Kowshik - SE_1

Perfect — I see your entire **database schema** clearly now!
Let's document **exactly what happens in your DB from start to end** in your AI Resume Screening system.

🧩 Full Database Flow – Step-by-Step Explanation

---

1. 👩‍💻 User signs up / logs in (Frontend → Supabase Auth)

- User authenticates via **Google** or **Email/Password** through Supabase Auth.
- After login:
  - A **JWT token** is issued
  - Frontend stores the token
- User's `user_id`, `email`, `name`, and **role** (`recruiter` or `student`) are stored inside the `users` **table.**

users table

Column	Purpose
`user_id`	Unique ID of user (primary key)
`email`	Email address
`password_hash`	(Optional for password login)
`name`	User's name
`role`	recruiter or student
`created_at`	Timestamp of registration

2. ➕ Recruiter adds a new screening (job description + uploads resumes)

Backend route:

```http
POST /upload-resumes/
```

- **Creates a new entry in `job_descriptions` table**:
  - `job_id`
  - `user_id` (recruiter)
  - `job_title`
  - `job_description`
  - `project_weight`
  - `experience_weight`
  - `created_at`

job_descriptions table

Column	Purpose
`job_id`	Unique ID of the screening/job
`user_id`	Linked recruiter
`job_title`	Title for the opening
`job_description`	Full JD content
`skills_required`	(optional if added later)
`created_at`	Created timestamp
`project_weight`	Weightage for project scoring
`experience_weight`	Weightage for experience scoring

✅ **At this point, a "job post" is live for screening resumes.**

---

3. 📄 Resume Uploads (ZIP Extraction + Storage)

- Each extracted resume file becomes a **new entry in** `resume_uploads`.
- For every uploaded file:
  - `resume_id` is created
  - `user_id` (recruiter)
  - `job_id` linked
  - `file_name` stored
  - `file_path` stored
  - `upload_timestamp`
  - (Optional) `original_hash` (for tracking if same resume uploaded multiple times)

resume_uploads table

Column	Purpose
`resume_id`	Unique per resume file
`user_id`	Who uploaded (recruiter)
`job_id`	Linked job
`file_name`	Resume filename
`file_path`	Resume file storage path
`upload_timestamp`	When uploaded
`original_hash`	Duplicate checking (future use)

---

4. 🧠 Resume Analysis by LLM (AI Screening)

- Each resume is passed to **Mistral LLM** (Mixtral / Mistral-Saba-24B).
- The model returns:
  - `Key Skills`
  - `Relevant Projects`
  - `Certifications`
  - `Soft Skills`
  - `Overall Match Score`
  - `Projects Relevance Score`
  - `Experience Relevance Score`
- This data is saved into `resume_analysis` **table**, linked to each resume.

resume_analysis table

Column	Purpose
`analysis_id`	Primary Key
`resume_id`	Which resume analyzed
`key_skills`	List of detected skills
`overall_analysis`	Free text analysis
`certifications_courses`	Certs from resume
`relevant_projects`	Projects from resume
`soft_skills`	Detected soft skills
`overall_match_score`	Overall fit score (0–10)
`projects_relevance_score`	Project relevance score
`experience_relevance_score`	Experience relevance score
`analysis_timestamp`	Timestamp
`notes`	Internal notes added later
`tagged_users`	User IDs tagged for resume

Column	Purpose
`search_vector`	Full text search (Postgres)

5. 🧮 Resume Ranking Calculation

- Each resume gets a `total_score` calculated as:

```makefile
total_score = experience_weight * experience_relevance_score
              + project_weight * projects_relevance_score
              + 100 * overall_match_score
```

- This ranking data is saved into `resume_rankings` **table.**

resume_rankings table

Column	Purpose
`ranking_id`	Unique
`resume_id`	Linked resume
`job_id`	Linked job/screening
`total_score`	Computed final score
`rank`	Position (1st, 2nd, 3rd, etc.)
`created_at`	When ranked

✅ Now recruiters can **see candidates ranked**, **search**, **export CSV**, etc.

6. 👩‍🎓 Student User Side (Future - Mock Interviews)

- Students will **login** with role = `student`

- When mock interviews are built:
  - New tables will track mock interview questions, answers, scores
- `user_id` will link the student's performance back to `users`

(No impact on current tables — only new tables added for mock interviews.)

✍️ Final Full Data Flow Summary:

Action	Table Affected
User Login/Register	users
Add Screening	job_descriptions
Upload Resumes	resume_uploads
Analyze Resume	resume_analysis
Score & Rank Resumes	resume_rankings
Add Notes / Search	resume_analysis

✅ Your Current DB is:

- 🚀 Ready for full AI Resume Screening
- 🛡️ Secure with Recruiter vs Student separation
- 🧠 Extendable easily to Mock Interviews next

Would you also like me to make a **beautiful flowchart diagram** showing these relationships visually? 📈🗺️ (Would be awesome for your documentation/handoff!)

Shall I? 🎯