🚦 **Critical Python Programs (must be running as services)**

These are the ones that become actual running processes when you start the system:

- `main.py` **(Flask Orchestrator, port 5000)**
  - **Entry point for Twilio calls (`/phone/incoming`) and the Admin Panel (`/admin`).**
  - **Boots the FastAPI backend (`app/main.py`) in a background thread.**
  - ✅ **Critical.**
- `app/main.py` **(FastAPI backend, port 8001)**
  - **Provides `/v1/chat/completions`, `/health`, etc.**
  - **Handles AI + memory calls once the Orchestrator passes input.**
  - ✅ **Critical (but started by `main.py`, not run separately).**
- `ai-memory/main.py` **(separate repo/service, port 8100)**
  - **Memory service, not in this repo screenshot but running on your droplet.**
  - ✅ **Critical (separate Python process via `ai-memory.service`).**
- `voice-bridge/app.py` **(separate repo/service, port 9100)**
  - **Handles Twilio WebSocket audio → STT → sends transcript to Orchestrator.**
  - ✅ **Critical (separate Python process via `voice-bridge.service`).**

Core Services (must be active)

- **Nginx Reverse Proxy**
  - Routes Twilio HTTPS + WebSocket traffic to the right internal ports.
  - Must be active (`systemctl status nginx`).
- **VoiceBridge (port 9100)**
  - Converts Twilio WebSocket audio → STT transcripts.
  - Forwards those transcripts to the Orchestrator.
  - Must be active (`systemctl status voice-bridge`).
- **Orchestrator (Flask/FastAPI, port 5000/8000)**
  - The “traffic cop.”
  - Handles greeting, memory lookup, LLM call, TTS call.
  - Must be active (`systemctl status orchestrator`).
- **AI-Memory (FastAPI, port 8100)**
  - Stores and retrieves caller/user context.
  - Must be active (`systemctl status ai-memory`).
- **LLM (currently OpenAI GPT-4o-mini / gpt-realtime)**

    - Generates the natural language response.
    - Must be reachable (curl to `https://api.openai.com/v1/chat/completions`).
- **ElevenLabs TTS**
    - Converts text → speech.
    - Must be reachable (curl test returns playable MP3).

🖥️ **Admin Panel / Interfaces (should be reachable)**

- **Only one admin interface exists: `/admin` (or `/admin.html`).**
- **This single page handles everything:**
    - 🎤 **Voice settings (ElevenLabs ID, stability, clarity).**
    - 🧠 **AI personality/instructions.**
    - 📞 **Call routing triggers.**
    - 🗄️ **Knowledge & memory management.**
    - ⚙️ **System status & configuration sources.**

Bottom line: instead of juggling two panels, you just use one combined Admin Panel for both controls and status.

    - Dashboard to update voice ID, personality, and LLM backend.
- **Admin Status (`/admin-status`)**
    - JSON health snapshot: memory count, LLM endpoint, ElevenLabs voice, config sources.

Critical Support Files (not running by themselves, but needed)

    - **`config.json`** → Active config (points to OpenAI, ai-memory URL, ElevenLabs voice ID).
    - **`config-internal.json`** → Documents internal ports/services for operators.
    - **`config_loader.py`** → Code that reads `config.json` and environment variables.
    - **`Dockerfile`**, **`docker-compose.yml`** → Deployment environment setup.
    - **`requirements.txt`**, **`pyproject.toml`** → Python dependencies.
    - **`nginx.conf`** → Reverse proxy config for Twilio ↔ Orchestrator.
    - ---

Optional / Helper Scripts (used when fixing, not running 24/7)

    - **`fix_llm_config.sh`**, **`fix_runpod_endpoint.sh`**, **`fix_voice_memory_speed.sh`**, **`fix_hangup_bug.sh`** → **Repair scripts for known issues.**

- `deploy.sh`, `start_server.py`, `run_app.py` → **Startup helpers (for dev/prod).**
- `init_db.py` → **One-time database initialization.**
- `demo_test.py` → **For testing locally.**
- ---

📄 **Non-Critical Docs / Metadata**

- **Markdown docs:** `AI_Phone_System_Technical_Documentation.md`, `deployment-guide.md`, **etc.**
- **Logs:** `server.log`.
- **Package files:** `package.json`, `uv.lock`, **etc. (dev ecosystem)**

External Services (must be reachable)

- **Twilio**
  - Active phone number configured.
  - Webhook set to `https://voice.theinsurancedoctors.com/phone/incoming`.
  - Logs show 200 OK responses.
- **OpenAI (LLM)**
  - Current model: `gpt-4o-mini` / `gpt-realtime`.
  - Endpoint: `https://api.openai.com/v1/chat/completions`.
  - Test: `curl` with your `OPENAI_API_KEY` returns JSON.
- **ElevenLabs (TTS)**
  - Endpoint: `https://api.elevenlabs.io/v1/text-to-speech/<VOICE_ID>`.
  - Test: `curl` with `ELEVENLABS_API_KEY` returns playable MP3.

DigitalOcean's Role

- **Droplet (your VM server)**
  Runs Ubuntu 22.04. This is where you've deployed Orchestrator, VoiceBridge, and AI-Memory.
- **Networking**
  Provides your public IP + DNS entries (e.g., `voice.theinsurancedoctors.com`) which Twilio hits.
- **Firewall / Ports**
  Needs to allow inbound traffic on:
  - 22 (SSH)
  - 80/443 (Nginx reverse proxy, HTTPS termination)
  - 8100 (AI-Memory, internal only)
  - 9100 (VoiceBridge, internal only)

✅ **What "ON" Means for DigitalOcean**

        - **Droplet is powered on** (check in DO dashboard or with `uptime`).
        - **System services running** (`systemctl status nginx`, `systemctl status voice-bridge`, `systemctl status ai-memory`, etc.).
        - **Disk, CPU, memory usage** within safe levels (`htop` or DO monitoring).
        - **Networking working** (can ping the droplet, `curl` to domains resolves).
        - **SSL Certificates valid** (Let's Encrypt certs auto-renew via DO/Certbot).

📂 **`app/` Modules**

- **`http_memory.py`** → Connector for your **AI-Memory** HTTP service (port 8100). Handles requests like `/memory/retrieve` and `/memory/store`.
- **`llm.py`** → Handles calls to the LLM (currently OpenAI GPT-4o-mini / realtime API). Formats requests and parses completions.
- **`main.py`** → The **FastAPI backend** that exposes `/v1/chat`, `/health`, etc. This runs alongside the Flask orchestrator to process AI requests.
- **`memory.py`** → Defines memory store logic (used to be Postgres/pgvector directly, now adapted to HTTP memory).
- **`models.py`** → Pydantic models for request/response validation (e.g., ChatRequest, ChatResponse).
- **`packer.py`** → Prompt engineering: takes user input + memory, builds the final prompt before sending to LLM.
- **`tools.py`** → Tool calling framework: external integrations (e.g., booking, sending messages). Allows LLM to trigger structured actions.

What's Missing or Could Be Added

- **Twilio Credentials & Connectivity**
    - You list Twilio webhooks and logs, but also confirm:
        - `TWILIO_ACCOUNT_SID` and `TWILIO_AUTH_TOKEN` are set as environment variables.
        - Outbound REST API calls (if you're doing outbound dialing) succeed.
- **Database (Postgres/pgvector behind AI-Memory)**
    - AI-Memory uses Postgres on DO. Checklist should confirm:
        - Database is reachable.
        - `DATABASE_URL` env is set.
        - No connection errors in ai-memory logs.
- **Certbot / SSL Renewal**
    - You noted SSL valid, but I'd add:
        - Confirm `certbot renew` is set up (cron or systemd timer).
- **Log Monitoring**
    - Make sure you can quickly check:

        - Orchestrator logs.
        - VoiceBridge logs.
        - AI-Memory logs.
    - Helpful for tracing call failures.
- **Resource Limits**
    - Not just “htop looks good,” but:
        - Verify Uvicorn workers aren’t capped.
        - Memory usage under thresholds.
- **Fallback Paths**
    - If OpenAI or ElevenLabs fail: what happens? (e.g. Twilio silence vs. fallback voice). Might be worth noting in the checklist.

📝 **Suggested Add-On Checklist Items**

- `echo $TWILIO_ACCOUNT_SID` / `echo $TWILIO_AUTH_TOKEN` present in env.
- `echo $OPENAI_API_KEY` present and working.
- `echo $ELEVENLABS_API_KEY` present and working.
- `systemctl status certbot` (or cron entry) confirms SSL renewal.
- `journalctl -u orchestrator -f` shows no 500 errors on incoming calls.
- `psql` test or `curl http://127.0.0.1:8100/health` confirms Postgres/AI-Memory OK.