Project Roadmap: Interactive Data Visualization for Job Skills Analysis

Team Members

- **Sean Schallberger**
- **Jose Traboulsi**
- **Karla Lopez**
- **Meghdut Noor**

How to Read and Use This Document

This roadmap provides a structured breakdown of the **Sprint-based workflow** for our project. Each **Sprint** consists of multiple tasks that can be worked on **concurrently** by different team members to ensure efficiency. Given the **overlapping nature of work**, it's expected that two people may be modifying the same file at the same time but on different segments to foster this work each member will work on individual files which will be manually merged at the end of each Sprint—this will allows for **collaboration, troubleshooting, and faster completion** of each section.

Collaboration & Communication

- **Ask for help** if stuck — chances are someone else is looking at the same code.
- **Document any issues** encountered so we can resolve them efficiently.
- **EDA (Exploratory Data Analysis) is critical** and is split into multiple parts. Tom, will definitely ask about it so it's essential we **explore the dataset deeply and document our findings clearly.**
- **Use GitHub for version control** and ensure all team members stay updated with the latest changes.

Sprint 1: Data Collection & Cleaning (Preliminary EDA)

🔹 Ticket 1.1: Data Ingestion & Initial Exploration - Competency Analysis

Final File: `eda_analysis.py`

Assignee: [TBD]

Task: Load dataset and evaluate its representation of the job market.

Input: `job_postings.csv`, `job_skills.csv`, `job_summary.csv`

Output: Summary of job distribution, industry coverage, and dataset completeness.

- **Competency Analysis:** Determine if the dataset sufficiently represents the job market by analyzing the distribution of job postings, industry coverage, and diversity of skill data.
- **Job Title Consistency:** Evaluate variations in job titles and potential need for standardization.

🔹 Ticket 1.2: Data Integrity & Cleaning Assessment

Final File: `eda_analysis.py`

Assignee: [TBD]

Task: Assess consistency in job titles, missing values, and duplicates.

Input: `job_postings.csv`, `job_skills.csv`

Output: Identification of inconsistencies, missing values, and duplicate handling strategies.

🔷 Ticket 1.3: Ethical & Bias Evaluation

Final File: `eda_analysis.py`

Assignee: [TBD]

Task: Examine dataset for biases, such as geographical skew or underrepresentation of roles.

Input: `job_postings.csv`, `job_skills.csv`

Output: Documentation of ethical considerations and dataset limitations.

- **Ethical Considerations:** Identify any biases in job postings (e.g., geographical skew, underrepresentation of certain roles) and data sourcing limitations.
- **Limitations Definition:** Document aspects of the dataset that may affect final recommendations, such as incomplete job descriptions, imbalanced job category representation, or outdated postings.

🔷 Ticket 2: Standardizing Job Titles & Cleaning Job Listings

Final File: `data_cleaning.py`

Assignee: [TBD]
Task: Standardize job titles and remove duplicate/irrelevant job postings.
Input: `job_postings.csv`
Output: Cleaned dataset with standardized job titles and grouped/mapped experience level in separate columns.

Process Details:

1. **Standardizing Job Titles:**
   - Normalize variations (e.g., "Sr. Data Analyst" → "Senior Data Analyst").
   - Remove abbreviations and ensure consistent formatting.
   - Example: "ML Engineer" → "Machine Learning Engineer".
2. **Experience Level Mapping:**
   - Define job title experience levels (e.g., "Junior", "Mid", "Senior", "Lead").
   - Example Mapping:
     - "Data Scientist" → "Mid-Level"
     - "Senior Data Scientist" → "Senior"
3. **Removing Duplicate/Irrelevant Job Listings:**
   - Identify and remove exact duplicates.
   - Filter out postings with incomplete or missing critical data fields.
   - Example: If two identical job postings exist with different URLs but the same metadata, remove one.

Expected Outcome:

- A structured, standardized dataset that allows for reliable mapping of job roles to required skills.
- Eliminated redundancy in job postings for cleaner data processing in subsequent steps.

Sprint 2: Skill Processing (Dependent on Sprint 1 Completion)

🔷 Ticket 3: Extracting & Mapping Skills to Job Titles

Final File: `skill_extraction.py`

Assignee: [TBD]

Task: Extract unique skills from `job_skills.csv` and map them to corresponding job titles and experience levels.

Input: `job_skills.csv`, `job_postings.csv`

Output: A structured mapping of skills to job titles and experience levels.

🔷 Ticket 4: Ranking & Categorizing Skills

Final File: `skill_extraction.py`

Assignee: [TBD]

Task: Rank skills by demand and group them into **categories and subcategories** (e.g., Programming → Python, Java).

Input: Mapped skill dataset.

Output: A ranked list of skills per job title and experience level within a structured hierarchy.

Sprint 3: Database Upload & API Development

🔹 Ticket 5: Upload Processed Data to MongoDB

Final File: `skill_extraction.py`

Assignee: [TBD]

Task: Upload all cleaned and structured data into a MongoDB database.

Input: Processed datasets containing job title, experience level, skills listed, date first seen, job location, and company.

Output: MongoDB collections with indexed data for fast querying.

🔹 Ticket 6: API Development - Skills Aggregation & Recommendations

Final File: `api_endpoints.py`

Assignee: [TBD]

Task: Create API endpoints for skill aggregation and personalized recommendations.

Input: API request with `job_title="Data Engineer"` and user-selected known skills.

Output: JSON response with ranked required skills and personalized skill recommendations.

Sprint 4: Dashboard Development & Visualization Using JavaScript (Dependent on API Completion)

🔷 Ticket 7: Implementing Data Visualizations with Recharts or Charts.js

Final File: `dashboard_visuals.js`

Assignee: [TBD]
Task: Develop interactive **skill ranking visualizations** using JavaScript libraries **Recharts** or **Charts.js**.

Potential Structure for the Dashboard Views

To enhance usability, the dashboard will be split into two key **views**:

1. **Data Science Job Market Overview View** – Provides high-level insights on job market trends and in-demand skills.
2. **Personalized Job Skill Assessment View** – Allows users to select a job title and evaluate their existing skills against job market demands.

Breakdown of Views & Features

View 1: Job Market Overview

- Displays an **aggregated view** of top skills across job titles and experience levels.
- **Visualizations:**
  - **Bar Chart:** Shows top 10 most in-demand skills across all job postings.
  - **Heatmap:** Illustrates the geographical distribution of job postings.
  - **Word Cloud:** Most frequently mentioned skills across all job descriptions within the Data Science Industry.
  -

View 2: Personalized Job Skill Assessment

- Allows users to **select a job title** and view specific skill recommendations.
- **Visualizations:**
  - **Bar Chart:** Displays the top 10 skills for the selected job title.
  - **Stacked Bar Chart:** Compares user-checked skills against recommended ones.
  - **Bubble Chart:** Correlation between experience level and skill demand.

Adding Interactivity: Allowing Users to Pick Job Titles from a List

To let users **filter results by job title**, we can implement:

- A **dropdown menu** (using a simple `<select>` HTML element or React's `useState` hook).
- An **autocomplete search bar** that fetches job titles dynamically from the database.
- An **event listener** that triggers updates to the displayed charts based on user selection.

Additional Enhancements

- Add **filters** to adjust experience level, job location, or company type.
- Enable **hover effects** on charts to display additional insights.
- Implement **tooltips** to provide explanations for each skill category.
- Create an option for users to **save their selected job role & skill set** for comparison.

Expected Outcome:

- Users can **interactively explore** the job market and understand the top **in-demand skills**.
- They can **assess their current skill set**, compare with industry demands, and identify missing skills.
- The dashboard delivers **actionable insights** for career progression and skill development.

EXTRAS: Cloud Integration with AWS

AWS Implementation Plan (Optional)

- **MongoDB Atlas for Cloud Database**
  - Use **MongoDB Atlas** to manage the database in the cloud.
- **AWS Lambda for Serverless Processing**
  - Automate **ETL processes** like data cleaning & updates.
- **AWS API Gateway + Flask Backend**
  - Deploy Flask API with **AWS API Gateway** for scalability.
- **AWS S3 + CloudFront for Frontend Hosting**
  - Host interactive dashboard on **AWS S3** with **CloudFront** for global access.

✅ Final Deliverables

- **MongoDB Database** storing job postings, skills, and experience levels.
- **Flask API** supporting job searches, skill recommendations, and personalized suggestions.
- **Interactive Dashboard** with skill ranking, filtering, and experience level selection.
- **GitHub Repository** with documented code and README.
