# K_1 - SE_2

Here's your **complete backend roadmap**, broken down into what's done ✅ and what's left 🚧 — all in brief and actionable form.

---

## ✅ Done So Far

- ✔️ **Resume Upload** to Supabase + text extraction
- ✔️ **Groq-based Interview Question Generation**
- ✔️ **Sequential Question Retrieval**
- ✔️ **Video-based Stress Detection** (emotion model + mapping)
- ✔️ **Per-question + Average Stress Score Storage**
- ✔️ **Supabase Integration**
- ✔️ **Basic Error Handling & Logging**

---

## 🚧 What More to Do (Backend Only)

---

### 🔹 1. Modular Refactor

> 🎯 Make codebase production-ready and maintainable

- Organize into `/api`, `/core`, `/utils`, `/models`, `/config`
- Separate Groq logic, Supabase utils, model loading, etc.

**Use:** Standard Python file structure + FastAPI `APIRouter`

---

### 🔹 2. Submit Answer Endpoint

> 🧠 Store user answers per question (for analysis)

- `POST /submit-answer/{session_id}/{question_number}`
  - Accepts **text/audio**
  - Stores it in Supabase table `mock_interview_answers`

**Use:** Supabase + optional transcription (Whisper API) for audio

---

## 🔹 3. Answer Evaluation (Optional but Powerful)

> 🧠 Analyze and score answers for feedback

- Use Groq (or OpenAI) to evaluate:
  - Clarity, completeness, relevance
- Store feedback per answer (optional for now)

**Use:** Groq API + prompt like:

> "Evaluate this candidate answer based on communication, clarity, and depth. Score 1-10 and provide feedback."

---

## 🔹 4. Final Interview Report API

> 📊 Generate a complete report for a session

- `GET /final-report/{session_id}`
  - Includes:
    - All questions
    - Answer (if submitted)
    - Stress score per question
    - Feedback (if answer analyzed)
    - Overall summary + recommendations

**Use:** Combine data from Supabase tables + generate JSON

---

### 🔹 5. Session Tracking & Stats

> 📈 Track total sessions, questions attempted, progress

- `GET /user-summary/{mock_user_id}`
  - Returns:
    - **of sessions**
    - Avg. stress trend
    - Weakest question types
    - Progress over time

**Use:** Supabase aggregation queries

---

### 🔹 6. Admin Utilities (Optional)

> ⚙️ For you to monitor & manage backend

- `GET /admin/sessions` → View all sessions
- `DELETE /session/{session_id}` → Cleanup old data
- Logs & performance summaries

---

## 🔧 Suggested Tools & Libraries

| Feature | Tool/Library |
| --- | --- |
| API Framework | FastAPI |
| Database | Supabase (PostgreSQL) |

| Feature | Tool/Library |
| --- | --- |
| Video Processing | OpenCV |
| Emotion Model | TensorFlow (MobileNetV2) |
| AI Text Generation | Groq API (Mixtral-8x7B) |
| Answer Evaluation (optional) | Groq or OpenAI |
| Speech-to-Text (optional) | Whisper API |
| Env & Secrets | python-dotenv |
| File Parsing | PyMuPDF (fitz) |

## 🗂️ Supabase Table Additions

| Table | Fields |
| --- | --- |
| `mock_interview_answers` | `id`, `session_id`, `question_number`, `answer_text/audio_url`, `score`, `feedback`, `created_at` |
| `mock_interview_reports` | `session_id`, `overall_summary`, `final_score`, `recommendation`, `created_at` |

## 🎯 Final Output from Backend (What Frontend Will Use)

1. **Resume Upload URL**
2. **List of Questions**
3. **/next-question → question text**
4. **/analyze-stress → stress score/level**
5. **/submit-answer → stores response**
6. **/final-report → JSON with full session summary**

7. **/user-summary → user dashboard metrics**

---

Let me know if you want:

- JSON schema for `/final-report` and `/submit-answer`
- Code stubs for missing routes

You're in the **final lap** of backend. Nail this, and the project's backend will be solid for production or publication. 🚀