

[Get started](#) · [Exercises](#)

Exercise 02: add a route and watch the gates

🕒 2 min read · Updated 25 September 2026

Add a harmless read-only route in a scratch branch and let the tests, the OpenAPI check and the generated routes page show every place a route must be declared — then throw it away.

<https://docs.gurbanisoul.com/exercises/02-add-a-route/>

Goal. Learn where a route lives by adding one and watching what breaks. **Needs.** the local setup, a scratch branch. 60 minutes. You will not keep the route.

1. A scratch branch

```
git switch -c scratch/exercise-02 integration
```

2. Add the handler

In `webapp/sogs/reader.py`, after `_route_meta`, add a handler that returns the counts the health check already knows:

```
def _route_counts(p, qs):
    return {'lines': db().execute('SELECT count(*) FROM lines').fetchone()[0],
            'angs': db().execute('SELECT count(DISTINCT ang) FROM lines').fetchone()[0]}
```

Register it in `webapp/serve.py`'s `ROUTES` table as `('counts', None): (_route_counts, 'reader')` (import it beside the other reader handlers). Start the server and call `curl -s http://127.0.0.1:7777/api/counts`. Expected: `{"lines": 60658, "angs": 1430}`.

3. Watch the gates

```
make test-web
```

Expected: **failures** — and each one names a place you have not yet declared the route:

- the OpenAPI test: `contract/openapi.json` does not cover every dispatcher route → declare it in `tools/gen_openapi.py:ROUTES` and run `make openapi`;
- the declared-tables test: your handler read `lines` — the reader context declares it, so this one passes; had you read another table, it would have failed and told you to add it to `TABLES`;

- the routes-page check: `python3 tools/gen_route_table.py --check → regenerate docs/api/routes.md.`

Fix each, rerun `make test-web`, and confirm it is green. Then run `make docs-check`: the generated page is current again.

4. Throw it away

```
git switch integration && git branch -D scratch/exercise-02
```

Self-check

Where must a new route be declared for the build to be green?

- ☐ The dispatcher, the OpenAPI declarations, and (if it reads a new table) the context's TABLES; the routes page is regenerated
- ☐ Only `serve.py`
- ☐ Only `gen_openapi.py`

Your handler runs a query against a table the context does not declare. What happens?

- ☐ `test_declared_tables` fails: the route ran under an authorizer that denies undeclared reads
- ☐ It works in the monolith and fails only in production
- ☐ SQLite raises at import time