Auth

July 31, 2026

Auth

What it is responsible for

The Auth subsystem manages entry through `AuthModule` and `AuthController`, where `signIn` and `getProfile` are named work paths. `AuthService`, `AuthGuard`, `Public`, `jwtConstants`, and `IS_PUBLIC_KEY` mark the stated authentication-facing surface. The evidence links refusal decisions to a token check and a comparison of `user?.password` with `pass`; it does not state output formats, successful-call results, or further policy. It names no relationship among those members beyond their presence in Auth as stated here alone.

What it refuses

The subsystem rejects work with `UnauthorizedException` when `!token`. It rejects work with `UnauthorizedException` when `user?.password !== pass`. These are the stated refusal conditions. No other rejection cases or exception messages are evidenced. The evidence does not attach a different message to either exception case.

What it needs, and who needs it

Auth depends on Core. Without Core, Auth lacks its declared dependency; the evidence does not name the Core members involved. Sample depends on Auth. Without Auth, Sample lacks the subsystem it declares as a dependency, including `AuthModule`, `AuthController`, `signIn`, and `getProfile`. The evidence does not identify Sample callers, the exact Core use, or effects beyond those declared dependency relationships. `AuthService`, `AuthGuard`, `Public`, `jwtConstants`, and `IS_PUBLIC_KEY` are named Auth members, without stated dependency direction here.

9 entities in `sample/19-auth-jwt/src/auth`. **1 other subsystem depends on it**, which makes it the 7th most depended-upon part of this codebase.

What it is made of

Its 9 entities sit in 6 files under `sample/19-auth-jwt/src/auth`: 2 HTTP endpoints, 2 services, 2 constants, 1 controller and 2 more.
`auth.controller.ts` holds 3 of them — more than any other file here.

Where work enters

1 controller publishes 2 HTTP endpoints — 1 `POST` and 1 `GET`. They answer under `/auth`. None of them declares a guard.

- `AuthController` — `sample/19-auth-jwt/src/auth/auth.controller.ts`:13
- `signIn` — `sample/19-auth-jwt/src/auth/auth.controller.ts`:17
- `getProfile` — `sample/19-auth-jwt/src/auth/auth.controller.ts`:24
- `AuthModule` — `sample/19-auth-jwt/src/auth/auth.module.ts`:10

How it refuses and fails

2 of its components record a refusal or a failure handler.
All 2 of them refuse work outright, under a condition written into the component itself.
Their `catch` blocks handle a failure that already happened in 1 place.

Boundaries

1 other subsystem depends on this one — `Sample`. Changing what it exposes changes them.

They hold 1 edge into it between them. 1 edge arrives and 1 leaves. What they reach is narrower than the folder: 1 of its 9 members carries every inbound edge — `AuthModule` (1).

It depends on `Core`, and on nothing else in this repository.