Kowshik - SE_1

Excellent — you're spot on:
If resumes are **extracted locally** and **not uploaded to Supabase**, the frontend won't be able to:

- Preview resumes
- Display results correctly
- Persist files across restarts

Let's fix this **completely and properly**, so the resumes are:

- ✅ Uploaded to **Supabase Storage bucket**
- ✅ Parsed and analyzed using the LLM
- ✅ Saved with permanent URLs
- ✅ Available to frontend via API

✅ What Needs to Happen (High-Level)

Step	Old (Broken)	✅ New (Correct)
1. Extract ZIP	Extracts to local `resumes/` folder	Extract to `resumes/{job_id}` temp folder
2. Parse Resume	Parsed from disk	✅ Parse from disk (temporary is fine)
3. Upload Resume	❌ Only on disk	✅ Upload to Supabase Storage after parsing
4. Save path	❌ Local file path	✅ Supabase Public URL stored in DB
5. Delete local copy	❌ Accumulates on disk	✅ Delete after upload

---

✅ What You Need to Update (Backend)

✅ 1. Modify `process_all_resumes()` and `upload_resume_info_to_db()`

Full correct pipeline:

1. **Extract ZIP to temp folder**
2. For each file:
    - Parse it to text
    - Upload it to Supabase Storage (e.g. `resumes/{job_id}/{filename}`)
    - Store `file_path` (public URL) in `resume_uploads` table
    - Send text to LLM for analysis
3. Save analysis in `resume_analysis`

✅ Step-by-Step Code Summary

🧩 1. In `process_resumes.py`:

✅ Extract and process:

```python
extract_zip(zip_path, resume_output_folder)

resumes = []
resume_id_map = {}

for file in os.listdir(resume_output_folder):
    path = os.path.join(resume_output_folder, file)
    text = read_resume(path)

    # ✅ Upload to Supabase
    resume_id = upload_resume_info_to_db(file_name=file, file_path=path,
job_id=job_id, user_id=user_id)
```

```
    # ✅ Append for LLM analysis
    if resume_id:
        resumes.append({"filename": file, "text": text})
        resume_id_map[file] = resume_id

    # ✅ Delete local after upload
    os.remove(path)
```

✅ 2. Update `upload_resume_info_to_db()`:

We already shared this, but to recap:

```python
def upload_resume_info_to_db(file_name, file_path, job_id, user_id):
    resume_id = str(uuid.uuid4())
    with open(file_path, "rb") as f:
        file_content = f.read()

    content_type, _ = mimetypes.guess_type(file_path)
    if content_type is None:
        content_type = "application/pdf"

    storage_path = f"{job_id}/{file_name}"

    # ✅ Upload to Supabase Storage
    supabase.storage.from_("resumes").upload(
        path=storage_path,
        file=file_content,
        file_options={"content-type": content_type, "upsert": True}
    )

    public_url = f"https://{SUPABASE_URL.split('https://')
[1]}/storage/v1/object/public/resumes/{storage_path}"

    # ✅ Save to resume_uploads table
    data = {
        "resume_id": resume_id,
        "user_id": user_id,
        "job_id": job_id,
        "file_name": file_name,
        "file_path": public_url,
    }

    supabase.table("resume_uploads").insert(data).execute()
    return resume_id
```

✅ 3. Final Flow (With Analysis)

```python
results = []
for resume in resumes:
    analysis = analyze_resume_mistral(resume["text"], job_description)
    results.append({
        "filename": resume["filename"],
        "analysis": analysis
    })
```

✅ Save `results` into `resume_analysis.json` and push to DB just like you're doing now.

✅ For Frontend Developer

They can now safely:

- Fetch results using `/export?job_id=...`
- Each candidate will have:

  ```json
  {
    "upload": {
      "file_name": "resume1.pdf",
      "file_path": "https://...supabase/resumes/job_id/resume1.pdf"
    },
  ```

```
   ...
}
```

- Use `file_path` to preview resumes using an `<iframe>` or PDF.js viewer

✅ Summary of Required Backend Changes:

Area	Action
Resume Extraction	Still to temp `/resumes/{job_id}/`
Resume Parsing	Same — extract & parse locally
Upload to Supabase	✅ After parsing but before deletion
Store public URL	✅ Save in `resume_uploads.file_path`
Delete local	✅ Prevents clutter

Would you like me to bundle the full final working `process_resumes.py` as a ZIP and PDF and send to you for handoff? 🎁✅
It'll make sharing easier.