# OpenAPI for Web Developers

Lorna Mitchell

@lornajane@indieweb.social

## About OpenAPI

Standard for describing APIs, for machines and for humans.

# OpenAPI Example

```yaml
openapi: 3.0.1
servers:
  - url: http://datasette.local
  - url: https://datasette.io
info:
  description: Execute SQL queries against a Datasette database
      and return the results as JSON
  title: Datasette API
  version: v1
paths:
  /content.json:
    get:
      description: Accepts SQLite SQL query, returns JSON. Does
          not allow PRAGMA statements.
```

Credit: https://github.com/APIs-guru/openapi-directory

## Code-First vs Design-First

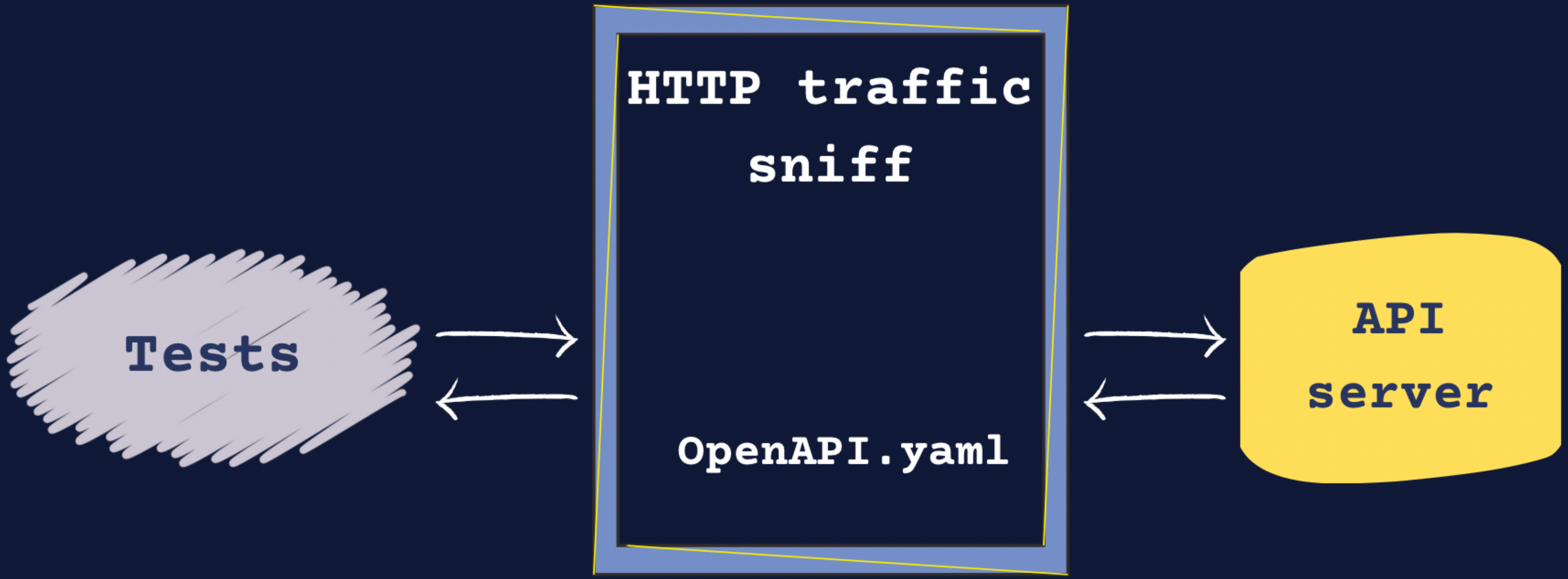
## If you wrote the API code already

Get an OpenAPI file of what you have, and then start using it design-first.

Two great options:

- generate from your codebase
- use a learning/sniffing tool like Optic

## If you wrote the API code already

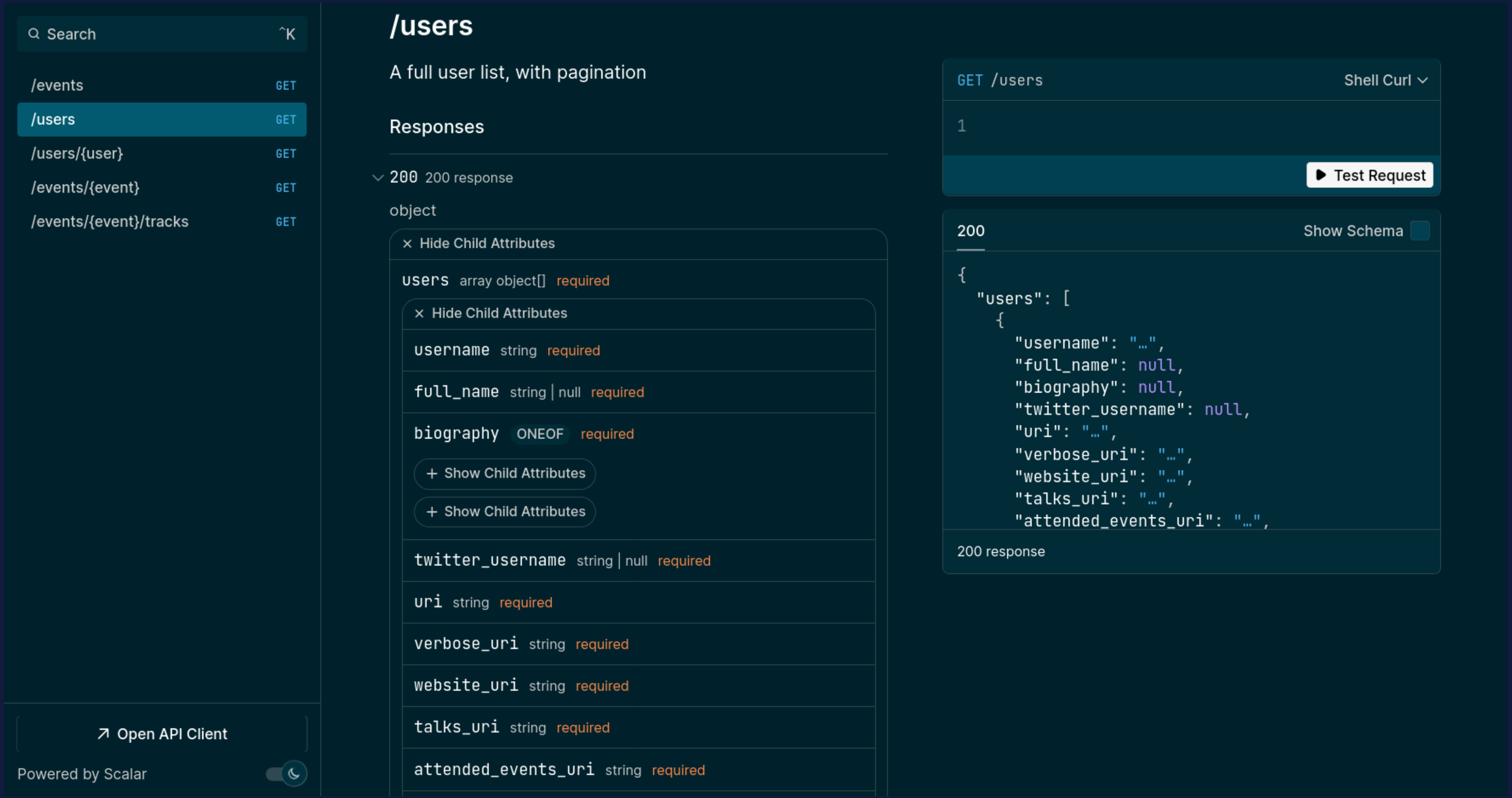


## If you have humans using your API

Make the OpenAPI human-readable with docs!

- Scalar (next-generation docs)
- Redoc (traditional docs)
- Bump.sh (short-lived hosted docs)

# If you have humans using your API



/users

A full user list, with pagination

Responses

200 200 response

object

× Hide Child Attributes

users array object[] required

× Hide Child Attributes

| | | |
|---|---|---|
| username | string | required |
| full_name | string \| null | required |
| biography | ONEOF | required |
| twitter_username | string \| null | required |
| uri | string | required |
| verbose_uri | string | required |
| website_uri | string | required |
| talks_uri | string | required |
| attended_events_uri | string | required |

```
{
  "users": [
    {
      "username": "…",
      "full_name": null,
      "biography": null,
      "twitter_username": null,
      "uri": "…",
      "verbose_uri": "…",
      "website_uri": "…",
      "talks_uri": "…",
      "attended_events_uri": "…",
```

@lornajane@indieweb.social

## If your OpenAPI needs editing

Designed OpenAPIs can be edited safely

But generated ones do have options.

- OpenAPI Overlay Specification is a standard for describing edits
- Speakeasy and Bump.sh have good Overlay tools
- Look out for other transformation tools

## Repeatably edit an OpenAPI

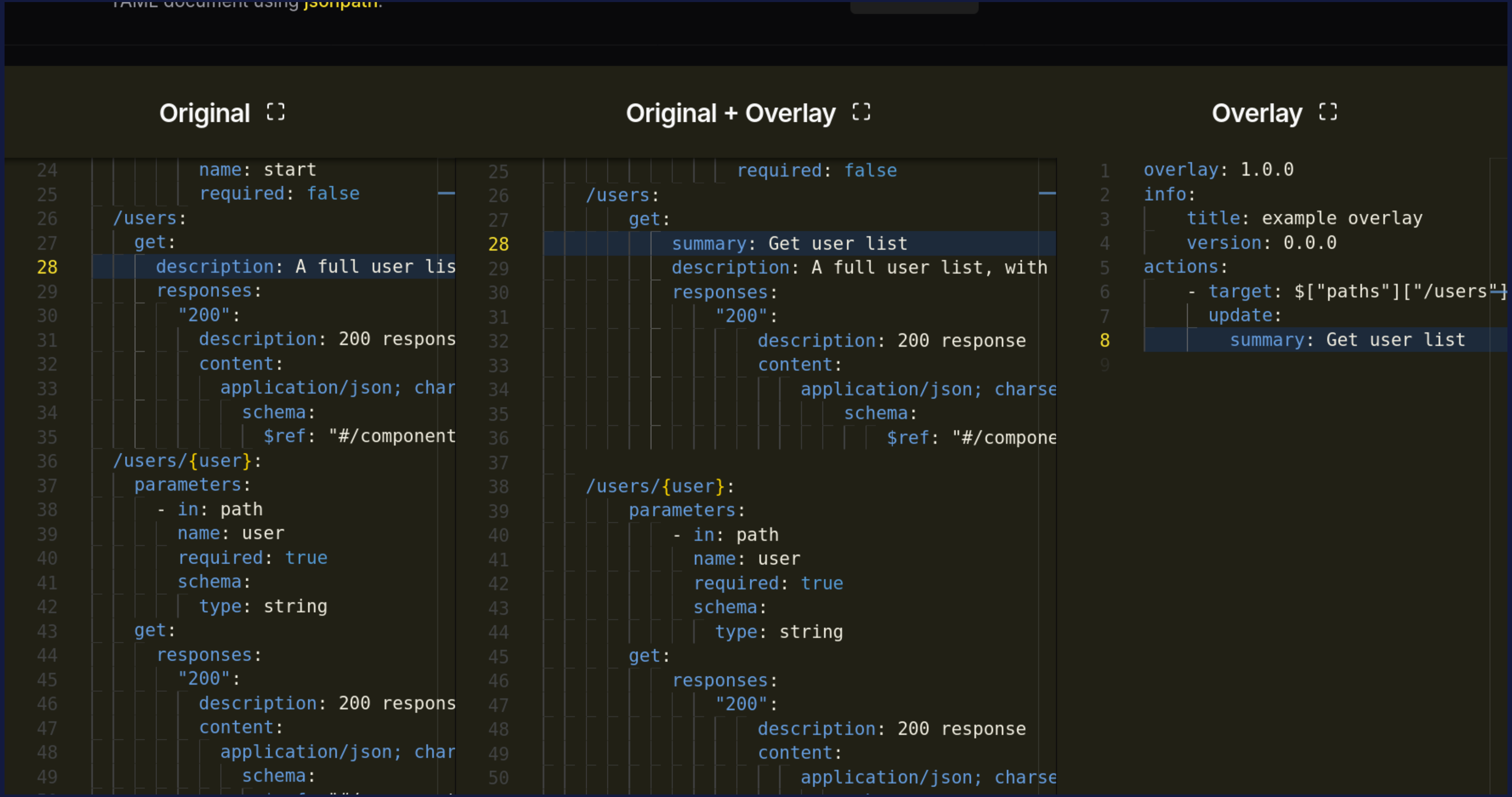
Part 1 is to prepare the Overlay:

- Copy the the OpenAPI file, and edit the copy
- Use `speakeasy compare` creates a diff as an Overlay
- Web GUI: https://overlay.speakeasy.com

Part 2 is to apply it in every build, locally and in CI

- Try the `bump` or `speakeasy` CLI tools.

# Repeatably edit an OpenAPI



YAML document using jsonpath.

**Original**

```yaml
        name: start
        required: false
  /users:
    get:
      description: A full user lis
      responses:
        "200":
          description: 200 respons
          content:
            application/json; char
              schema:
                $ref: "#/component
  /users/{user}:
    parameters:
      - in: path
        name: user
        required: true
        schema:
          type: string
    get:
      responses:
        "200":
          description: 200 respons
          content:
            application/json; char
              schema:
```

**Original + Overlay**

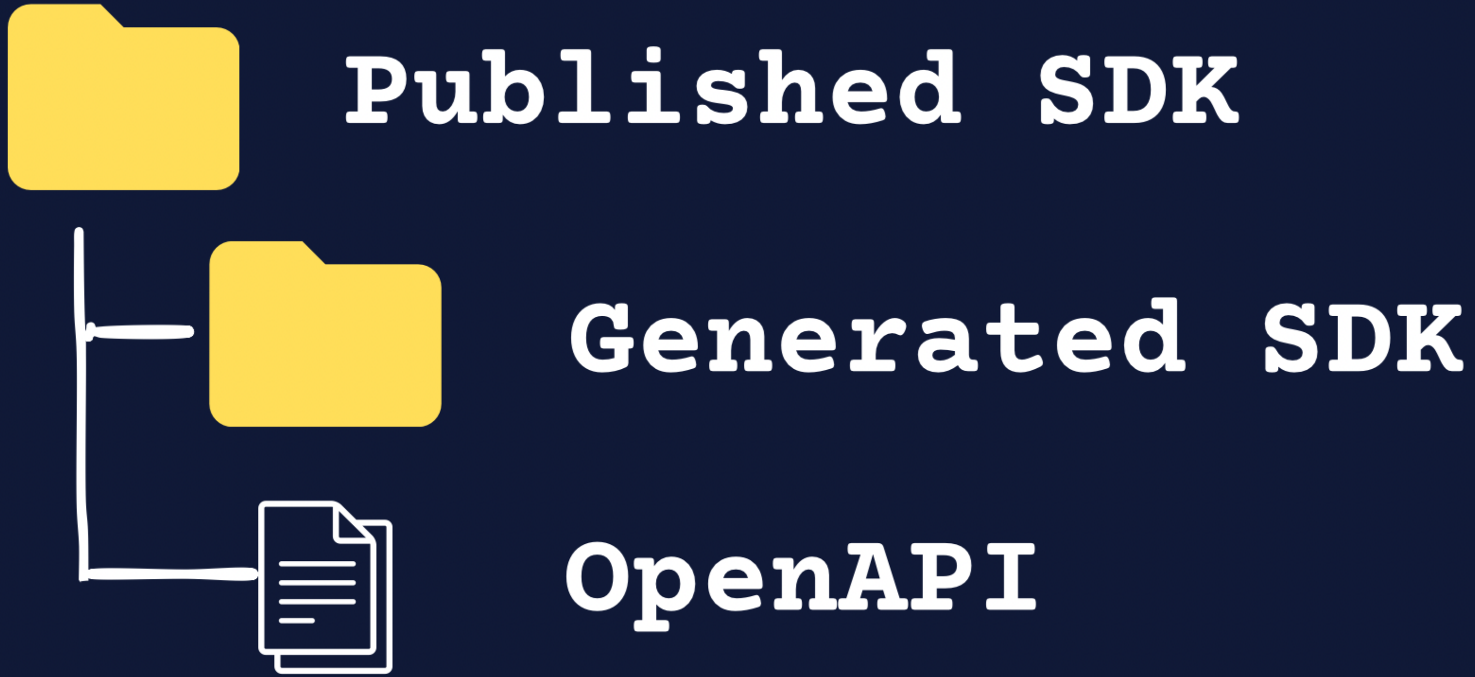
```yaml
                  required: false
    /users:
        get:
            summary: Get user list
            description: A full user list, with
            responses:
                "200":
                    description: 200 response
                    content:
                        application/json; charse
                            schema:
                                $ref: "#/compone

    /users/{user}:
        parameters:
            - in: path
              name: user
              required: true
              schema:
                type: string
        get:
            responses:
                "200":
                    description: 200 response
                    content:
                        application/json; charse
```

**Overlay**

```yaml
overlay: 1.0.0
info:
    title: example overlay
    version: 0.0.0
actions:
    - target: $["paths"]["/users"]
      update:
        summary: Get user list
```

## If you have code using your API



## If you like API quality and consistency

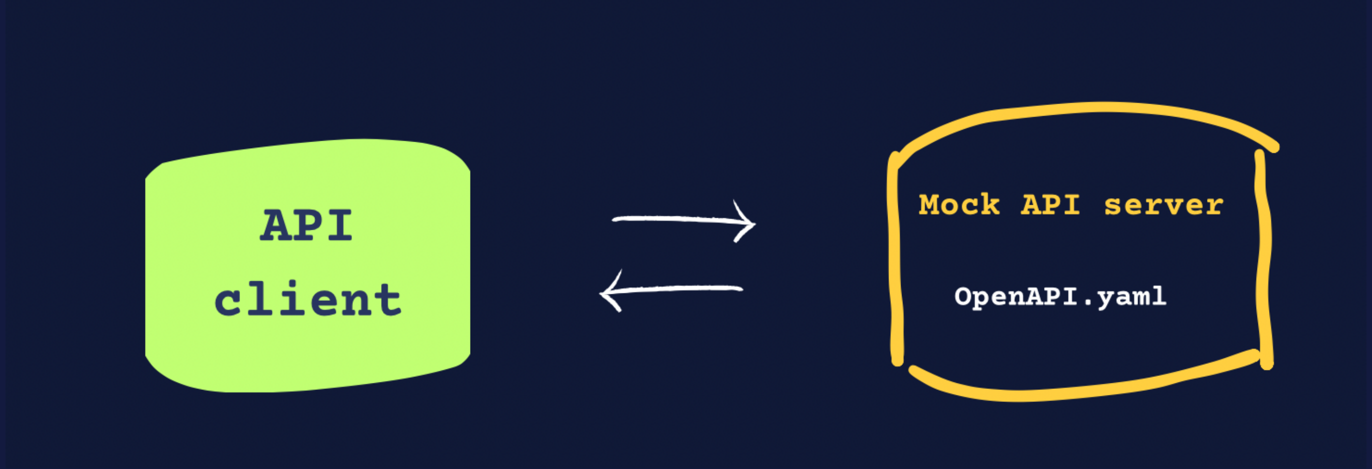
API standards are a document. Use linting to check:

- plurals and casing
- every operation has an ID
- everything named `*-date` uses the expected format
- errors are defined and in the expected format

Try: Spectral or Vacuum.

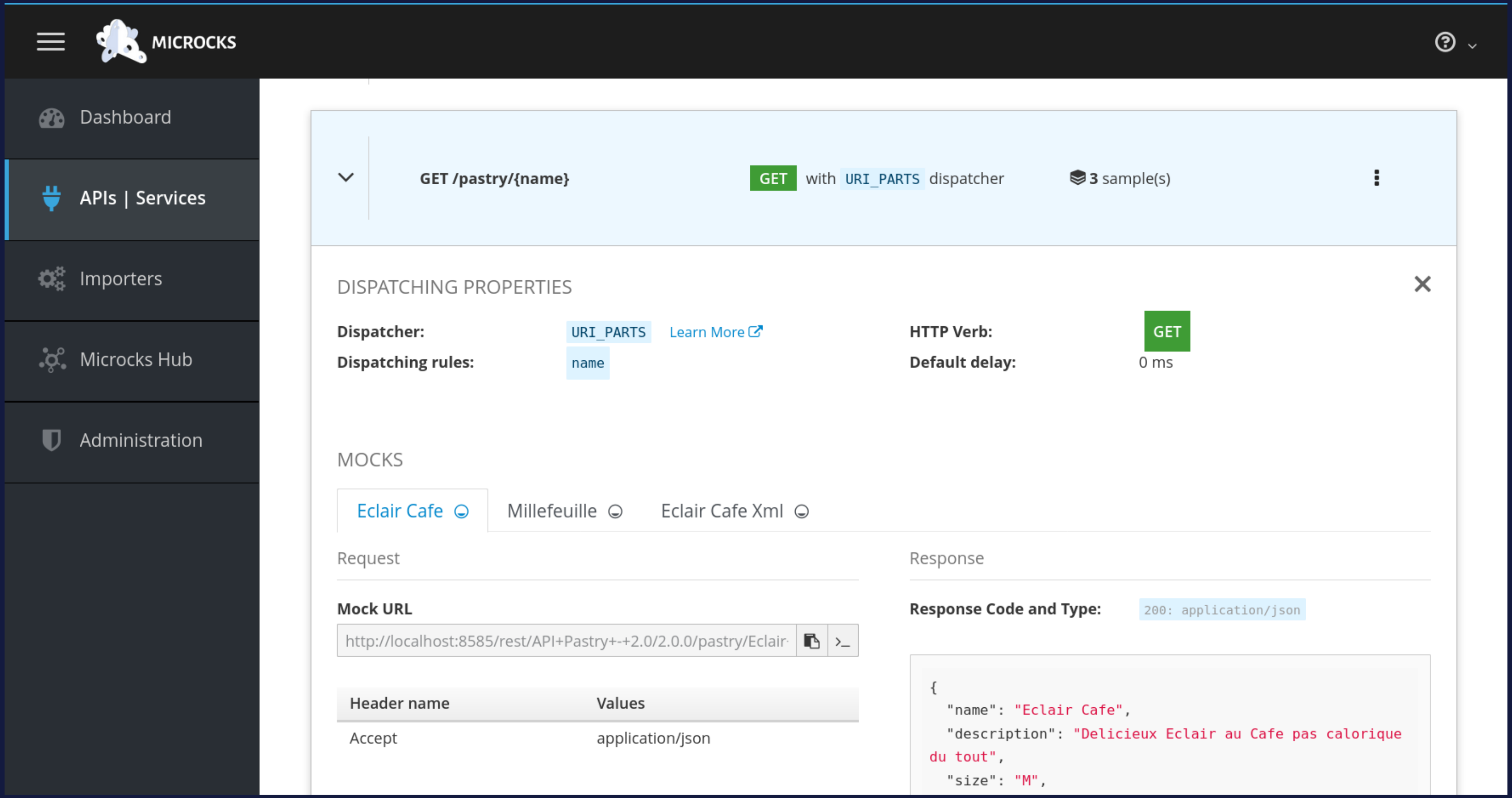
## If an API sandbox would be useful

Use an API Mock Server tool.



Tools include: Microcks, Prism

# If an API sandbox would be useful



GET /pastry/{name} GET with URI_PARTS dispatcher 3 sample(s)

DISPATCHING PROPERTIES

| | | | |
|---|---|---|---|
| Dispatcher: | URI_PARTS Learn More | HTTP Verb: | GET |
| Dispatching rules: | name | Default delay: | 0 ms |

MOCKS

Eclair Cafe | Millefeuille | Eclair Cafe Xml

Request

Mock URL

http://localhost:8585/rest/API+Pastry+-+2.0/2.0.0/pastry/Eclair

| Header name | Values |
|---|---|
| Accept | application/json |

Response

Response Code and Type: 200: application/json

```
{
  "name": "Eclair Cafe",
  "description": "Delicieux Eclair au Cafe pas calorique
du tout",
  "size": "M",
```

@lornajane@indieweb.social

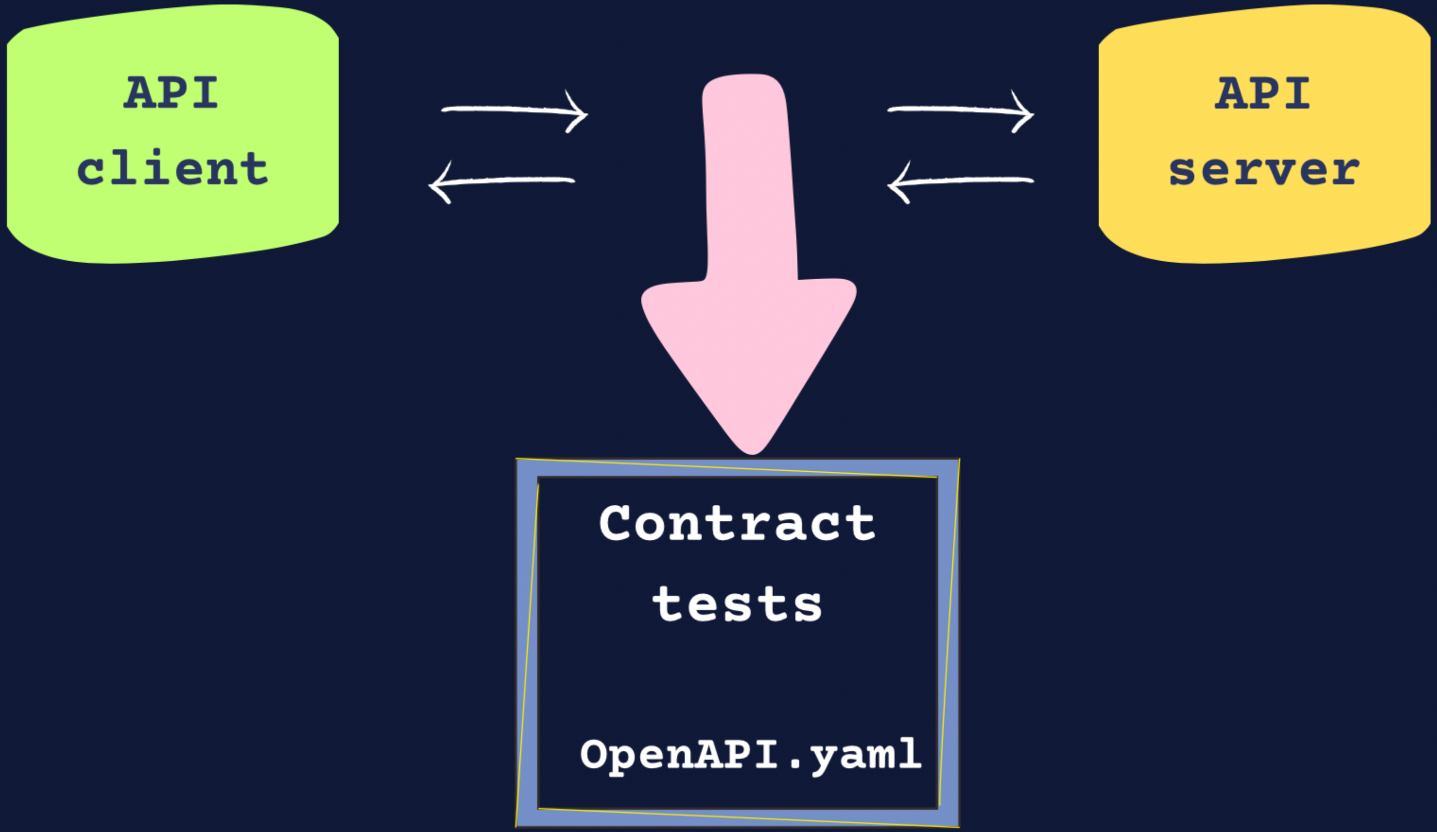
## To avoid API drift

Contract testing is your friend!

Check that what the API does matches what is described.

Tools include: Microcks, WireMock, Pact

## To avoid API drift



## API Description Pipelines

OpenAPI isn't a static asset.

## OpenAPI Community

# OpenAPI Community

- Part of the Linux Foundation
- Standard is developed in the open
- https://openapis.org
- Public GitHub repository
- Active Slack groups
- Weekly technical meetings

# OpenAPI Standards

- Overlays map repeatable amendments to an OpenAPI file (v1.0.0)
- Arazzo describe a sequence of API calls (v1.0.1)
- OpenAPI describes your API for all the tools to use
    - stable: 3.1.1
    - planned: 3.2.0
    - early days: 4.0 "Moonwalk"

## OpenAPI

Open standard for API descriptions

# Resources

- https://lornajane.net
- https://openapis.org
- https://github.com/opticdev/optic
- https://speakeasy.com
- https://bump.sh
- https://microcks.io/
- https://apisyouwonthate.com/