FREE RESOURCE

Free FastAPI Debugging Cheatsheet

A compact 2-page checklist for diagnosing the most common FastAPI production failures. Use this when your API returns 500s, behaves differently in prod, or becomes unreliable under load.

60■Second Triage	**Goal:** decide whether this is **environment** or **code** before changing anything.
1) Reproduce	Hit the smallest failing request (curl/Postman). If only frontend fails → isolate backend endpoint.
2) Read logs	Use server logs/tracebacks (Uvicorn/Gunicorn). Never debug from the client response only.
3) Reduce scope	Temporarily return a dummy payload. If it works → the bug is inside your logic/DB/deps.

1) Environment mismatch

- Confirm you run the app from the **project root** and the same entrypoint in prod (e.g., **uvicorn app.main:app**).
- Compare **Python version**, dependency versions, and build command between local and prod.
- Print/inspect critical env vars at startup (never secrets): **DATABASE_URL present?** correct hostname? correct scheme?
- If it works locally but fails in prod: treat it as **deployment config** until proven otherwise.

2) SQLModel session issues

- Never create a DB session globally at import time. Each request should get its own session via dependency injection.
- Sync pattern: **yield** a session and always close it in **finally**. Async pattern: **async with** session factory.
- Symptoms: random 500s, connection exhaustion, "event loop is closed", stale data, hanging requests.
- Quick test: add a log line when creating/closing sessions and verify one session per request.

3) Misconfigured dependencies (DI)

- Never instantiate heavy objects in default parameters (e.g., **repo=Repo()**). Use **Depends**.
- If a dependency reads env/settings, make sure it's evaluated at the right time (not at import time).
- Common symptom: **TypeError missing argument** or dependency not injected as expected.
- Temporarily bypass dependency chain to isolate: call the underlying function with a mocked object.

4) Logging blind spots

- Ensure your server prints tracebacks: run with **--log-level debug** locally.
- Log one line at request start + after DB call. You want a breadcrumb trail across the endpoint.

- Avoid logging sensitive payloads. Prefer correlation id + key fields + error stack.
- If prod logs show nothing, your issue is **logging configuration** (not the endpoint).

5) Async traps

- Do not call blocking I/O inside async routes (e.g., requests). Use async clients or move to background tasks.
- Do not mix sync SQLAlchemy sessions with async endpoints unless you know the pattern.
- Symptoms: hanging endpoints, timeouts, tasks never finishing, weird concurrency issues.
- Quick isolation: convert the endpoint to sync temporarily or wrap blocking calls properly.

Next step: If this cheatsheet helps, the paid pack contains the full debugging blueprint plus deeper recipes and architecture patterns.

Practical fixes you can apply immediately

A) Get a full traceback (local)

- Run: **uvicorn app.main:app --reload --log-level debug**
- Reproduce with curl/Postman; copy the traceback line that points to your code.
- If the traceback points to imports/settings → fix structure before touching business logic.

B) Minimal reproduction drill

- Replace endpoint body with **return {'ok': True}**. If it works, the failure is inside your logic/DB/deps.
- Re-introduce logic in small increments (DB call → transform → response model).
- When it breaks again, you found the exact failing step.

C) Session lifecycle sanity check

- Count DB connections in logs; connection count growing over time usually means sessions are not closed.
- Ensure session dependency uses **yield** and closes in **finally** (sync) or **async with** (async).
- If you use pooling, confirm pool size fits your server workers/threads.

D) Import/circular dependency quick test

- Start the app fresh after clearing caches. If import errors appear, simplify module boundaries.
- Rule: routers import services; services import repos; repos import models. Avoid reverse imports.
- Move settings/logging initialization out of module top-level into startup.

E) Production checklist

- Confirm runtime: working directory, command, and env vars in your host (Render/Vercel/Docker).
- Confirm packages installed match your lockfile. A missing dependency often shows up as import error.
- If prod-only bug: compare logs first, then config, then code.

Want the full playbook?

FastAPI Backend Pack #1 (29 €) includes the full Debugging Blueprint plus 10+ in-depth articles on SQLModel, async patterns, logging, dependency injection and architecture.

Get the pack: silentgpt.gumroad.com/l/fastapi-backend-pack-1

Tip: Put this cheatsheet into your repo as **docs/fastapi-debugging-cheatsheet.pdf** and link it in your internal onboarding.