**Automated Job Application Workflow – Detailed Explanation**

**1.Problem Explanation**

Manually searching for jobs every day, tailoring cover letters, and tracking applications is **time-consuming and repetitive**. The goal of this project is to **automate the end-to-end job search process** by:

* Fetching relevant job postings daily from Google Jobs or Indeed.
* Matching them with your skills and resume.
* Generating a customized cover letter.
* Storing everything in Google Sheets.
* Sending a daily email summary of the top matches.

This workflow saves hours of manual effort and ensures consistency in job tracking.

**2.n8n Workflow Breakdown (Node-by-Node)**

 **Trigger Node – Cron**

* Runs daily at **10:00 AM IST**.
* Starts the entire workflow automatically.

 **HTTP Request (Fetch Jobs)**

* Calls **SerpAPI (Google Jobs)** or **JSearch/RapidAPI (Indeed)**.
* Query parameters: role, skills, location, and freshness (e.g., last 7 days).

 **Function Node (Normalize Listings)**

* Extracts required fields:
  + Job Title
  + Company
  + Location
  + Description
  + URL
  + Posted Date

 **LLM Node (Match Scoring)**

* Uses embeddings (e.g., Gemini API embeddings or OpenAI’s text-embedding-3-small).
* Compares each job description against your **resume skills**.
* Produces a similarity score (0–100).

 **LLM Node (Cover Letter Generation)**

* Prompt to Gemini:
  + Input: Job title, company, description, and your profile summary.
  + Output: Short tailored cover letter.

 **Google Sheets Node**

* Appends: job details, score, cover letter.
* Creates a structured record for tracking applications.

 **Gmail Node (Send Email Summary)**

* Sends an email with the **top N job matches**.
* Includes: Title, Company, Location, Score, and Job Link.

**3.Challenges & Solutions**

1. **API Rate Limits**
   * **Problem:** SerpAPI / RapidAPI have strict daily request limits.
   * **Solution:**
     + Cache results locally for testing.
     + Use pagination + only fetch recent jobs.
     + Implement retry with exponential backoff.
2. **Unstructured Job Data (Text Parsing)**
   * **Problem:** Job descriptions vary (extra tags, location format).
   * **Solution:**
     + Normalize fields using regex + JSON transformation.
     + Store only required fields in Google Sheets.
3. **Embedding Costs & Latency**
   * **Problem:** Processing every job listing with embeddings is expensive.
   * **Solution:**
     + Pre-filter jobs (keywords + location) before embedding.
     + Batch embedding requests.
4. **Email Formatting**
   * **Problem:** Long descriptions clutter email.
   * **Solution:**
     + Send only Title, Company, Location, Score, and Job Link in email.
     + Keep cover letter stored in Google Sheets.
5. **Summary of Learnings**

 **Automation saves time**: End-to-end automation drastically reduces manual job search effort.

 **APIs can be fragile**: Rate limits and inconsistent data formats require careful handling.

 **LLMs are powerful for personalization**: Embedding-based scoring + Gemini cover letters produce relevant and personalized outputs.

 **Tracking is key**: Storing everything in Google Sheets ensures visibility and history.

 **Scalability matters**: Efficient filtering + batching avoids unnecessary API calls and costs.