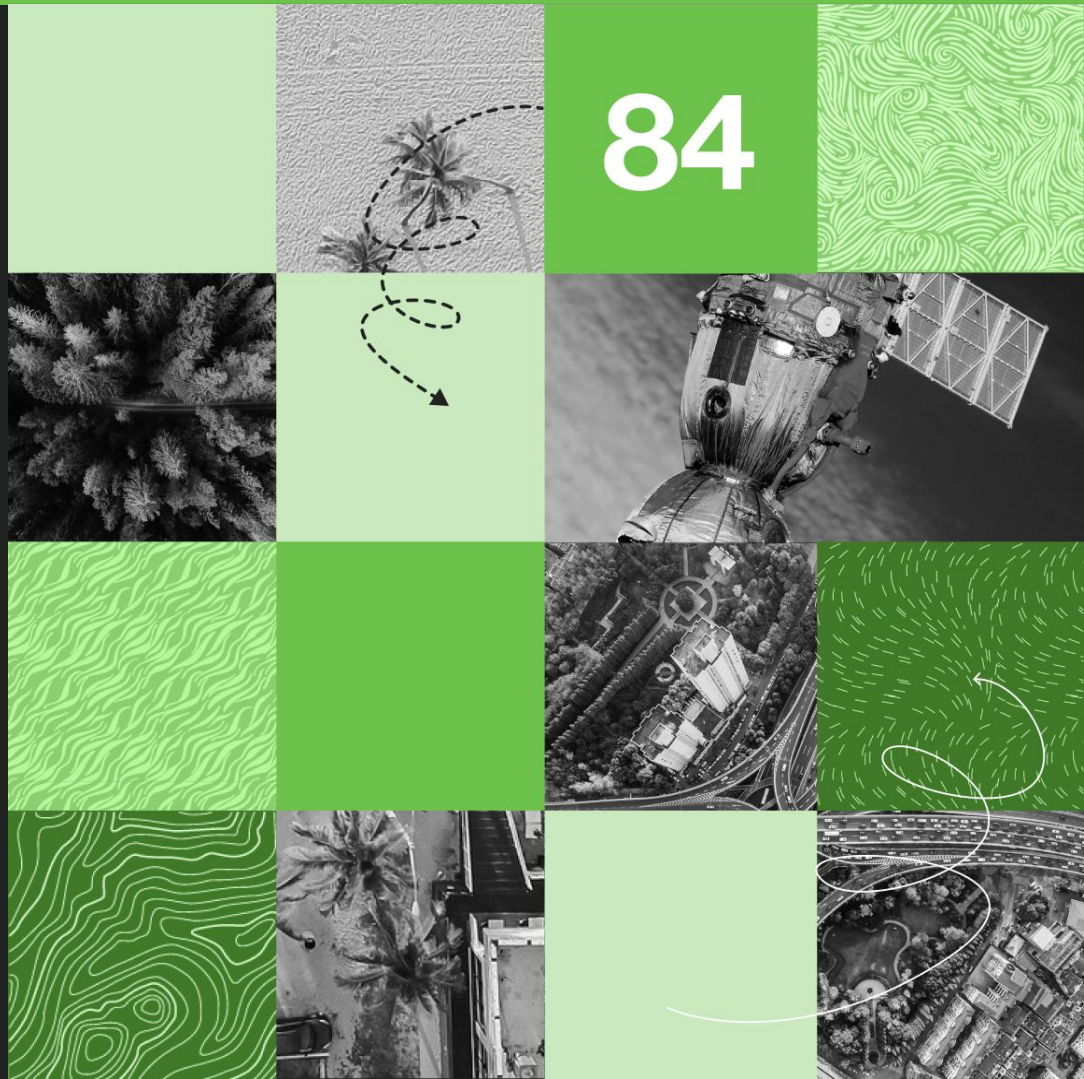
Satellite Tasking

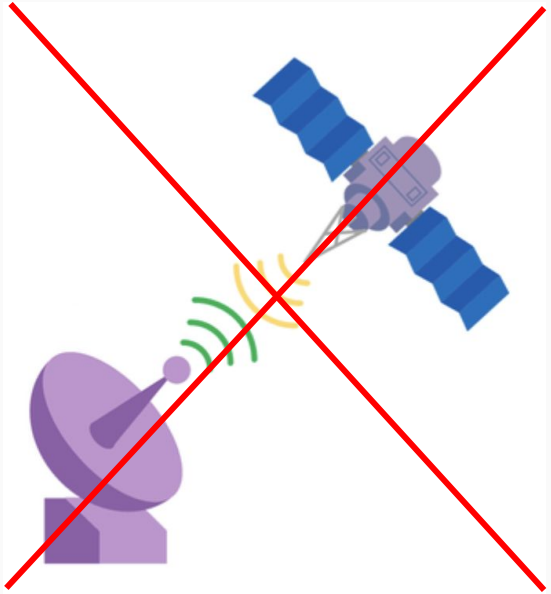
STAPI - A Tasking API for satellite interoperability

Matthew Hanson
@geoskeptic.bsky.social

Element 84

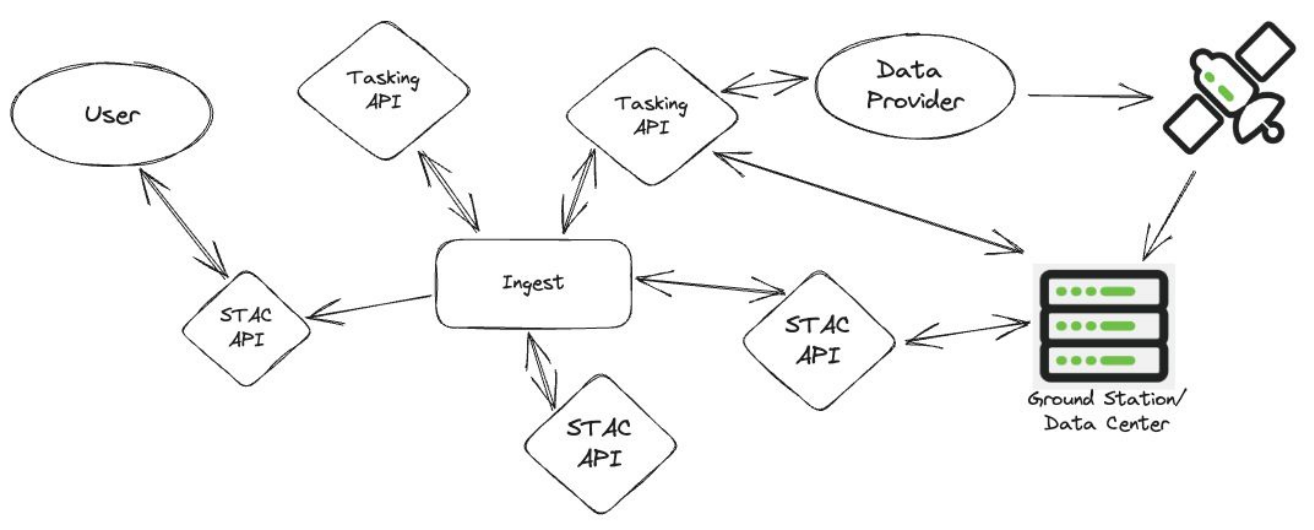


What is a Tasking API?

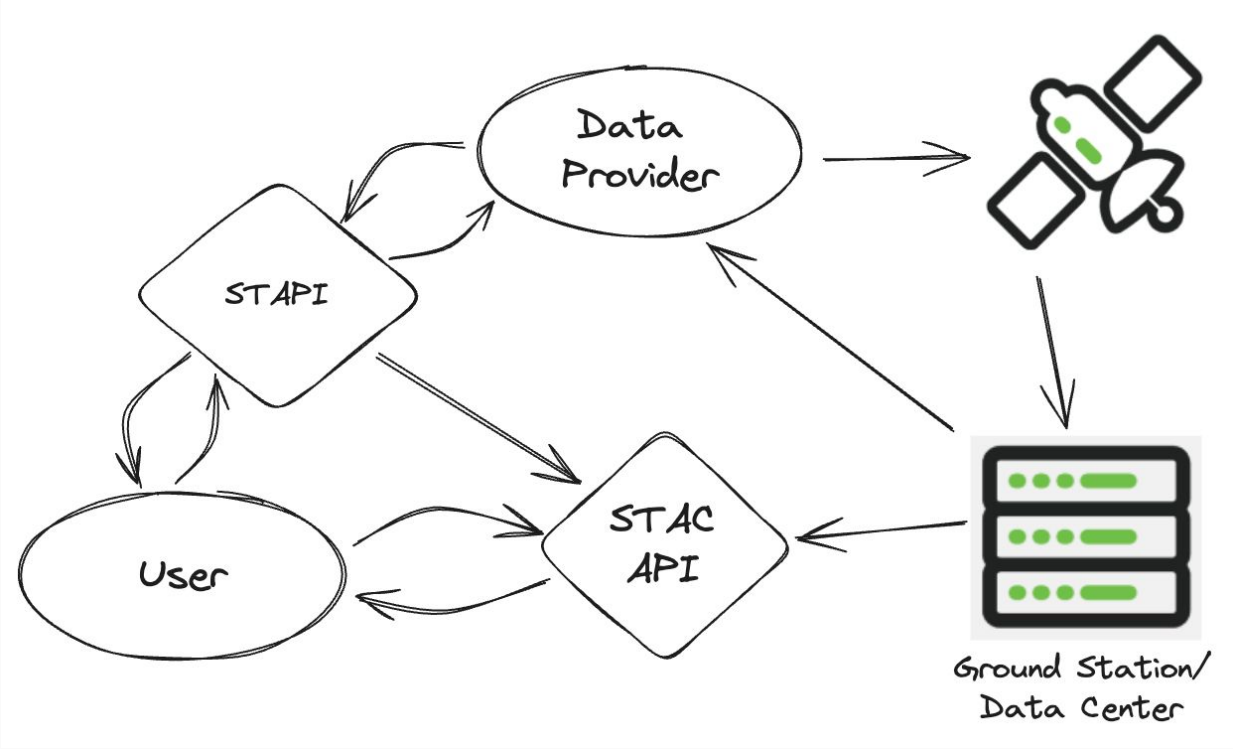


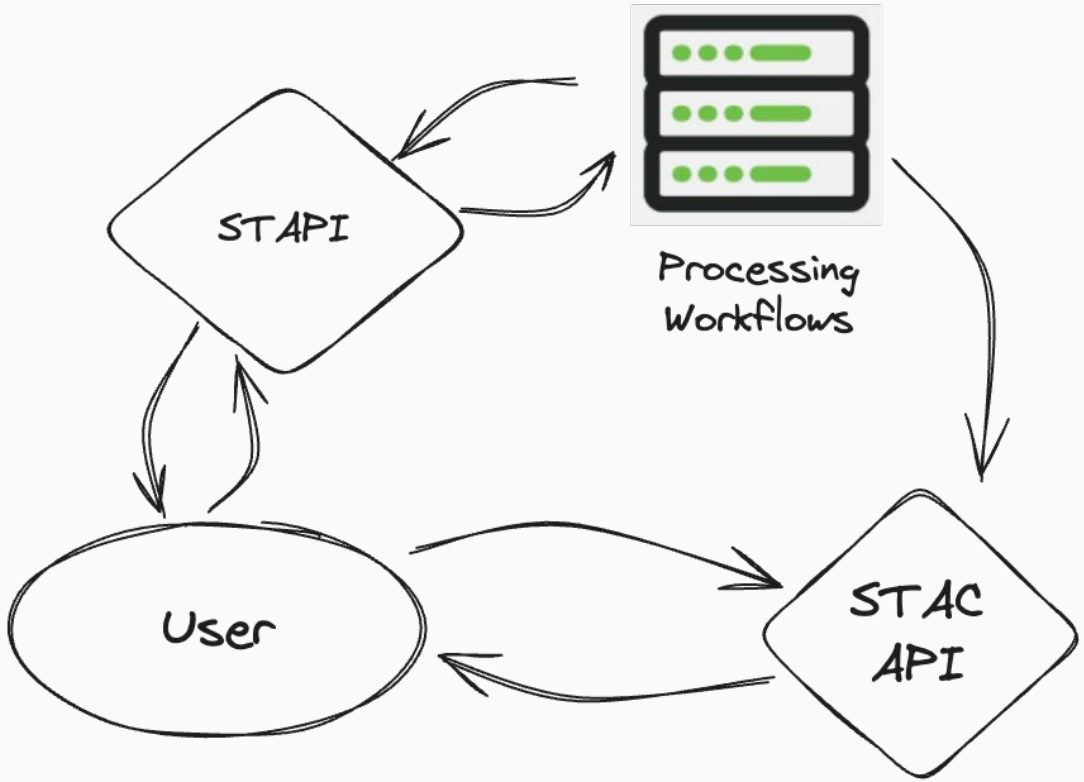

Submarine
Base Midway
Northwestern
Hawaiian Islands

Empower Users




STAC
TO THE FUTURE
2020


Data Provider
STAPI
User
STAC API
Ground Station/ Data Center


STAPI
Processing
Workflows
User
STAC
API

The Story so far

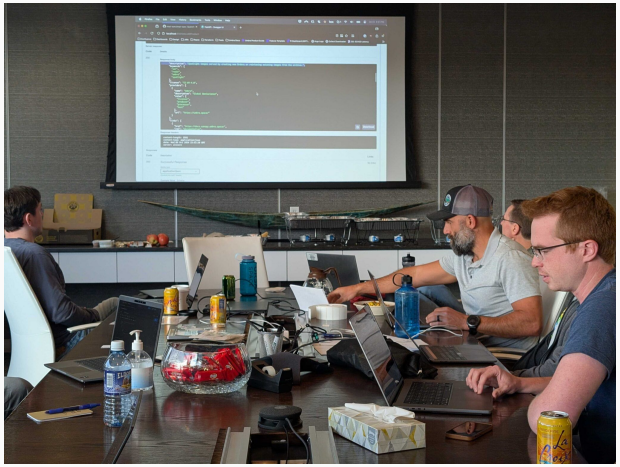
2022 - 1 Day in Alexandria, VA

2023 - 2 Days in Philadelphia, PA


2024 TASKING SPRINT
BERLIN, GERMANY


up
A new planeta
perspective.


2024 STAPI SPRINT
BROOMFIELD, COLORADO




Element 84
OPEN COSMOS
ALBEDO
planet.
aws
UP 42
OSK
geo
SKYFI
LiveEO
COGNITIVE SPACE
HYDROSAT
SKYWATCH
EUSI
EUROPEAN SPACE IMAGING
CARBON MAPPER
esri
Microsoft
SatVu
EOI Space
UMBRA
URSA SPACE
SATELLOGIC
NV5
Capella Space
Open Geospatial Consortium
bolts
innovative research
ORBIT LOGIC
BLACK SKY

Satellite Tasking API (STAPI) Specification

README.md

Welcome to the Sensor Tasking API Specification!

This organization contains repositories for the STAPI specification and the open-source tooling for creating and consuming APIs.

Pinned

Customize pins

stapi-spec Public

SpatioTemporal Asset Tasking API Spec

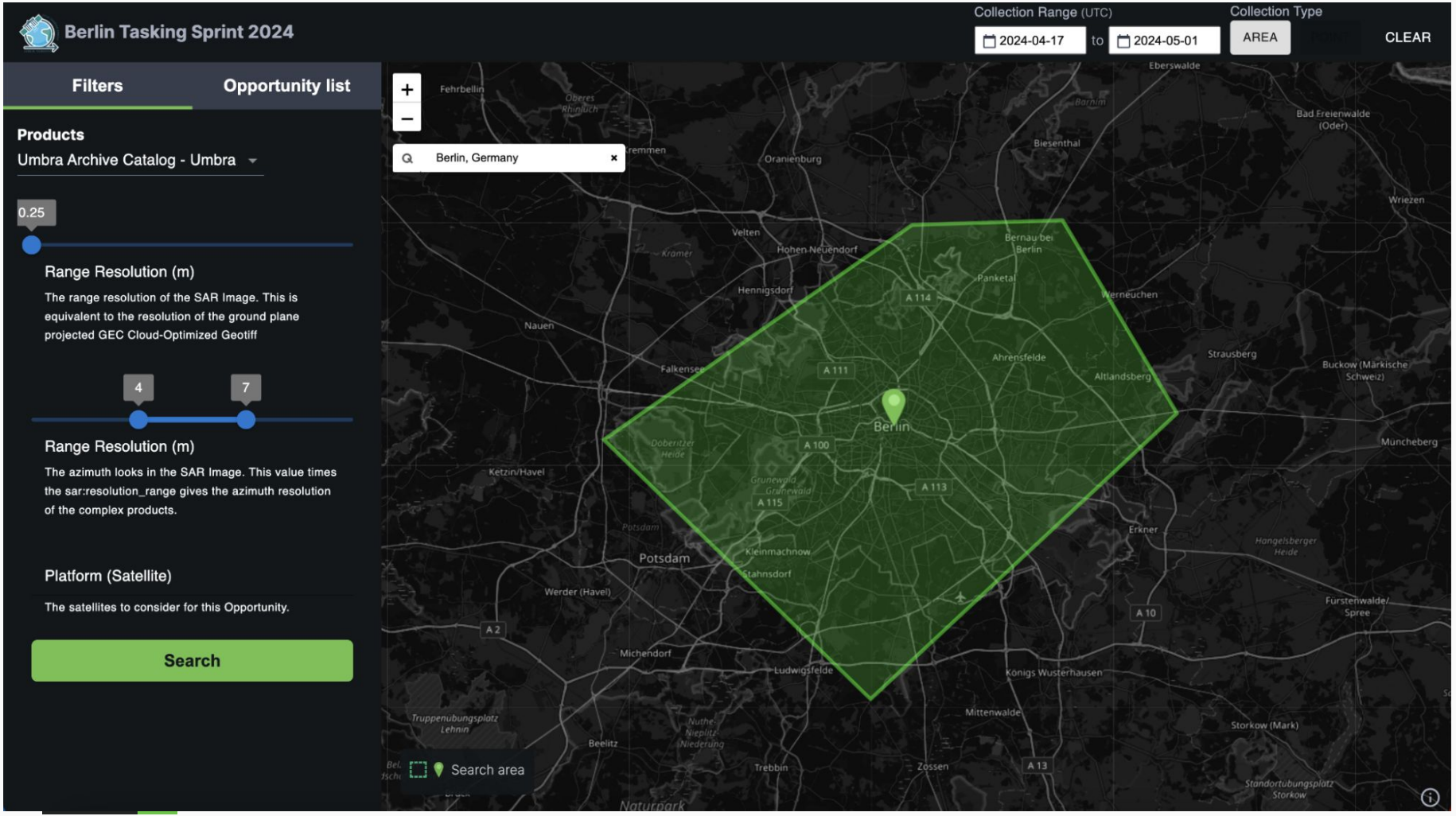
50 10

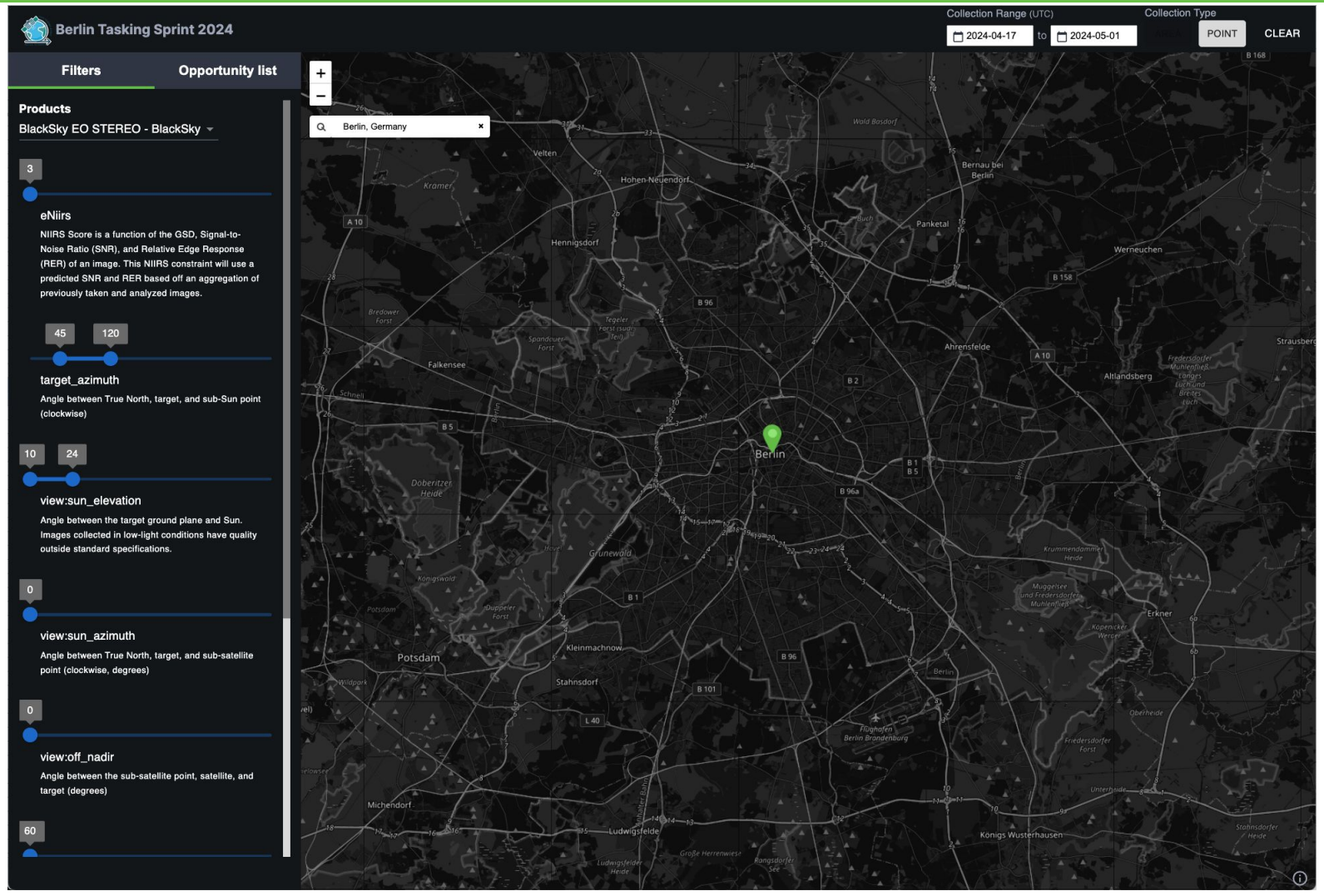
stapi-fastapi Public

Spatio Temporal Asset Tasking with FastAPI

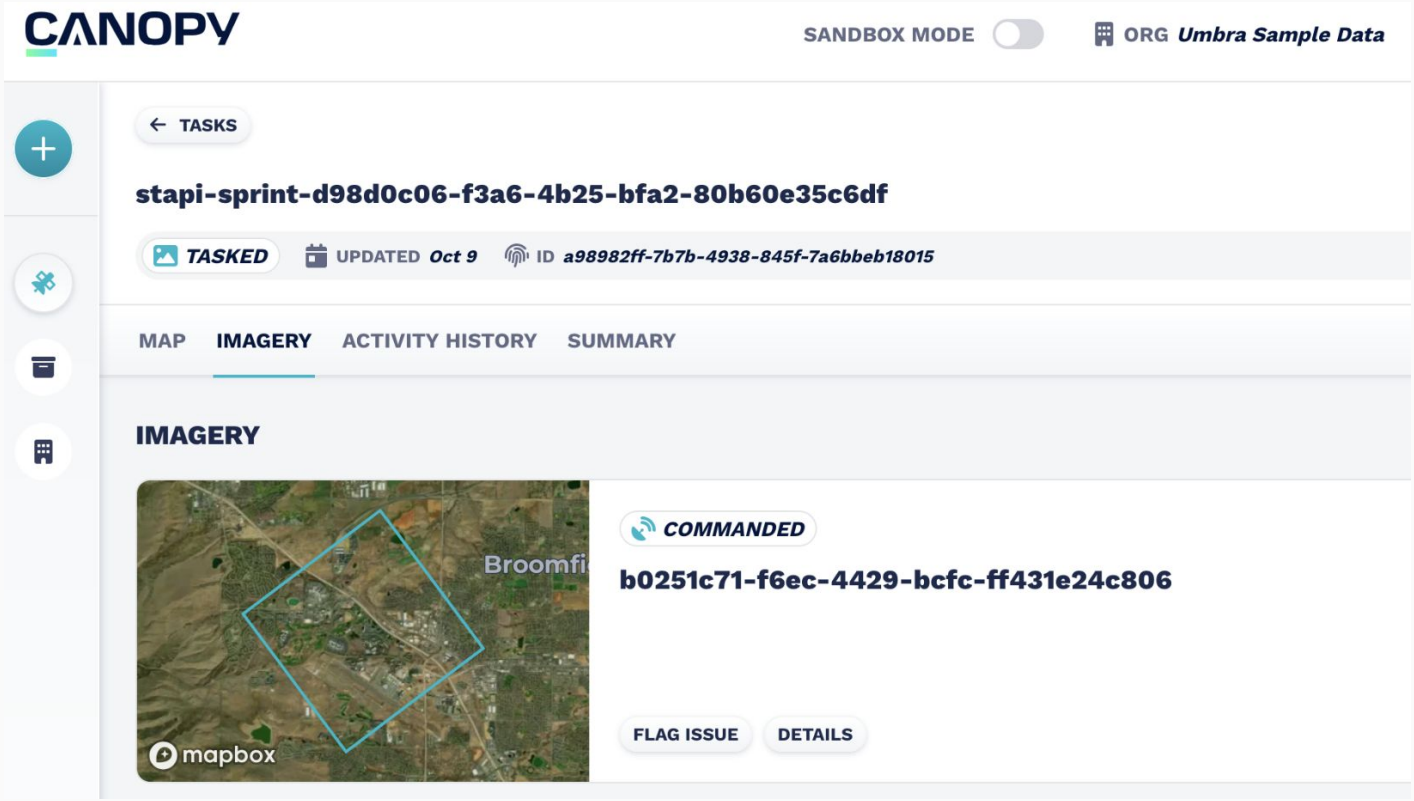
Python 14 5

https://github.com/stapi-spec


Berlin Tasking Sprint 2024
Collection Range (UTC)
2024-04-17
to
2024-05-01
Collection Type
AREA
CLEAR
Filters
Opportunity list
Products
Umbra Archive Catalog - Umbra
0.25
Range Resolution (m)
The range resolution of the SAR Image. This is equivalent to the resolution of the ground plane projected GEC Cloud-Optimized Geotiff
4
7
Range Resolution (m)
The azimuth looks in the SAR Image. This value times the sar:resolution_range gives the azimuth resolution of the complex products.
Platform (Satellite)
The satellites to consider for this Opportunity.
Search
Berlin, Germany
Search area
Berlin
Potsdam
Oranienburg
Hohen Neuendorf
Bernau bei Berlin
Panketal
Hennigsdorf
Velten
Nauen
Falkensee
Ahrensfelde
Altlandsberg
Strausberg
Werneuchen
Biesenthal
Eberswalde
Bad Freienwalde (Oder)
Wriezen
Buckow (Märkische Schweiz)
Müncheberg
Erkner
Fürstenwalde/ Spree
Königs Wusterhausen
Mittenwalde
Storkow (Mark)
Ludwigsfelde
Michendorf
Werder (Havel)
Kleinmachnow
Stahnsdorf
Ketzin/Havel
Beelitz
Trebbin
Zossen
A 10
A 13
A 100
A 111
A 113
A 114
A 115
A 2


Berlin Tasking Sprint 2024
Collection Range (UTC)
2024-04-17
to
2024-05-01
Collection Type
POINT
CLEAR
Filters
Opportunity list
Products
BlackSky EO STEREO - BlackSky
3
eNiirs
NIIRS Score is a function of the GSD, Signal-to-Noise Ratio (SNR), and Relative Edge Response (RER) of an image. This NIIRS constraint will use a predicted SNR and RER based off an aggregation of previously taken and analyzed images.
45
120
target_azimuth
Angle between True North, target, and sub-Sun point (clockwise)
10
24
view:sun_elevation
Angle between the target ground plane and Sun. Images collected in low-light conditions have quality outside standard specifications.
0
view:sun_azimuth
Angle between True North, target, and sub-satellite point (clockwise, degrees)
0
view:off_nadir
Angle between the sub-satellite point, satellite, and target (degrees)
60
Berlin, Germany
Berlin
Potsdam
Falkensee
Hennigsdorf
Velten
Hohen Neuendorf
Bernau bei Berlin
Panketal
Werneuchen
Ahrensfelde
Altlandsberg
Erkner
Kleinmachnow
Stahnsdorf
Michendorf
Ludwigsfelde
Königs Wusterhausen
Grunewald
Tegeler Forst (südl. Teil)
Spandauer Forst
Bredower Forst
Döberitzer Heide
Potsdamer Forst
Königswald
Wildpark
Flughafen Berlin Brandenburg
Friedersdorfer Forst
Oberheide
Unterheide
Krummendammer Heide
Muggelsee und Fredersdorfer Mühlenfließ
Kopenicker Werder
Große Herrenwiese
Rangsdorfer See
Ludwigsfelder Heide
Stahnsdorfer Heide
A 10
B 96
B 158
B 2
B 5
B 1
B 96a
B 101
L 40
B 168

STAPI used to order!



The Specification

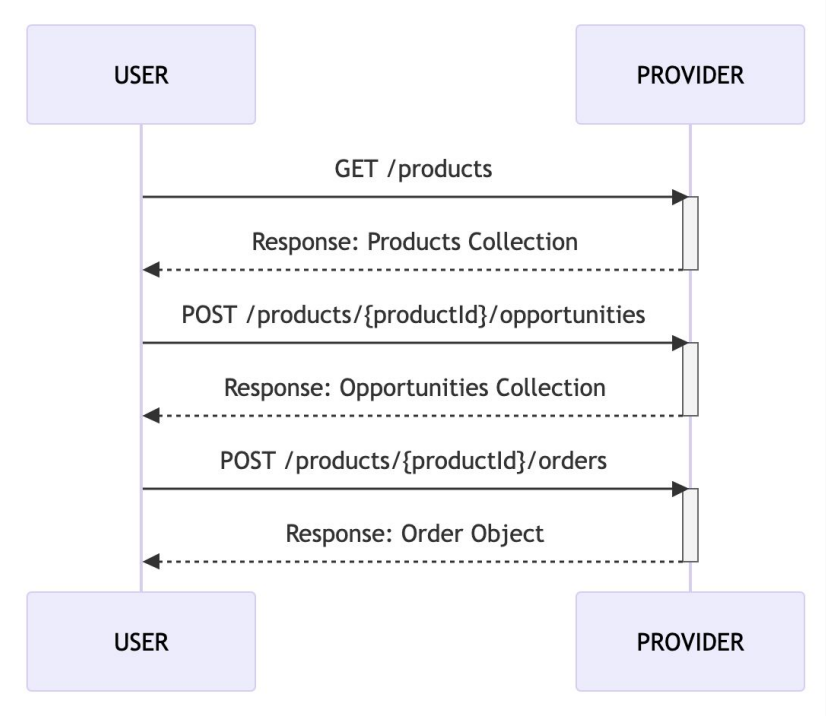
Entities

- Product
  - Description of the data to be collected (assets)
  - Constraints (i.e., CQL2 queryables)
  - Order Parameters (non-data order settings)
  - Opportunity Properties (a range or enumeration of potential values)
- Opportunity
  - For a single product
  - Description of geospatial data that may be collected in the future
  - GeoJSON describing collection location
  - A range of dates and a product.
  - Optionally include potential values or ranges for arbitrary fields
- Order
  - For a single product
  - State can change until fulfilled or window of opportunity has passed

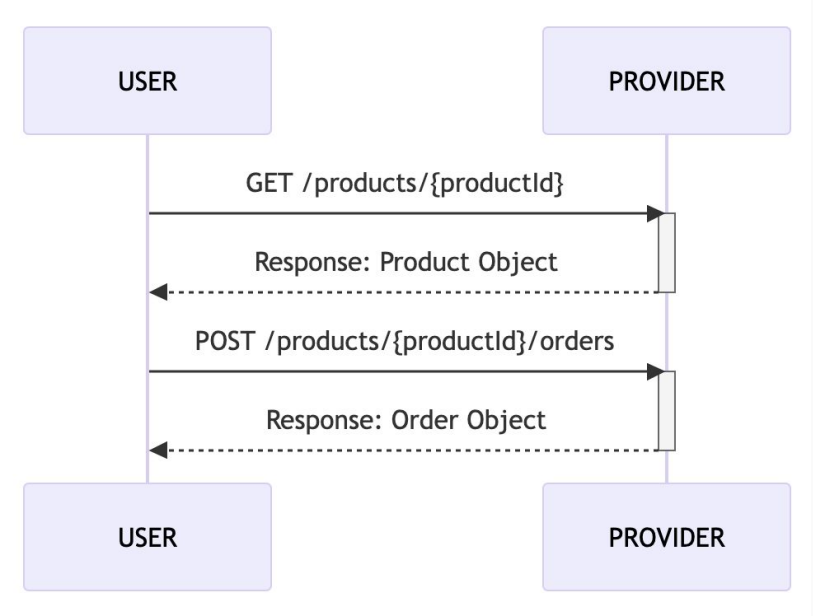
Endpoints

Endpoint	Relation Type
`GET /conformance`	`conformance`
`GET /products`	`products`
`GET /products/{productId}`	`product`
`GET /products/{productId}/constraints`	`constraints`
`GET /products/{productId}/order-parameters`	`order-parameters`
`POST /products/{productId}/orders`	`create-order`
`GET /products/{productId}/orders`	`orders`
`POST /products/{productId}/opportunities`	`opportunities`
`GET /orders`	`orders`
`GET /orders/{orderId}`	`order`
`GET /orders/{orderId}/status`	`monitor`

User Workflows



View Opportunities then order



Go straight to ordering

Product

Element	Type	Description
type	string	**REQUIRED.** Must be set to `Product` to be a valid Product.
id	string	**REQUIRED.** Identifier for the Product that is unique across the provider.
conformsTo	[string]	Conformance classes that apply to the product specifically.
title	string	A short descriptive one-line title for the Product.
description	string	**REQUIRED.** Detailed multi-line description to fully explain the Collection. CommonMark 0.29 syntax MAY be used for rich text representation.
keywords	[string]	List of keywords describing the Product.
license	string	**REQUIRED.** Collection's license(s), either a SPDX License identifier, `various` if multiple licenses apply or `proprietary` for all other cases.
providers	[Provider Object]	A list of providers, which may include all organizations capturing or processing the data or the hosting provider. Providers should be listed in chronological order with the most recent provider being the last element of the list.
links	[Link Object]	**REQUIRED.** A list of references to other documents.

Opportunity Request

for `POST /products/{productId}/opportunities`

Field Name	Type	Description
datetime	string	**REQUIRED.** Time interval with a solidus (forward slash, `/`) separator, using RFC 3339 datetime, empty string, or `..` values.
productId	string	**REQUIRED.** Product identifier. The ID should be unique and is a reference to the parameters which can be used in the parameters field.
geometry	GeoJSON Geometry Object	**REQUIRED.** Defines the full footprint of the asset represented by this item, formatted according to RFC 7946, section 3.1. The footprint should be the default GeoJSON geometry, though additional geometries can be included. Coordinates are specified in Longitude/Latitude or Longitude/Latitude/Elevation based on WGS 84.
filter	CQL2 Object	A set of add

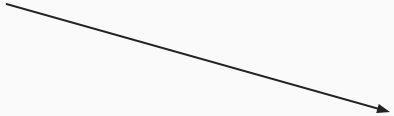
Create Order Request

The following fields can be sent to `POST /products/{productId}/orders`:

Field Name	Type	Description
datetime	string	**REQUIRED.** Time interval with a solidus (forward slash, `/`) separator, using RFC 3339 datetime, empty string, or `..` values.
productId	string	**REQUIRED.** Product identifier. The ID should be unique and is a reference to the parameters which can be used in the parameters field.
geometry	GeoJSON Object | JSON Reference	**REQUIRED.** Provide a Geometry that the tasked data must be within.
filter	CQL2 JSON	A set of additional parameters in CQL2 JSON based on the parameters exposed in the product.

Constraints

```
{
    "type": "application/schema+json",
    "rel": "constraints",
    "href": "/products/<productId>/constraints"
}
```




```
"description": "Umbra Spotlight constraints docstring",
"properties": {
    "sceneSize": {
        "allOf": [
            {
                "$ref": "#/parameters/$defs/SceneSize"
            }
        ],
        "default": "5x5_KM",
        "description": "The scene size of the Spotlight collect. The first "
    },
    "grazingAngleDegrees": {
        "type": "number",
        "minimum": 40,
        "maximum": 70,
        "description": "The minimum angle between the local tangent plane at the target location a
        "title": "Grazing Angle Degrees"
    },
    "satelliteIds": {
        "description": "The satellites to consider for this Opportunity.",
        "items": {
            "type": "string",
            "regex": "Umbra-\\d{2}"
        },
        "title": "Satelliteids",
        "type": "array"
    },
```

Constraints vs Order Parameters

- Constraints
  - Limitations on the collection of the data
  - Supplied to Opportunities and Order endpoints
  - Examples
    - View angles
    - Allowable cloud
- Order Parameters
  - Options that apply to delivery of the data
  - Supplied to just the Order endpoint
  - Examples
    - Data format
    - Destination bucket

Opportunity Object

This object describes a STAPI Opportunity. The input fields will be contained `properties` of each Feature in the GeoJSON response.

Field Name	Type	Description
type	string	**REQUIRED.** Type of the GeoJSON Object. MUST be set to `Feature`.
id	string	Provider identifier. This is not required, unless the provider tracks user requests and state for opportunities.
geometry	GeoJSON Geometry Object | null	**REQUIRED.** Defines the full footprint of the asset represented by this item, formatted according to RFC 7946, section 3.1. The footprint should be the default GeoJSON geometry, though additional geometries can be included. Coordinates are specified in Longitude/Latitude or Longitude/Latitude/Elevation based on WGS 84.
bbox	[number]	**REQUIRED if `geometry` is not `null`.** Bounding Box of the asset represented by this Item, formatted according to RFC 7946, section 5.
properties	Properties Object	**REQUIRED.** A dictionary of additional metadata for the Item.
links	[Link Object]	List of link objects to resources and related URLs. There **must** be a `rel=create-order` link that allows the user to Order this opportunity. See Link Object - rel=create-order below.

Order Object

Field Name	Type	Description
id	string	Unique provider generated order ID
user	string	User or organization ID ?
created	datetime	When the order was created
status	Order Status Object	Current Order Status object
links	[Link Object]	

If the `GET /orders/{orderId}/statuses` endpoint is implemented, there must be a link to the endpoint using the relation type `monitor`.

Order Status

Field Name	Type	Description
timestamp	datetime	**REQUIRED.** ISO 8601 timestamp for the order status
status_code	string	**REQUIRED.** Enumerated status code
reason_code	string	Enumerated reason code for why the status was set
reason_text	string	Textual description for why the status was set
links	[Link Object]	**REQUIRED.** list of references to documents, such as delivered asset, processing log, delivery manifest, etc.

The Ecosystem

STAPI Repos

STAPI GitHub org: https://github.com/orgs/stapi-spec

Core STAPI repos:

- stapi-spec
- stapi-fastapi
- stapi-client
- stapi-validator

Provider proxy implementations:

- stapi-fastapi-blacksky
- stapi-fastapi-planet
- stapi-fastapi-tle
- stapi-fastapi-up42
- stapi-fastapi-umbra

stapi-fastapi

stapi-fastapi

RootRouter

Endpoint	Relation Type
GET /conformance	conformance
GET /products	products
GET /products/{productId}	product
GET /products/{productId}/constraints	constraints
GET /products/{productId}/order-parameters	order-parameters
POST /products/{productId}/orders	create-order
GET /products/{productId}/orders	orders
POST /products/{productId}/opportunities	opportunities
GET /orders	orders
GET /orders/{orderId}	order
GET /orders/{orderId}/status	monitor

ProductRouter

stapi-fastapi

Want to use stapi-fastapi?

First define a product!

stapi-fastapi

```python
product_test_spotlight_sync_opportunity = Product(
    id="test-spotlight",
    title="Test Spotlight Product",
    description="Test product for test spotlight",
    license="CC-BY-4.0",
    keywords=["test", "satellite"],
    providers=[provider],
    links=[],
    create_order=mock_create_order,
    search_opportunities=mock_search_opportunities,
    search_opportunities_async=None,
    get_opportunity_collection=None,
    constraints=MyProductConstraints,
    opportunity_properties=MyOpportunityProperties,
    order_parameters=MyOrderParameters,
)
```

stapi-fastapi

Product Behaviors

```
product_test_spotlight_sync_opportunity = Product(
    id="test-spotlight",
    title="Test Spotlight Product",
    description="Test product for test spotlight",
    license="CC-BY-4.0",
    keywords=["test", "satellite"],
    providers=[provider],
    links=[],
  > create_order=mock_create_order,
  > search_opportunities=mock_search_opportunities,
  > search_opportunities_async=None,
  > get_opportunity_collection=None,
    constraints=MyProductConstraints,
    opportunity_properties=MyOpportunityProperties,
    order_parameters=MyOrderParameters,
)
```

stapi-fastapi

```
product_test_spotlight_sync_opportunity = Product(
    id="test-spotlight",
    title="Test Spotlight Product",
    description="Test product for test spotlight",
    license="CC-BY-4.0",
    keywords=["test", "satellite"],
    providers=[provider],
    links=[],
  > create_order=mock_create_order,
  > search_opportunities=mock_search_opportunities,
  > search_opportunities_async=None,
  > get_opportunity_collection=None,
    constraints=MyProductConstraints,
    opportunity_properties=MyOpportunityProperties,
    order_parameters=MyOrderParameters,
)
```

Product Behaviors

Product Models

stapi-fastapi

Single Value

```
class MyProductConstraints(BaseModel):
    off_nadir: int
```

Example Constraints

Example Opportunity Properties

```
class OffNadirRange(BaseModel):
    minimum: int = Field(ge=0, le=45)
    maximum: int = Field(ge=0, le=45)

    @model_validator(mode="after")
    def validate_range(self) -> Self:
        if self.minimum > self.maximum:
            raise ValueError("range minimum cannot be greater than maximum")
        return self


class MyOpportunityProperties(OpportunityProperties):
    off_nadir: OffNadirRange <- Range
```

stapi-fastapi

Now create a RootRouter and add your product!

stapi-fastapi

```
root_router = RootRouter(
    get_orders=mock_get_orders,
    get_order=mock_get_order,
    get_order_statuses=mock_get_order_statuses,
    get_opportunity_search_records=mock_get_opportunity_search_records,
    get_opportunity_search_record=mock_get_opportunity_search_record,
    conformances=[CORE, OPPORTUNITIES, ASYNC_OPPORTUNITIES],
)
root_router.add_product(product_test_spotlight_sync_opportunity)
app: FastAPI = FastAPI()
app.include_router(root_router)
```

stapi-fastapi

Root Router Behaviors

```
root_router = RootRouter(
 > get_orders=mock_get_orders,
 > get_order=mock_get_order,
 > get_order_statuses=mock_get_order_statuses,
 > get_opportunity_search_records=mock_get_opportunity_search_records,
 > get_opportunity_search_record=mock_get_opportunity_search_record,
   conformances=[CORE, OPPORTUNITIES, ASYNC_OPPORTUNITIES],
)
root_router.add_product(product_test_spotlight_sync_opportunity)
app: FastAPI = FastAPI()
app.include_router(root_router)
```

stapi-fastapi

Root Router Behaviors

```
root_router = RootRouter(
    get_orders=mock_get_orders,
    get_order=mock_get_order,
    get_order_statuses=mock_get_order_statuses,
    get_opportunity_search_records=mock_get_opportunity_search_records,
    get_opportunity_search_record=mock_get_opportunity_search_record,
    conformances=[CORE, OPPORTUNITIES, ASYNC_OPPORTUNITIES],
)
root_router.add_product(product_test_spotlight_sync_opportunity)
app: FastAPI = FastAPI()
app.include_router(root_router)
```

Advertised Conformance

stapi-fastapi

Root Router Behaviors

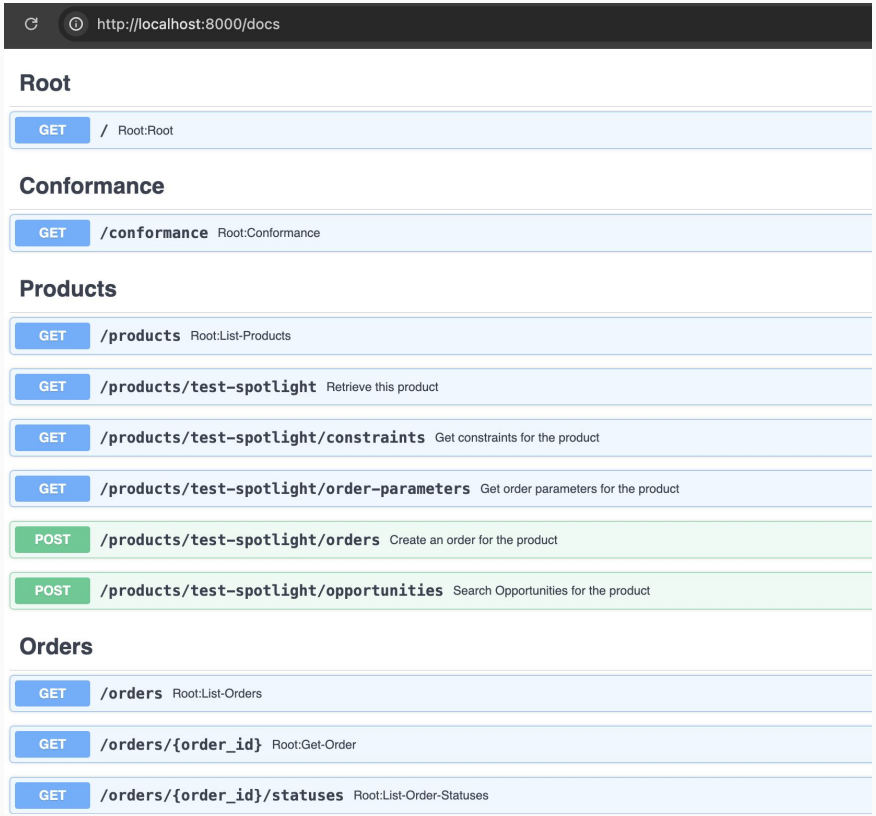
```
root_router = RootRouter(
    get_orders=mock_get_orders,
    get_order=mock_get_order,
    get_order_statuses=mock_get_order_statuses,
    get_opportunity_search_records=mock_get_opportunity_search_records,
    get_opportunity_search_record=mock_get_opportunity_search_record,
    conformances=[CORE, OPPORTUNITIES, ASYNC_OPPORTUNITIES],
)
root_router.add_product(product_test_spotlight_sync_opportunity)
app: FastAPI = FastAPI()
app.include_router(root_router)
```

Advertised Conformance

Adding our Product

And our Router to the App

stapi-fastapi



Lisbon Sprint Plan

Made possible by:

aws

developmentSEED

Spec Discussion Topics

- Core and Optional Extensions - Conformance Classes
- Archive ordering vs Tasking
  - https://github.com/stapi-spec/stapi-spec/issues/204
- Opportunities, tasks, collects, orders
  - https://github.com/stapi-spec/stapi-spec/issues/241
- Async Opportunities
  - https://github.com/stapi-spec/stapi-spec/pull/240
- CQL2 Filtering
  - https://github.com/stapi-spec/stapi-spec/issues/234

stapi-spec 0.1

- Update all examples
- Get stapi-spec up to date with implementation (stapi-fastapi)
- Resolve open PRs
- Update OpenAPI documentation
- Cut release of spec

Ecosystem

- Refactor stapi-fastapi to create multiple packages
  - pystapi
  - pystapi-fastapi
  - Pystapi-client
  - https://github.com/stapi-spec/stapi-fastapi/issues/127

- API implementations
  - Data provider proxies to existing tasking APIs
  - TLE backend for opportunities
  - Landsat and/or Sentinel-2 backend for end-to-end POC
  - API validation library

Thank you!

Matthew Hanson

Trevor Skaggs

Phil Varner

element84.com

Element 84

Learn More at
https://github.com/stapi-spec