### Scope of the Code

* **Platform:** A Python-based backend prototype (to simulate the app’s logic), which can later be integrated into a mobile app.
* **Features:**
  + Resume parsing using NLP (spaCy).
  + Job matching using TF-IDF and cosine similarity (simplified ML model).
  + Application generation (simulated cover letter).
  + Tracking and notification simulation (console-based).
* **Limitations:**
  + No live job board scraping (due to legal and API restrictions; uses mock data instead).
  + No mobile app UI (focuses on backend logic; assumes a future React Native frontend).
  + Simplified feedback loop (manual updates rather than real-time ML retraining).
* **Purpose:** To validate the concept and provide a foundation you can expand for your project report or future development.

### Code Explanation

1. **Resume Parsing (**parse\_resume**):**
   1. Uses spaCy to extract skills, experience, and location preferences from a mock resume.
   2. Simplifies NLP by using keyword matching for skills and basic sentence parsing for experience and location.
2. **Job Matching (**match\_jobs**):**
   1. Uses TF-IDF and cosine similarity to match the resume summary to job descriptions.
   2. Returns a ranked list of jobs based on similarity scores.
3. **Application Generation (**generate\_application**):**
   1. Creates a basic cover letter tailored to the job, incorporating resume data.
   2. Outputs a pre-filled application ready for manual submission (simulating compliance with job board terms).
4. **Submission Simulation (**submit\_application**):**
   1. Prints the application details as a preview, simulating the manual submission step.
   2. Returns a "Submitted" status for tracking.
5. **Tracking and Notification (**track\_applications**):**
   1. Simulates status updates (e.g., "Under Review," "Interview Scheduled") with random assignments.
   2. Prints notifications to the console, mimicking push notifications.
6. **Feedback Loop (**update\_matching\_model**):**
   1. Analyzes tracking outcomes to suggest improvements in matching (e.g., increase weight for successful matches).
   2. Simplified as console output; a full implementation would retrain the ML model.
7. **Main Flow (**main**):**
   1. Orchestrates the entire process, from resume parsing to feedback, with console outputs for each step.

### Expanding to a Full App

To turn this into a complete mobile app for your project:

1. **Mobile Frontend:**
   1. Use **React Native** to build the UI, with screens for:
      1. Resume upload (file picker).
      2. Preferences input (text fields).
      3. Job matches display (list view).
      4. Application preview and submission (form view).
      5. Notification inbox (push notifications via Firebase).
   2. Replace console prints with UI updates.
2. **Backend:**
   1. Deploy the Python code as a Flask/Django API on a cloud service (e.g., AWS EC2, Heroku).
   2. Use REST endpoints (e.g., /parse\_resume, /match\_jobs) to connect the app to the backend.
3. **Job Scraping:**
   1. Integrate **Scrapy** with legal APIs (e.g., Indeed API, Naukri API if available) instead of mock data.
   2. Handle rate limits and terms of service compliance.
4. **Advanced ML:**
   1. Replace TF-IDF with a neural network (e.g., BERT via Hugging Face) for better matching.
   2. Implement real-time retraining for the feedback loop using user outcomes.
5. **Notifications:**
   1. Use **Firebase Cloud Messaging (FCM)** for push notifications instead of console prints.
6. **Database:**
   1. Store user data and application statuses in a database (e.g., SQLite for prototype, PostgreSQL for production).

### Output Sample

Running the code produces output like this (simplified):

text

CollapseWrapCopy

=== AI-Powered Job Application Assistant Prototype ===

1. Parsing Resume... Extracted Skills: ['Python', 'Machine Learning', 'Java'] Experience: 5 years Location Preference: Bangalore, Delhi Resume Summary for Matching: Python Machine Learning Java 5 years
2. Matching Jobs... Top Matched Jobs: Job: Python Developer (ID: JOB001), Score: 0.85, Location: Bangalore Job: Data Scientist (ID: JOB003), Score: 0.62, Location: Delhi Job: Front-end Developer (ID: JOB002), Score: 0.12, Location: Mumbai
3. Generating Applications...
4. Submitting Applications... --- Application Preview --- Job ID: JOB001 Resume Summary: Python Machine Learning Java 5 years Cover Letter: Dear Hiring Manager, I am excited to apply for the Python Developer position at your company (Job ID: JOB001). With my 5 years in Python, Machine Learning, Java, I believe I am a strong fit for this role. The job description mentions Looking for a Python developer with machine learning experience, and I have hands-on experience in these areas. Thank you for considering my application. I look forward to the opportunity to discuss further. Sincerely, [Your Name] Status: Pre-filled form ready for manual submission
5. Tracking Applications... [Notification] Job ID: JOB001 - Status Updated: Interview Scheduled [Notification] Job ID: JOB003 - Status Updated: Under Review 6. Updating Matching Model... Job ID: JOB001, Score: 0.85,

Outcome: Interview Scheduled Positive feedback: Increase weight for this match. Job ID: JOB003, Score: 0.62, Outcome: Under Review === Process Complete ===