tables to run synapse stored procedure to refresh reporting tables

# GitHub Links

Below are the GitHub Links for sample code. Click on the name to go to repo

- Data Factory
- Databricks
- Azure Synapse