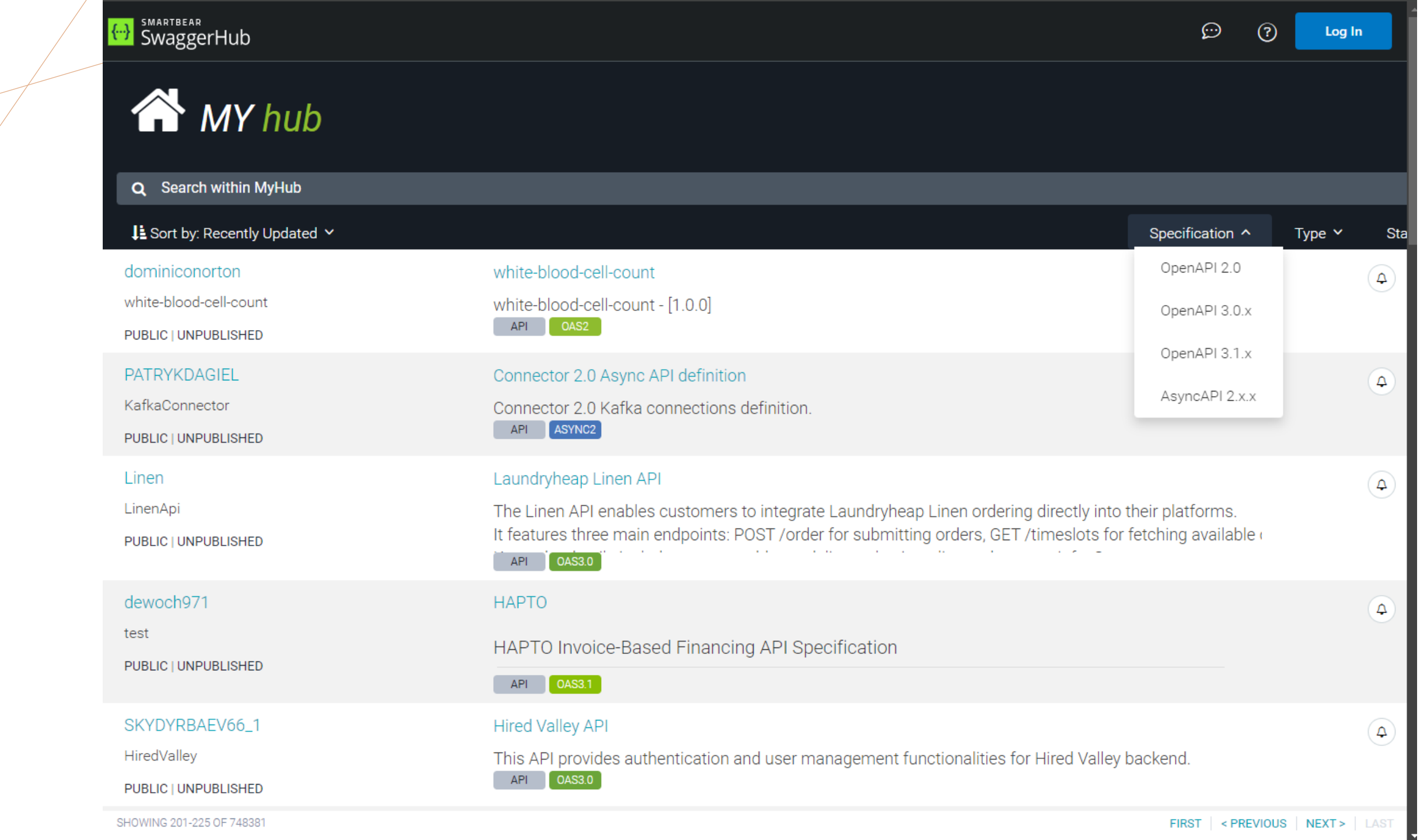
A DEEP DIVE INTO OPENAPI SECURITY

ANDREI AGAPE – OWASP COPENHAGEN 2024

AGENDA

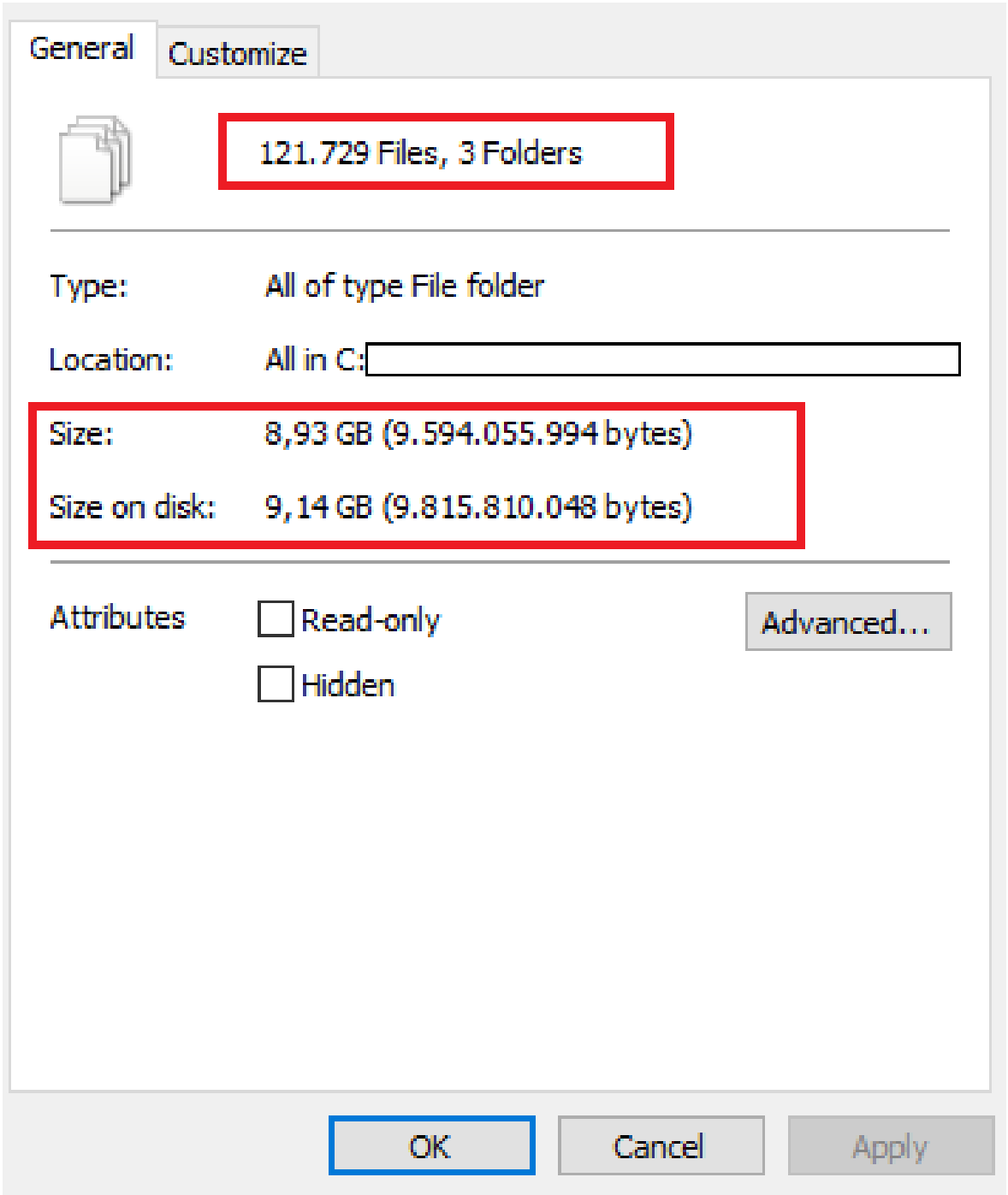
1. BACKGROUND
2. OPENAPI ANALYSIS
3. RESEARCH
4. TOOL
5. RESULTS
6. CONCLUSION

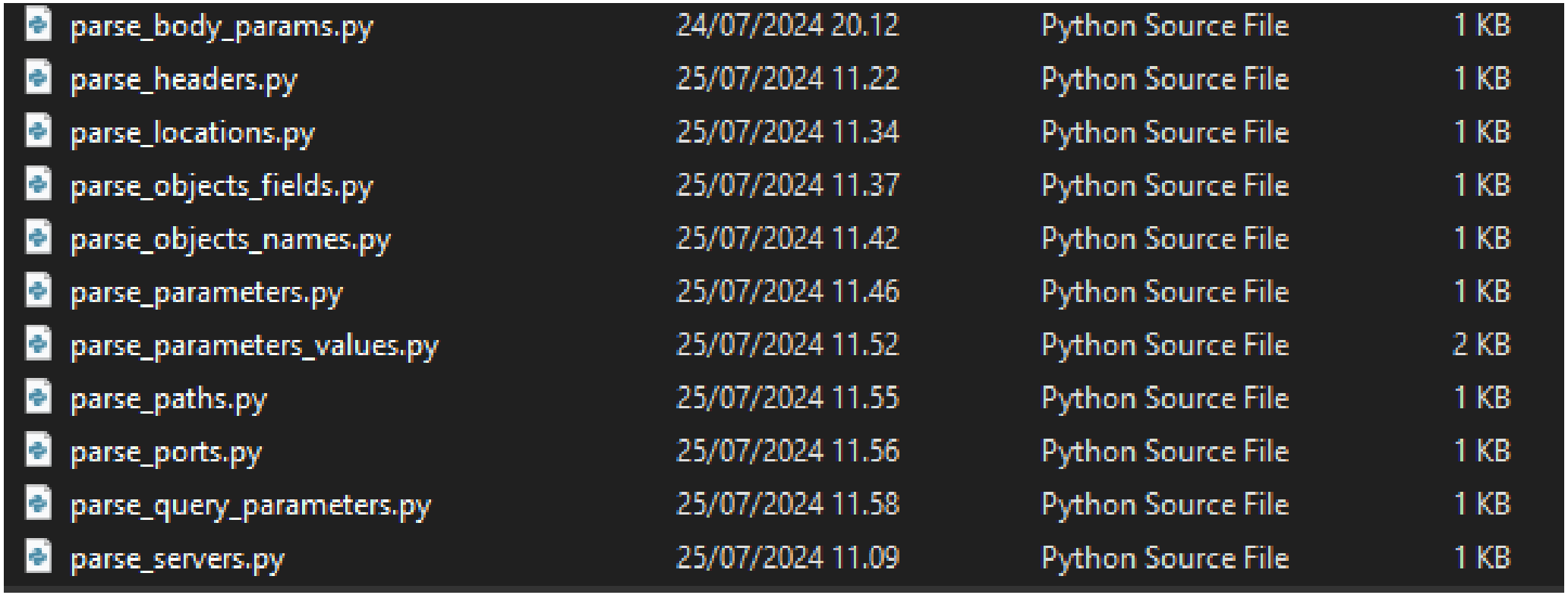
1. BACKGROUND


SMARTBEAR
SwaggerHub
Log In
MY hub
Search within MyHub
Sort by: Recently Updated
Specification
Type
Sta
OpenAPI 2.0
OpenAPI 3.0.x
OpenAPI 3.1.x
AsyncAPI 2.x.x
dominiconorton
white-blood-cell-count
PUBLIC | UNPUBLISHED
white-blood-cell-count
white-blood-cell-count - [1.0.0]
API
OAS2
PATRYKDAGIEL
KafkaConnector
PUBLIC | UNPUBLISHED
Connector 2.0 Async API definition
Connector 2.0 Kafka connections definition.
API
ASYNC2
Linen
LinenApi
PUBLIC | UNPUBLISHED
Laundryheap Linen API
The Linen API enables customers to integrate Laundryheap Linen ordering directly into their platforms.
It features three main endpoints: POST /order for submitting orders, GET /timeslots for fetching available
API
OAS3.0
dewoch971
test
PUBLIC | UNPUBLISHED
HAPTO
HAPTO Invoice-Based Financing API Specification
API
OAS3.1
SKYDYRBAEV66_1
HiredValley
PUBLIC | UNPUBLISHED
Hired Valley API
This API provides authentication and user management functionalities for Hired Valley backend.
API
OAS3.0
SHOWING 201-225 OF 748381
FIRST
< PREVIOUS
NEXT >
LAST

SHOWING 201-225 OF 748381

```
index = 0

for api_url in api_urls:
    try:
        index = index + 1
        print(api_url.strip())
        burp0_url = api_url.strip()
        burp0_cookies = {"authenticated": "false", "publicShared": "null", "liveChatShared":
        burp0_headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:128.0)
        response = requests.get(burp0_url, headers=burp0_headers, cookies=burp0_cookies)


        json_object = json.loads(response.content)

        with open("api_20/"+str(index)+'.json', 'w', encoding='utf-8') as f:
            json.dump(json_object, f, ensure_ascii=False, indent=4)
    except:
        print("ERROR: " + burp0_url)
```


General
Customize
121.729 Files, 3 Folders
Type:
All of type File folder
Location:
All in C:
Size:
8,93 GB (9.594.055.994 bytes)
Size on disk:
9,14 GB (9.815.810.048 bytes)
Attributes
Read-only
Hidden
Advanced...
OK
Cancel
Apply



parse_body_params.py	24/07/2024 20.12	Python Source File	1 KB
parse_headers.py	25/07/2024 11.22	Python Source File	1 KB
parse_locations.py	25/07/2024 11.34	Python Source File	1 KB
parse_objects_fields.py	25/07/2024 11.37	Python Source File	1 KB
parse_objects_names.py	25/07/2024 11.42	Python Source File	1 KB
parse_parameters.py	25/07/2024 11.46	Python Source File	1 KB
parse_parameters_values.py	25/07/2024 11.52	Python Source File	2 KB
parse_paths.py	25/07/2024 11.55	Python Source File	1 KB
parse_ports.py	25/07/2024 11.56	Python Source File	1 KB
parse_query_parameters.py	25/07/2024 11.58	Python Source File	1 KB
parse_servers.py	25/07/2024 11.09	Python Source File	1 KB

IT'S SOMETHING .. ¯_(ツ)_/¯

- 744,000+ endpoints
- 357,000+ object properties
- 211,000+ object names
- 127,000+ query parameters
- 74,000+ parameter values
- 35,000+ path parameters
- 8,300+ headers
- 5,300+ paths
- 880+ common ports

Her: he's probably thinking about other girls...

Him: There must be more in those Open APIs

CAN WE DO BETTER?

2. OPENAPI ANALYSIS

WHAT IS OPENAPI

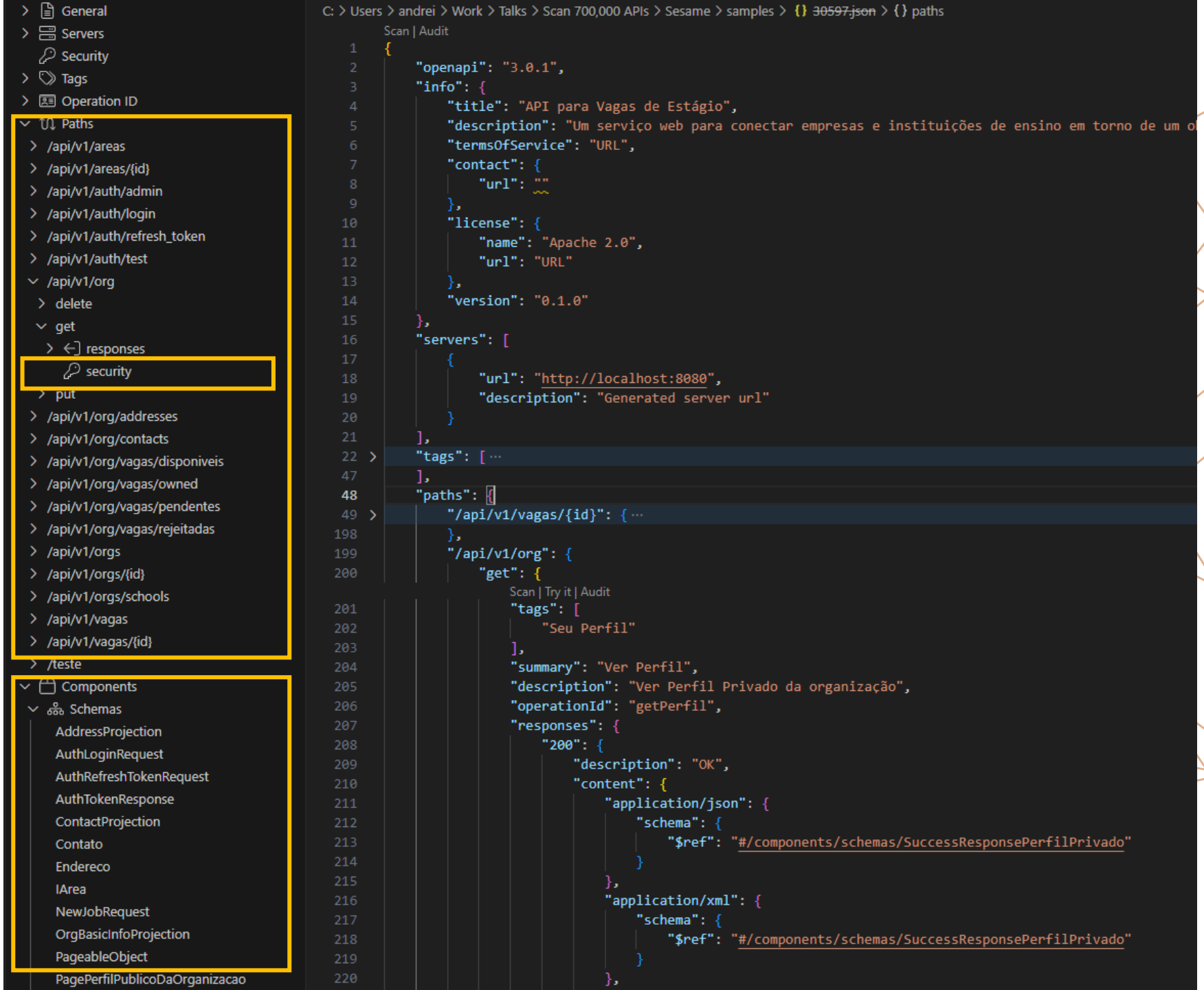
1. a “specification language” for REST APIs
2. describes the structure and syntax of the API
3. irrelevant of the programming language in which the API was created
4. Previously known as SwaggerFile (v2.x) it became OpenAPI (v3.x +)
5. has YAML or JSON format

WHY IS NEEDED

1. quickly discover how an API works
2. a standardized way to describe the API
3. no need to understand how the API is implemented
4. improves design/deployment/testing of API

Field Name	Type	Description
openapi	`string`	**REQUIRED**. This string MUST be the semantic version number of the OpenAPI Specification version that the OpenAPI document uses. The `openapi` field SHOULD be used by tooling specifications and clients to interpret the OpenAPI document. This is *not* related to the API `info.version` string.
info	Info Object	**REQUIRED**. Provides metadata about the API. The metadata MAY be used by tooling as required.
servers	[Server Object]	An array of Server Objects, which provide connectivity information to a target server. If the `servers` property is not provided, or is an empty array, the default value would be a Server Object with a url value of `/`.
paths	Paths Object	**REQUIRED**. The available paths and operations for the API.
components	Components Object	An element to hold various schemas for the specification.
security	[Security Requirement Object]	A declaration of which security mechanisms can be used across the API. The list of values includes alternative security requirement objects that can be used. Only one of the security requirement objects need to be satisfied to authorize a request. Individual operations can override this definition. To make security optional, an empty security requirement (`{}`) can be included in the array.

Field Name	Type	Description
openapi	`string`	**REQUIRED**. This string MUST be the semantic version number of the OpenAPI Specification version that the OpenAPI document uses. The `openapi` field SHOULD be used by tooling specifications and clients to interpret the OpenAPI document. This is *not* related to the API `info.version` string.
info	Info Object	**REQUIRED**. Provides metadata about the API. The metadata MAY be used by tooling as required.
servers	[Server Object]	An array of Server Objects, which provide connectivity information to a target server. If the `servers` property is not provided, or is an empty array, the default value would be a Server Object with a url value of `/`.
paths	Paths Object	**REQUIRED**. The available paths and operations for the API.
components	Components Object	An element to hold various schemas for the specification.
security	[Security Requirement Object]	A declaration of which security mechanisms can be used across the API. The list of values includes alternative security requirement objects that can be used. Only one of the security requirement objects need to be satisfied to authorize a request. Individual operations can override this definition. To make security optional, an empty security requirement (`{}`) can be included in the array.


General
Servers
Security
Tags
Operation ID
Paths
/api/v1/areas
/api/v1/areas/{id}
/api/v1/auth/admin
/api/v1/auth/login
/api/v1/auth/refresh_token
/api/v1/auth/test
/api/v1/org
delete
get
responses
security
put
/api/v1/org/addresses
/api/v1/org/contacts
/api/v1/org/vagas/disponiveis
/api/v1/org/vagas/owned
/api/v1/org/vagas/pendentes
/api/v1/org/vagas/rejeitadas
/api/v1/orgs
/api/v1/orgs/{id}
/api/v1/orgs/schools
/api/v1/vagas
/api/v1/vagas/{id}
/teste
Components
Schemas
AddressProjection
AuthLoginRequest
AuthRefreshTokenRequest
AuthTokenResponse
ContactProjection
Contato
Endereco
IArea
NewJobRequest
OrgBasicInfoProjection
PageableObject
PagePerfilPublicoDaOrganizacao
C: > Users > andrei > Work > Talks > Scan 700,000 APIs > Sesame > samples > {} 30597.json > {} paths
Scan | Audit
{
"openapi": "3.0.1",
"info": {
"title": "API para Vagas de Estágio",
"description": "Um serviço web para conectar empresas e instituições de ensino em torno de um o
"termsOfService": "URL",
"contact": {
"url": ""
},
"license": {
"name": "Apache 2.0",
"url": "URL"
},
"version": "0.1.0"
},
"servers": [
{
"url": "http://localhost:8080",
"description": "Generated server url"
}
],
"tags": [···
],
"paths": {
"/api/v1/vagas/{id}": { ···
},
"/api/v1/org": {
"get": {
Scan | Try it | Audit
"tags": [
"Seu Perfil"
],
"summary": "Ver Perfil",
"description": "Ver Perfil Privado da organização",
"operationId": "getPerfil",
"responses": {
"200": {
"description": "OK",
"content": {
"application/json": {
"schema": {
"$ref": "#/components/schemas/SuccessResponsePerfilPrivado"
}
},
"application/xml": {
"schema": {
"$ref": "#/components/schemas/SuccessResponsePerfilPrivado"
}
},

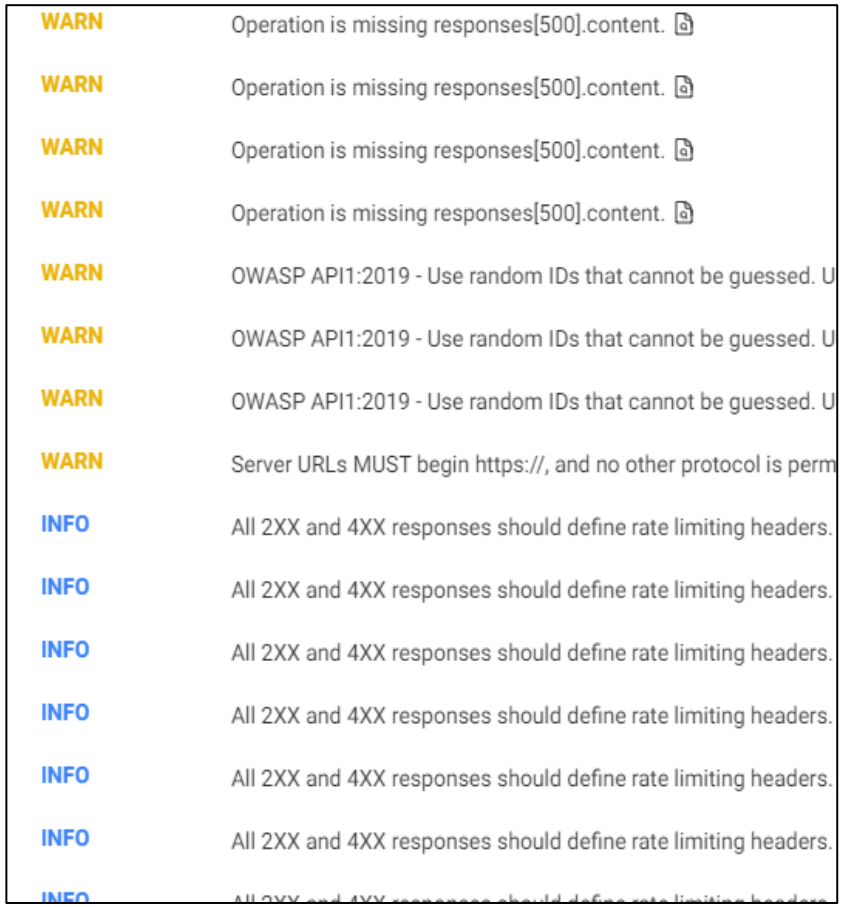
OpenAPI.Tools

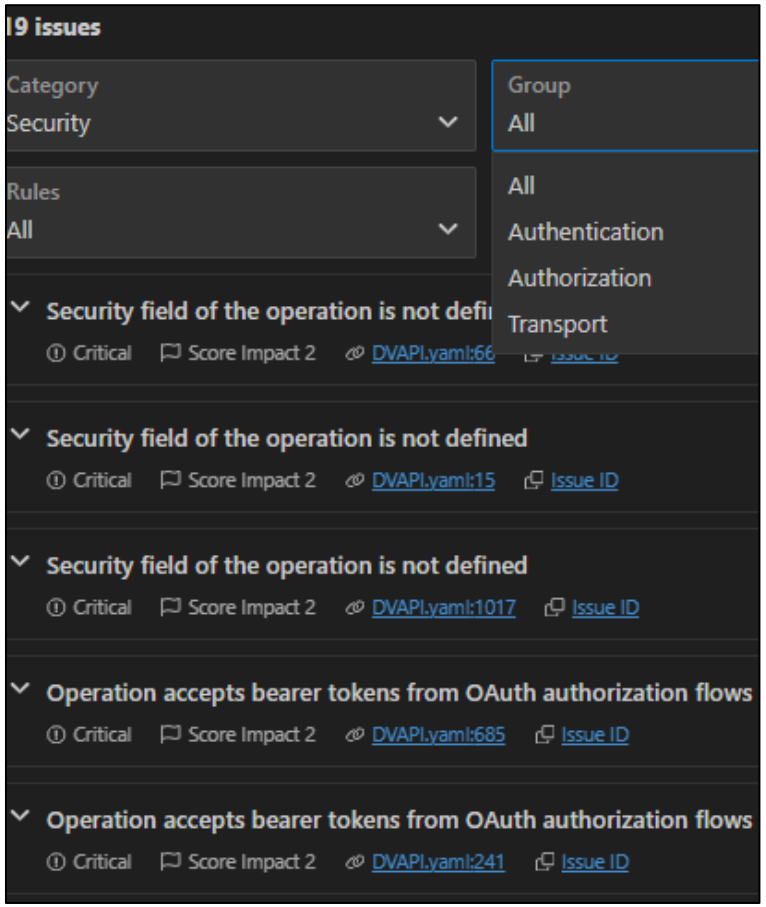
Sponsored by Zudoku - Open-source, OpenAPI powered, highly customizable API documentation.

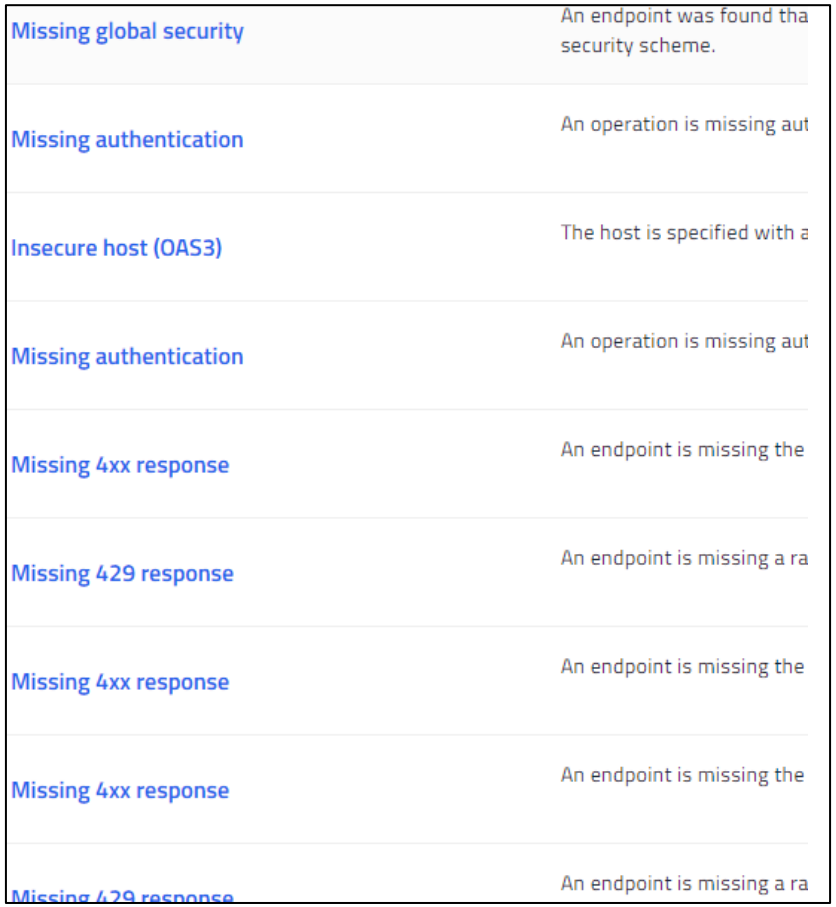
Tool Types

We've organised everything into categories so you can jump to the section you're interested in.

- **Auto Generators:** Tools that will take your code and turn it into an OpenAPI Specification document
- **Converters:** Various tools to convert to and from OpenAPI and other API description formats.
- **Data Validators:** Check to see if API requests and responses are lining up with the API description.
- **Description Validators:** Check your API description to see if it is valid OpenAPI.
- **Documentation:** Render API Description as HTML (or maybe a PDF) so slightly less technical people can figure out how to work with the API.
- **DSL:** Writing YAML by hand is no fun, and maybe you don't want a GUI, so use a Domain Specific Language to write OpenAPI in your language of choice.
- **Gateways:** API Gateways and related tools that have integrated support for OpenAPI.
- **GUI Editors:** Visual editors help you design APIs without needing to memorize the entire OpenAPI specification.
- **Learning:** Whether you're trying to get documentation for a third party API based on traffic, or are trying to switch to design-first at an organization with no OpenAPI at all, learning can help you move your API spec forward and keep it up to date.
- **Miscellaneous:** Anything else that does stuff with OpenAPI but hasn't quite got enough to warrant its own category.
- **Mock Servers:** Fake servers that take description document as input, then route incoming HTTP requests to example responses or dynamically generates examples.
- **Monitoring:** Monitoring tools let you know what is going on in your API.
- **Parsers:** Loads and read OpenAPI descriptions, so you can work with them programmatically.
- **SDK Generators:** Generate code to give to consumers, to help them avoid interacting at a HTTP level.
- **Security:** By poking around your OpenAPI description, some tools can look out for attack vectors you might not have noticed.
- **Server Implementations:** Easily create and implement resources and routes for your APIs.
- **Testing:** Quickly execute API requests and validate responses on the fly through command line or GUI interfaces.
- **Text Editors:** Text editors give you visual feedback whilst you write OpenAPI, so you can see what docs might look like.

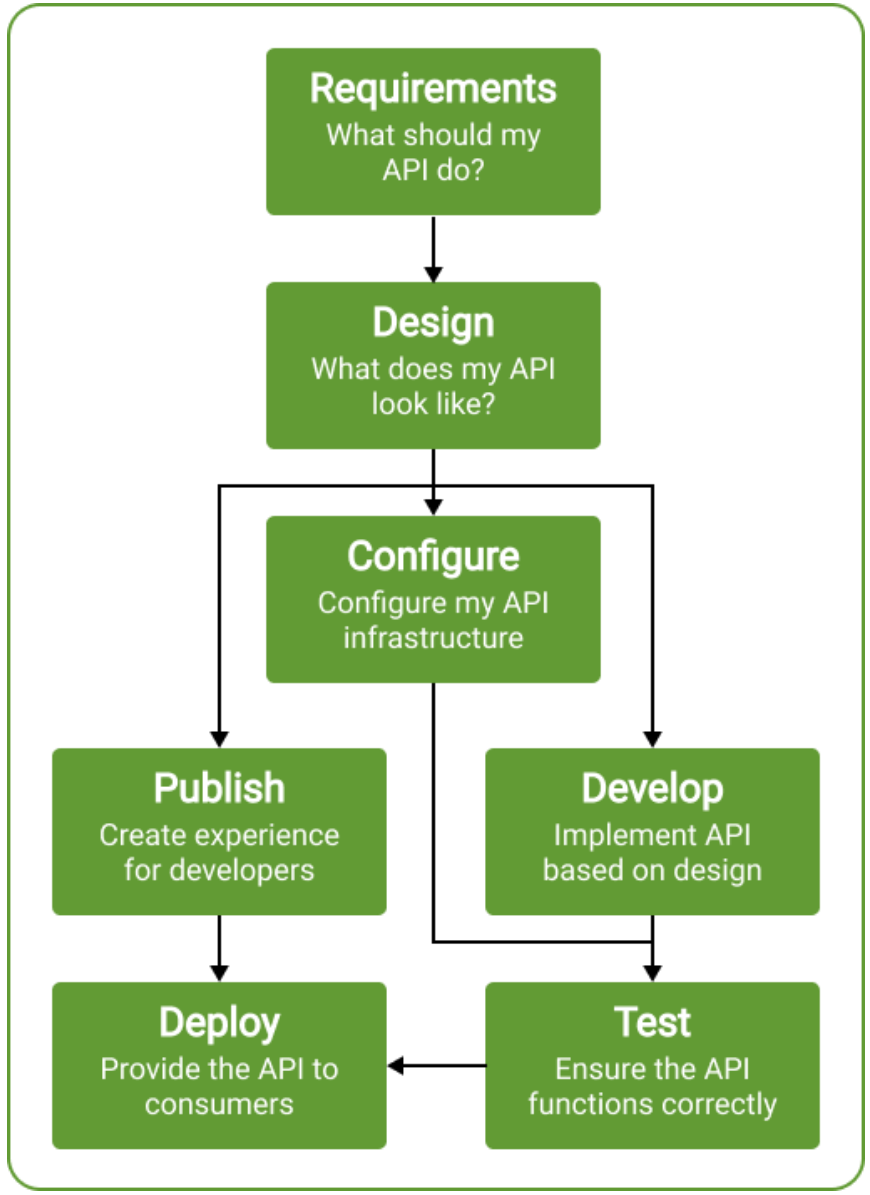

WARN Operation is missing responses[500].content.
WARN Operation is missing responses[500].content.
WARN Operation is missing responses[500].content.
WARN Operation is missing responses[500].content.
WARN OWASP API1:2019 - Use random IDs that cannot be guessed. U
WARN OWASP API1:2019 - Use random IDs that cannot be guessed. U
WARN OWASP API1:2019 - Use random IDs that cannot be guessed. U
WARN Server URLs MUST begin https://, and no other protocol is perm
INFO All 2XX and 4XX responses should define rate limiting headers.
INFO All 2XX and 4XX responses should define rate limiting headers.
INFO All 2XX and 4XX responses should define rate limiting headers.
INFO All 2XX and 4XX responses should define rate limiting headers.
INFO All 2XX and 4XX responses should define rate limiting headers.
INFO All 2XX and 4XX responses should define rate limiting headers.

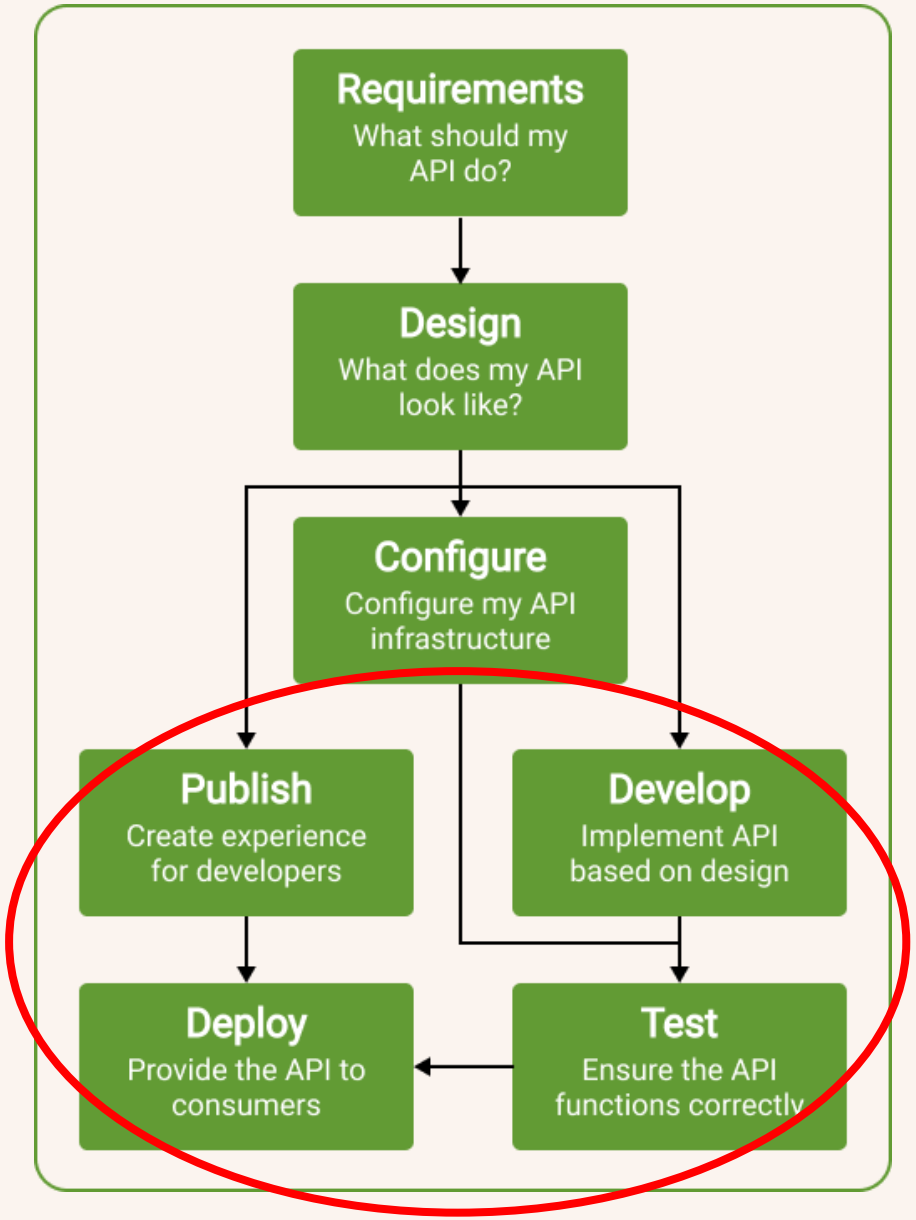

19 issues
Category
Security
Group
All
Rules
All
All
Authentication
Authorization
Transport
Security field of the operation is not defi
Critical
Score Impact 2
DVAPI.yaml:66
Security field of the operation is not defined
Critical
Score Impact 2
DVAPI.yaml:15
Issue ID
Security field of the operation is not defined
Critical
Score Impact 2
DVAPI.yaml:1017
Issue ID
Operation accepts bearer tokens from OAuth authorization flows
Critical
Score Impact 2
DVAPI.yaml:685
Issue ID
Operation accepts bearer tokens from OAuth authorization flows
Critical
Score Impact 2
DVAPI.yaml:241
Issue ID

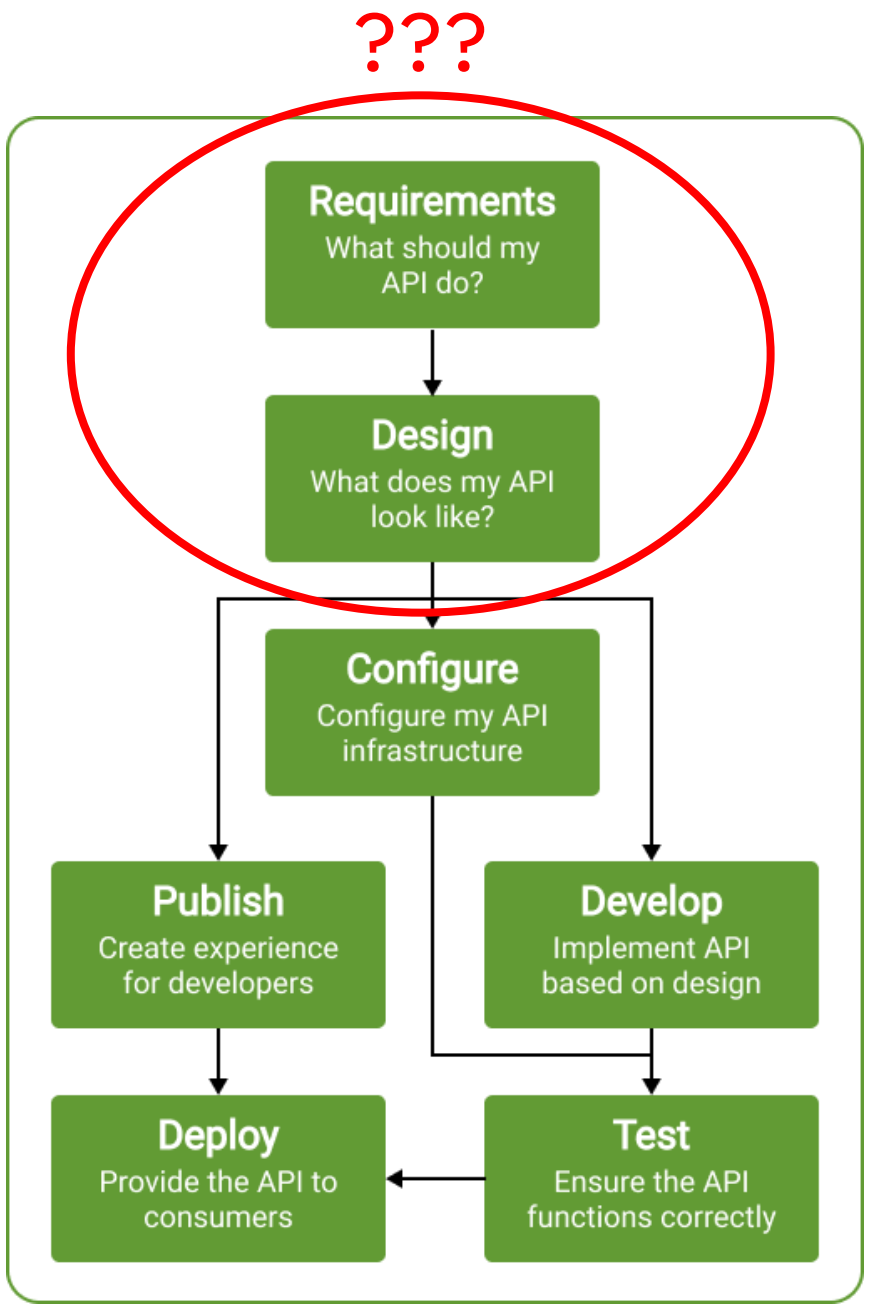

Missing global security
An endpoint was found tha
security scheme.
Missing authentication
An operation is missing aut
Insecure host (OAS3)
The host is specified with a
Missing authentication
An operation is missing aut
Missing 4xx response
An endpoint is missing the
Missing 429 response
An endpoint is missing a ra
Missing 4xx response
An endpoint is missing the
Missing 4xx response
An endpoint is missing the
An endpoint is missing a ra

ROLE IN DEVELOPMENT LIFECYCLE

1. Requirements
2. Design
3. Configuration
4. Development
5. Publish
6. Deployment
7. Test




Requirements
What should my API do?
Design
What does my API look like?
Configure
Configure my API infrastructure
Publish
Create experience for developers
Develop
Implement API based on design
Deploy
Provide the API to consumers
Test
Ensure the API functions correctly


???
Requirements
What should my API do?
Design
What does my API look like?
Configure
Configure my API infrastructure
Publish
Create experience for developers
Develop
Implement API based on design
Deploy
Provide the API to consumers
Test
Ensure the API functions correctly

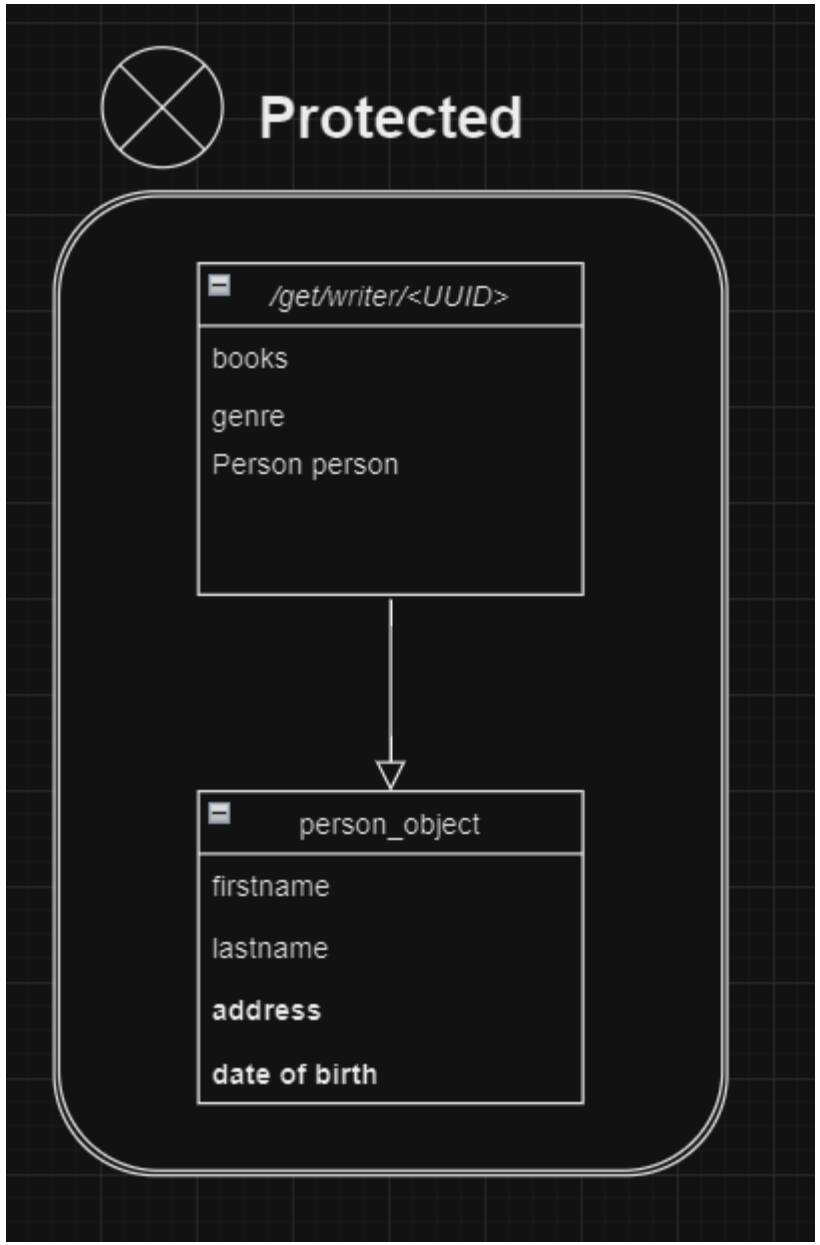
3. RESEARCH

QUESTIONS

1. CAN WE USE THE OPENAPI DOCS TO FIND DESIGN FAULTS?
2. WHAT TYPE OF VULN CLASSES ARE THESE?
3. IDEAS AND EXAMPLES?

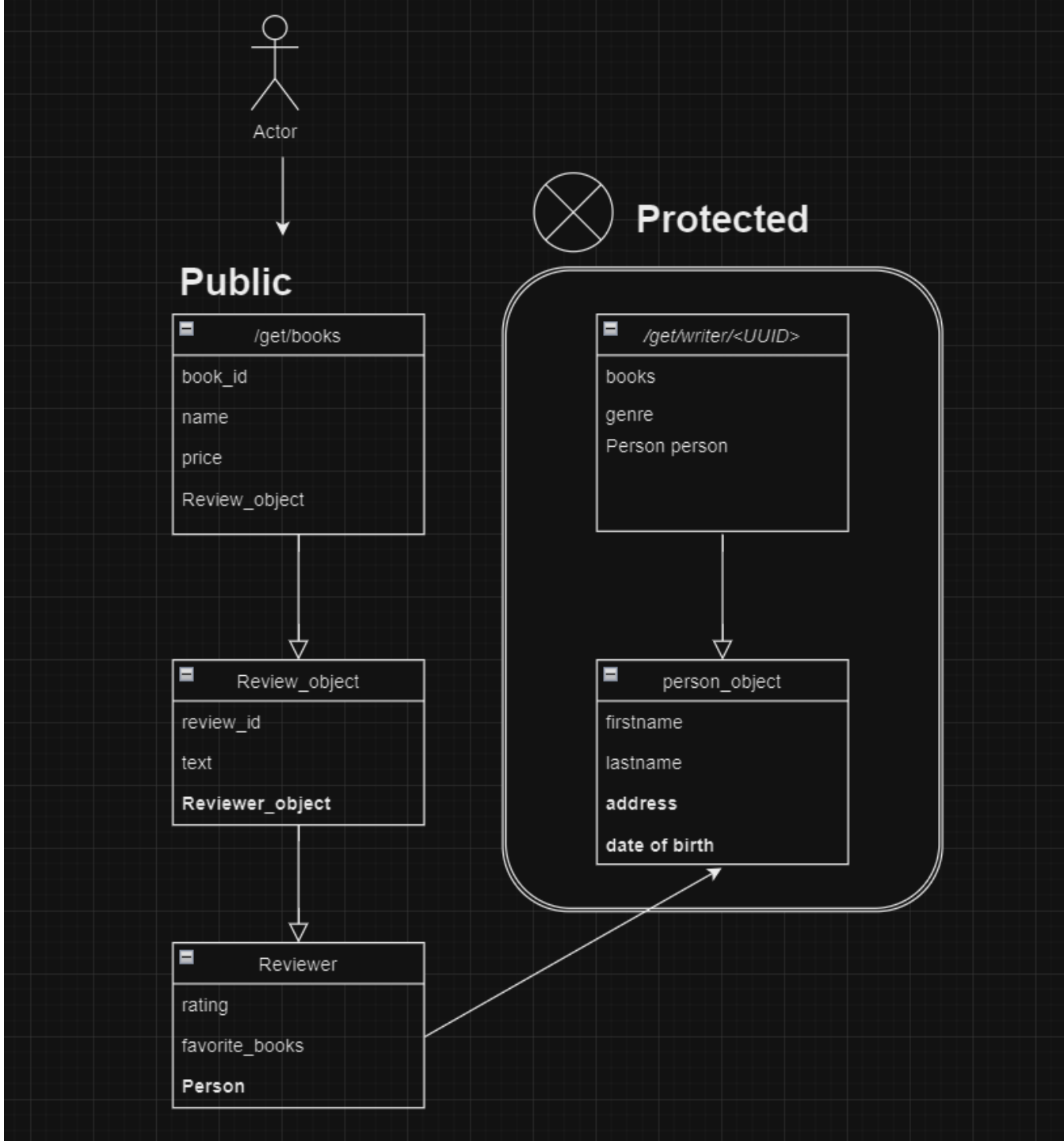
1. INDIRECT BROKEN ACCESS CONTROL

Looks fine ¯_(ツ)_/¯



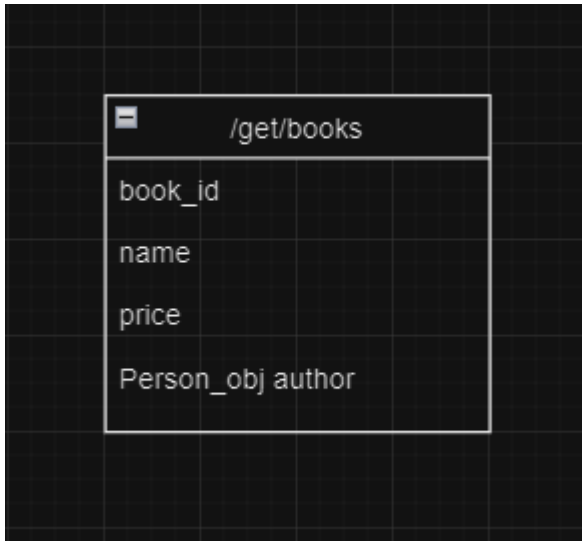
1. INDIRECT BROKEN ACCESS CONTROL

1. FIND OBJECT WITH SENSITIVE INFO
2. CHECK ALL PATHS TO READ/MODIFY IT
3. CHECK FOR DIRECT AND INDIRECT ACCESS



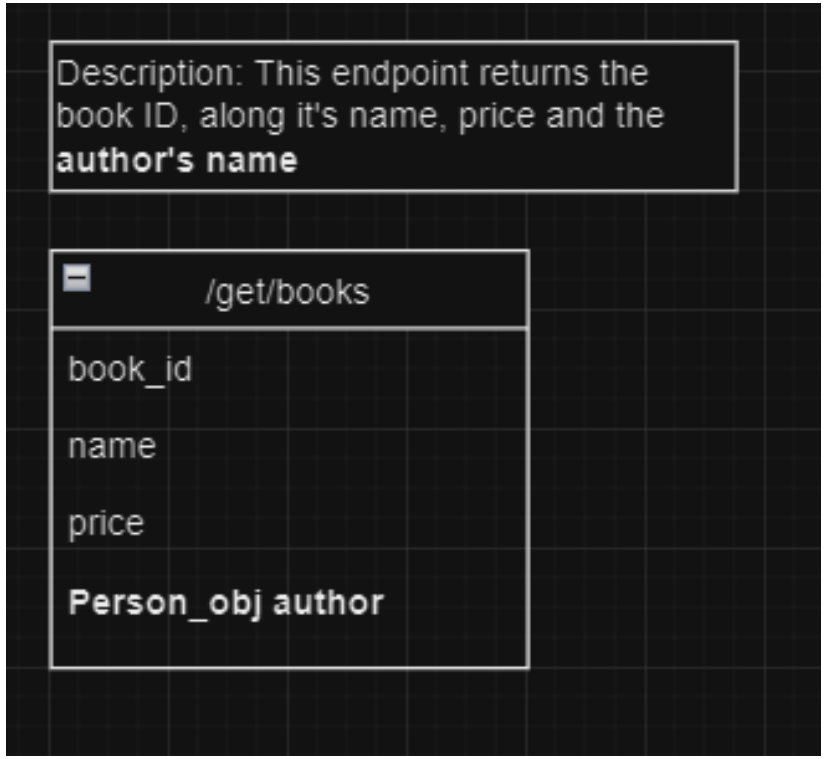
2. INCONSISTENT DESIGN

VULNERABLE?



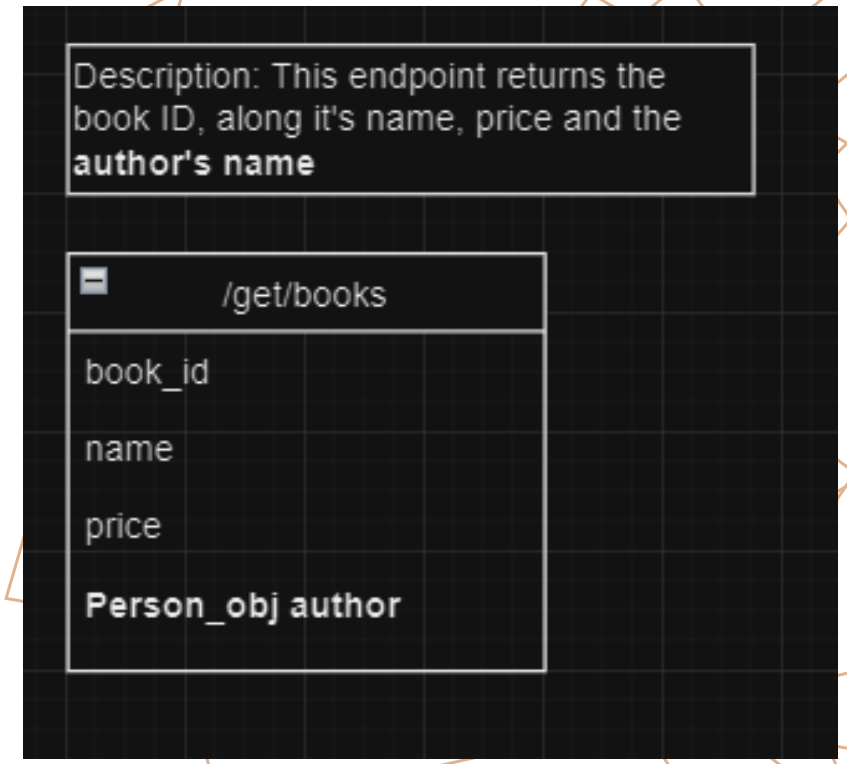
2. INCONSISTENT DESIGN

VULNERABLE?



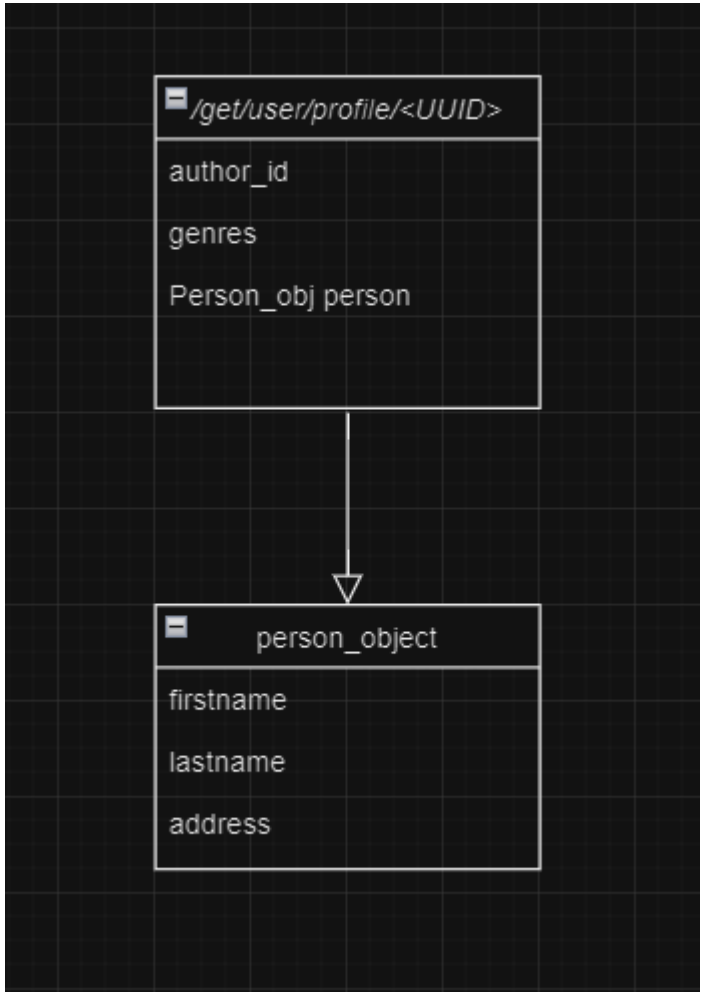
2. INCONSISTENT DESIGN

1. CHECK ENDPOINT'S INPUT/OUPUT
2. ANALYZE ENDPOINT'S DESCRIPTION
3. DETERMINE IF THEY MATCH



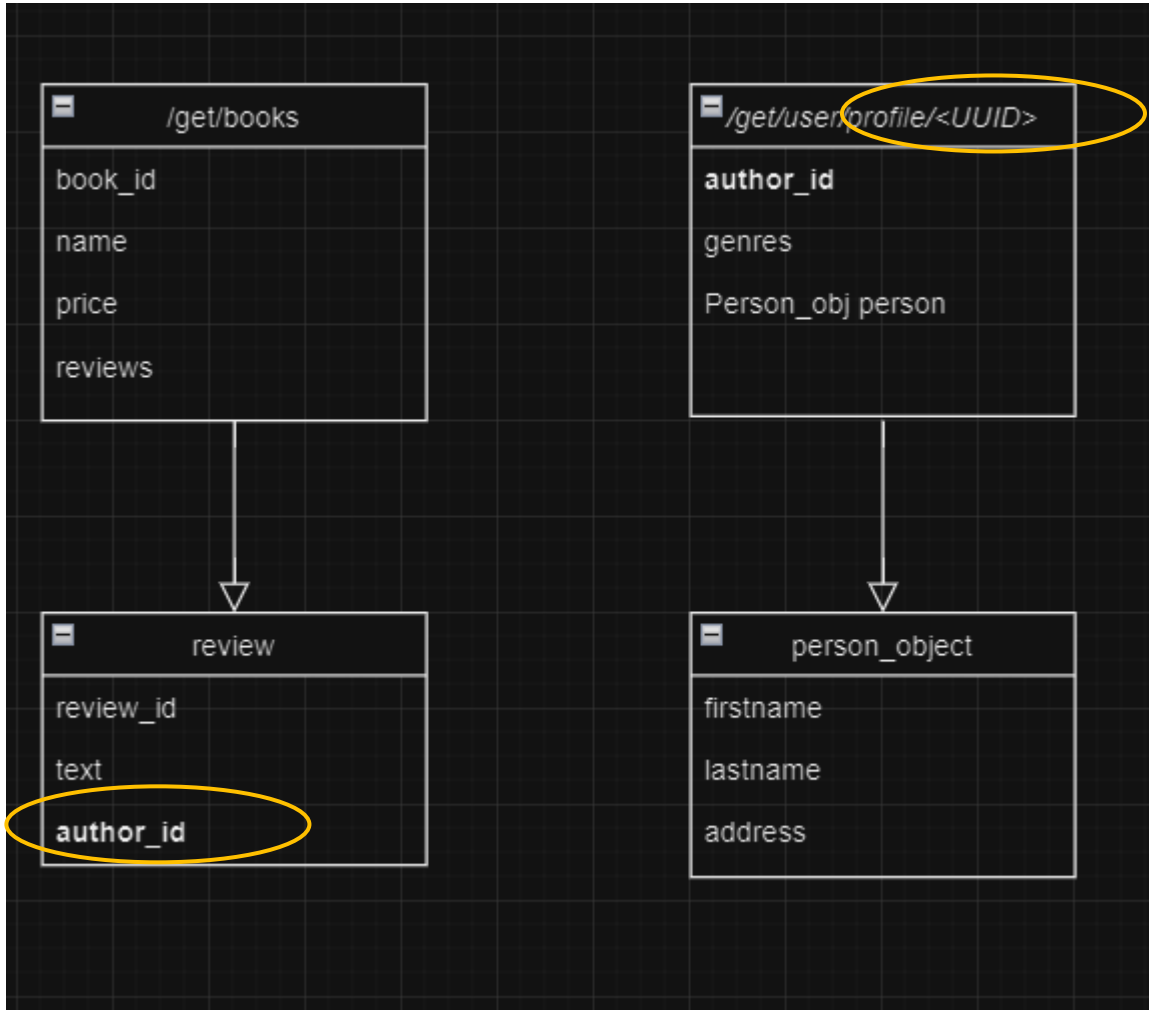
3. IDOR ID FINDER

1. UUID might be used for access control
2. But how to check?



3. IDOR ID FINDER

1. FIND ENDPOINTS THAT REQUIRE ID
2. DETERMINE WHERE THESE IDS ARE LEAKED
3. OBTAIN THEM AND TEST ACCESS CONTROL



4. THE TOOL

CHARACTERISTICS

1. WORK ON ANY OPENAPI REGARDLESS OF THE API STRUCTURE
2. VISUAL + SCRIPTING
3. EASY CONFIGURABLE

OPTIONS

1. JSON SCHEMA-BASED OBJECT NAVIGATION
2. GRAPH TRAVERSAL ALGORITHMS
3. GRAPH DATABASE LIKE NEO4J

SABE@TOKYO.JAPAN.LOCAL

Database Info | Node Info | Queries

Node Info

Name	LEO@JAPAN.LOCAL
OS	Windows 7 Enterprise N Service Pack 1
Enabled	True
Allows Unconstrained Delegation	False
Compromised	False
Sessions	1
Sibling Objects in the Same OU	3
Effective Inbound GPOs	1

See Computer within Domain/OU Tree

Local Admins

Explicit Admins	3
Unrolled Admins	7
Derivative Local Admins	27

Group Memberships

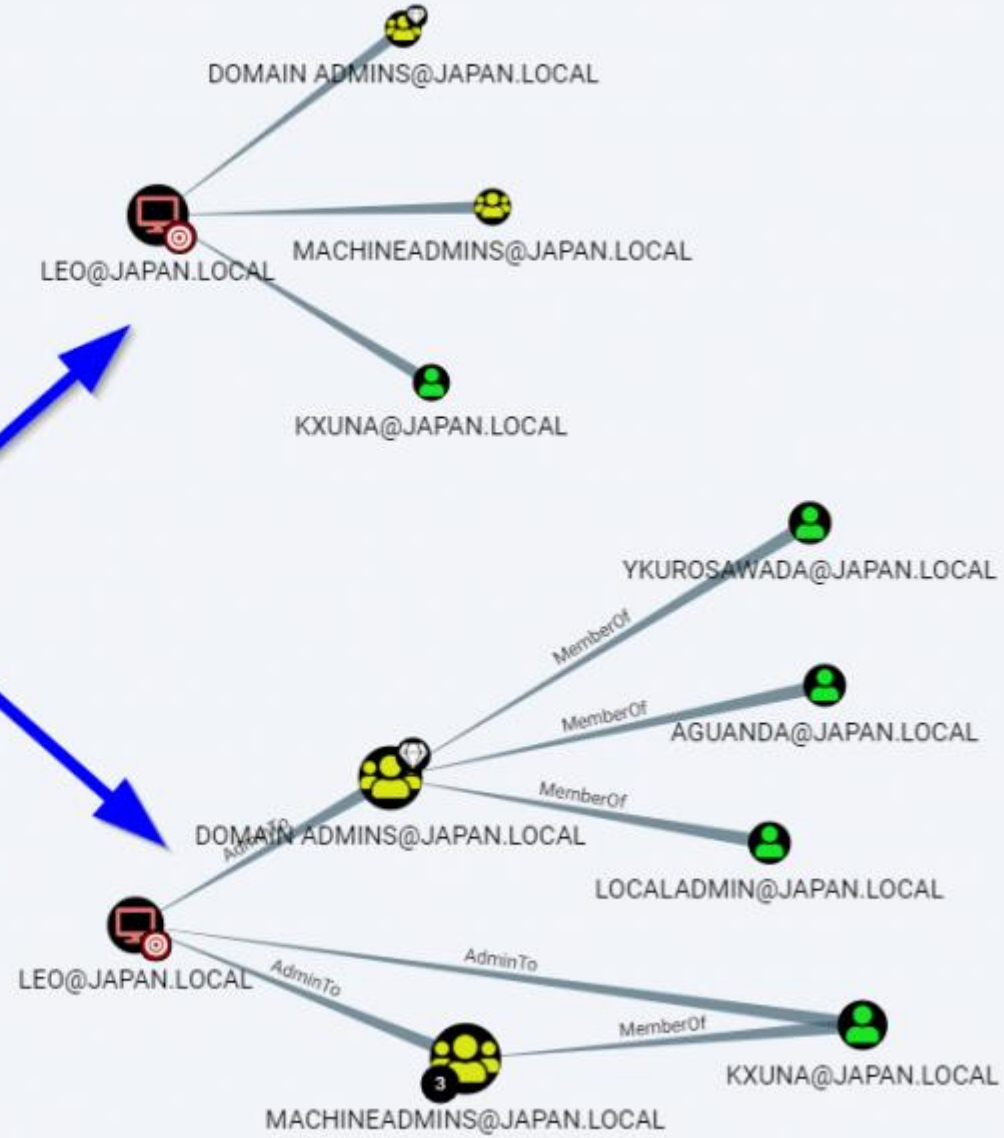
First Degree Group Membership	1
Unrolled Group Membership	1
Foreign Group Membership	0

Local Admin Rights

First Degree Local Admin	0
Group Delegated Local Admin	0
Derivative Local Admin	0

Outbound Object Control

First Degree Object Control	0
Group Delegated Object Control	0
Transitive Object Control	1



CONCEPT

1. SIMILAR TO BLOODHOUND, BUT FOR REST API
2. ENDPOINTS, PARMETERS, OBJECTS, ETC. -> NODES
3. FIND VULNS USING NEO4J QUERIES AND PATHS

Medium Search Write

Get unlimited access to the best of Medium for less than $1/week. Become a member

Uploading JSON data to Neo4j

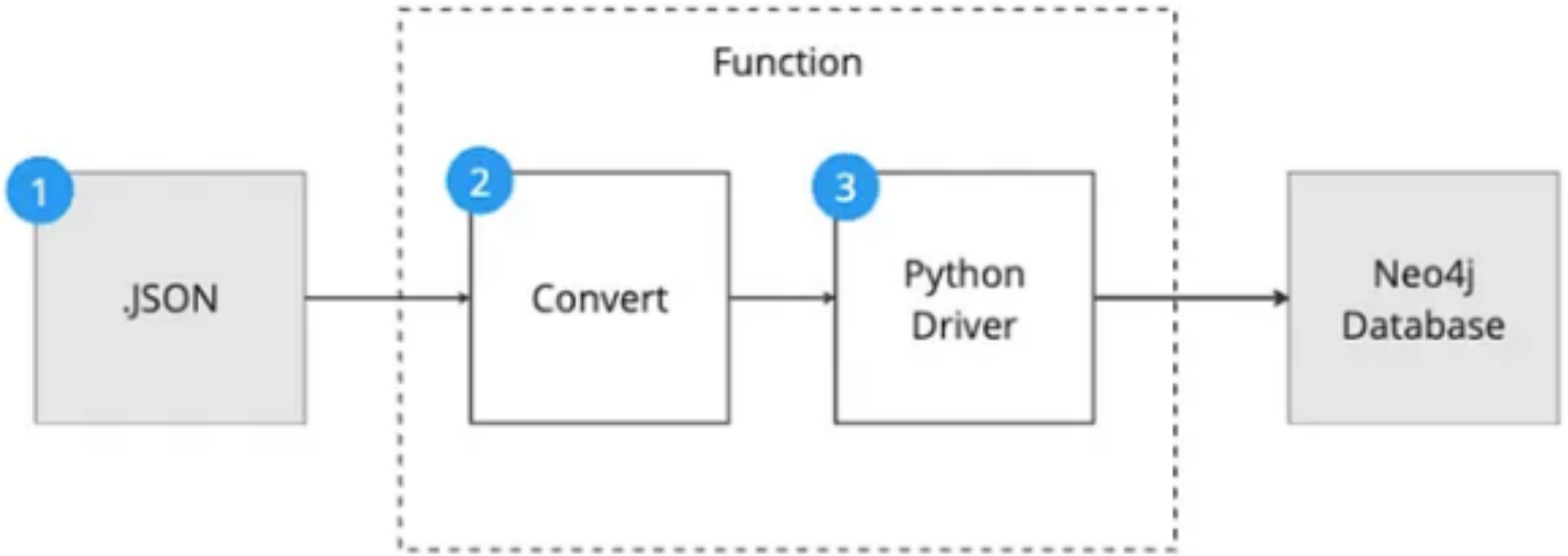
Jason Koo · Follow

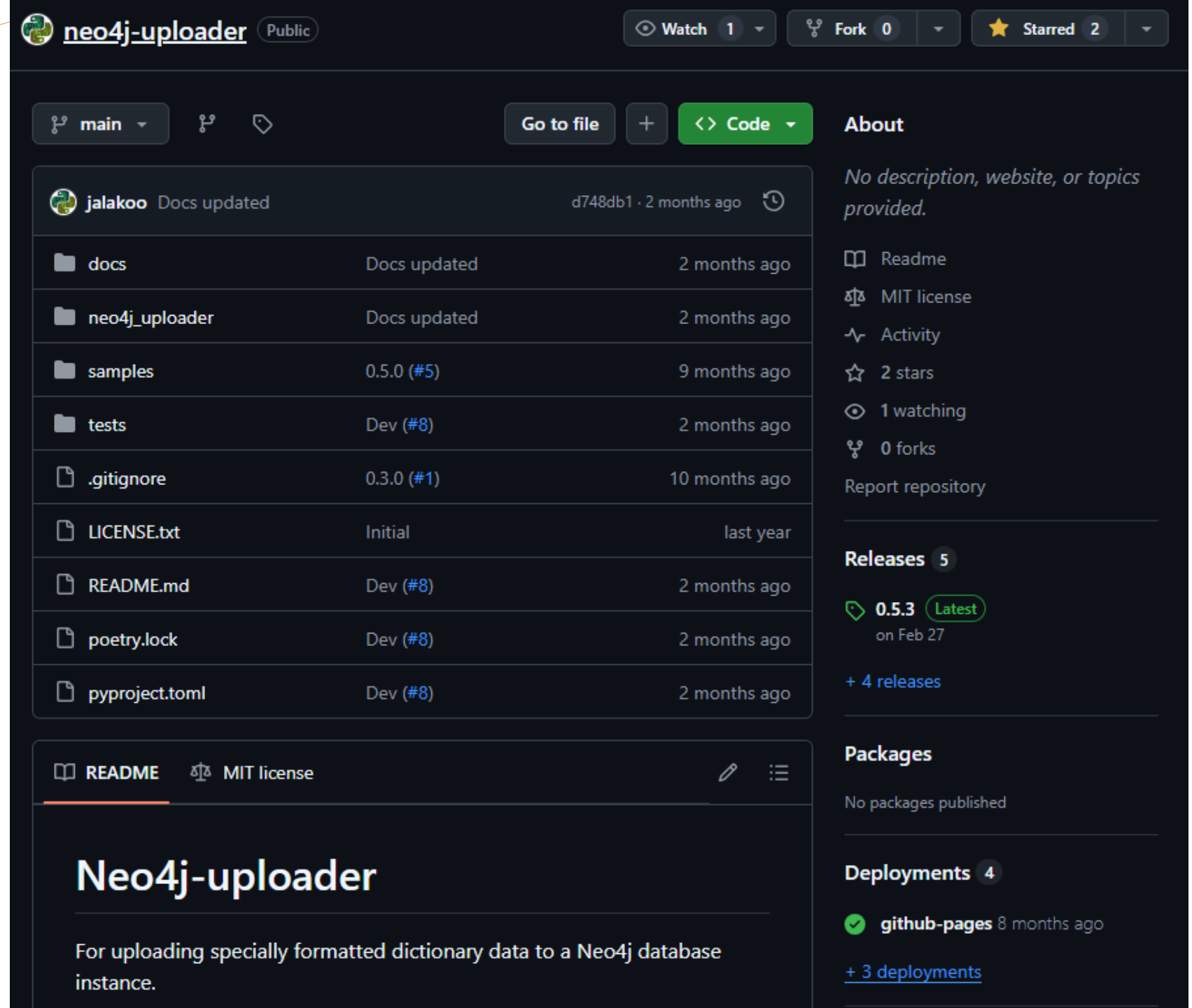
7 min read · Dec 15, 2023

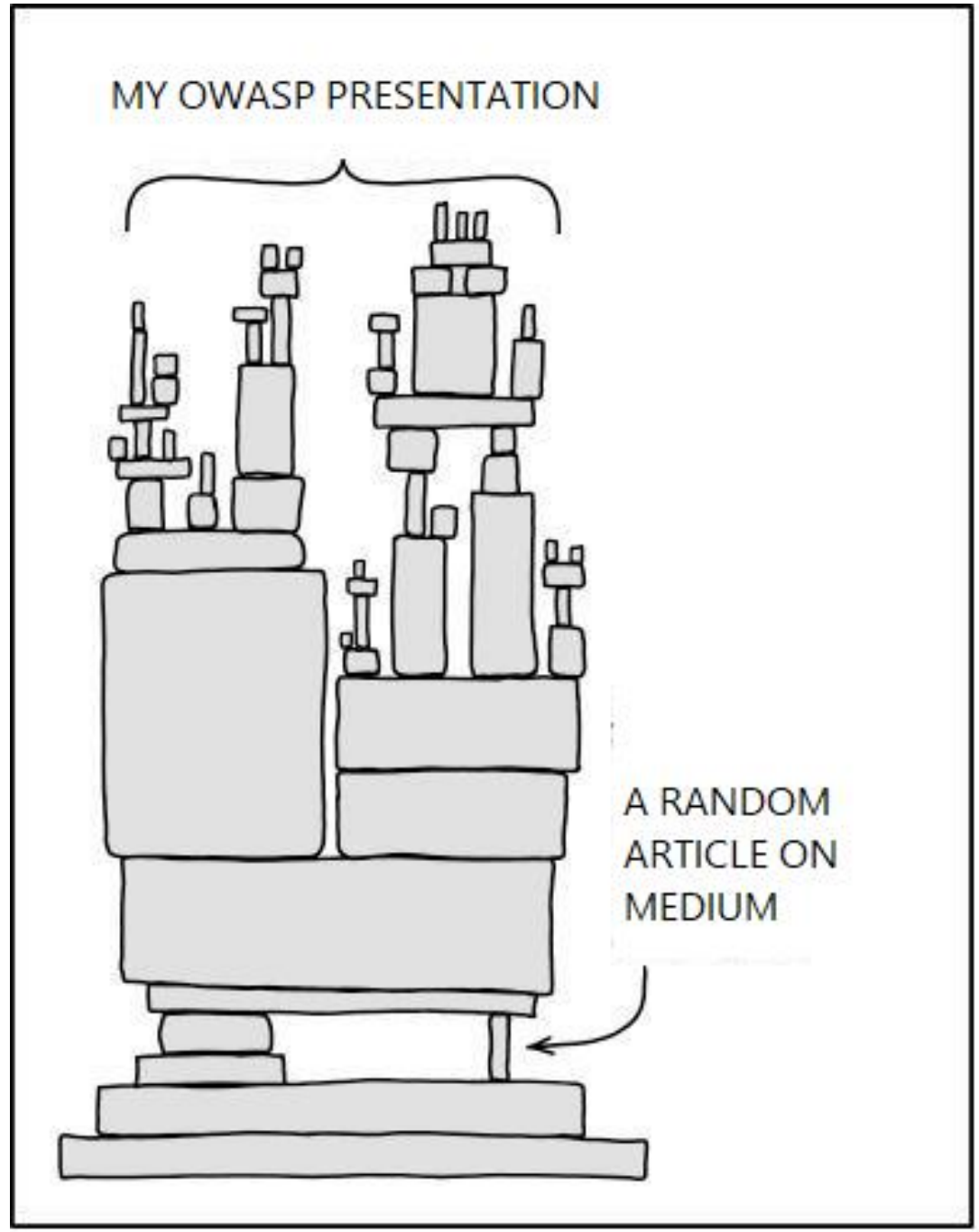
5

There are 3 main parts to this:

1. An opinionated .json schema that the source data needs to conforms to.
2. Convert the data for use by the official Neo4j Python driver.
3. Upload the converted data via the driver

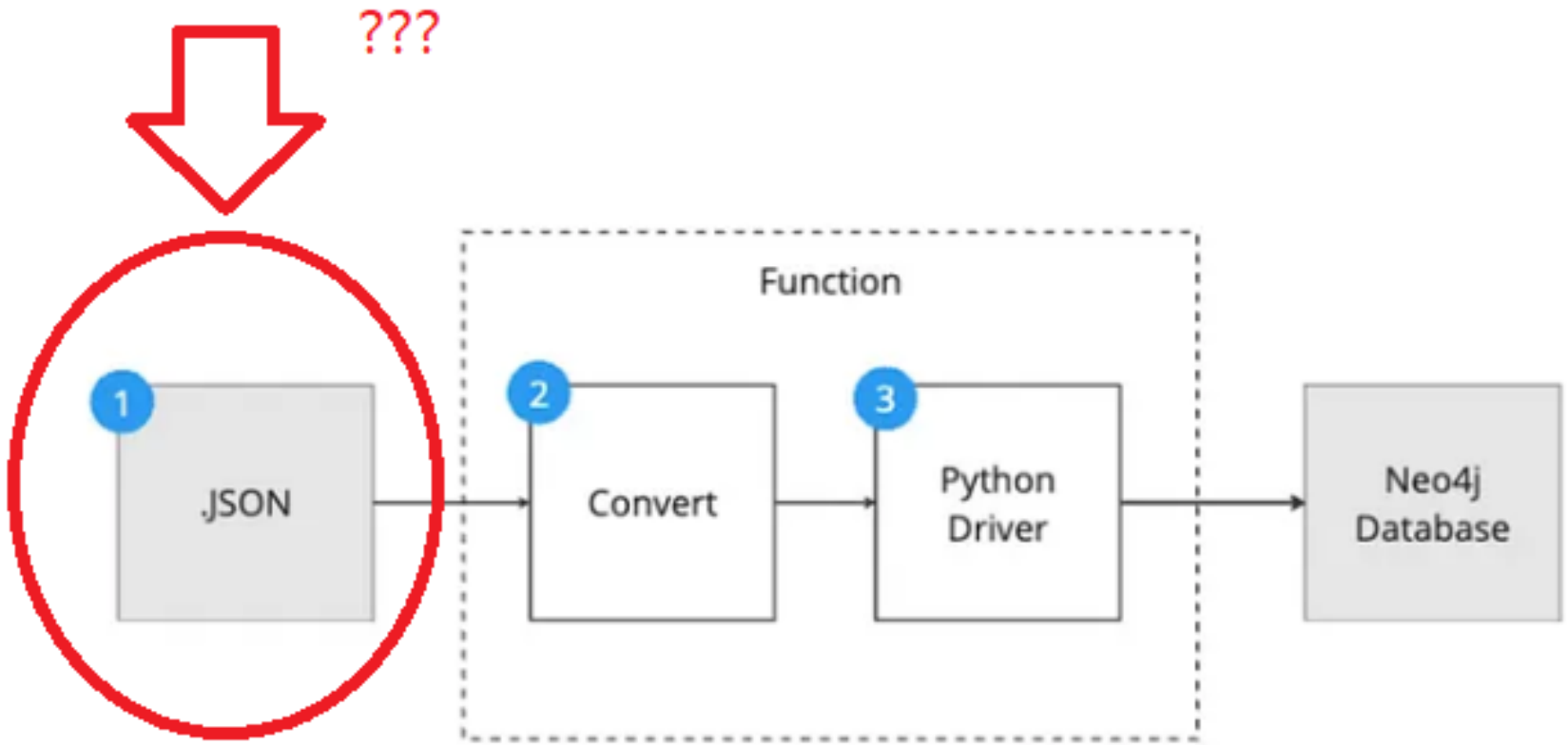


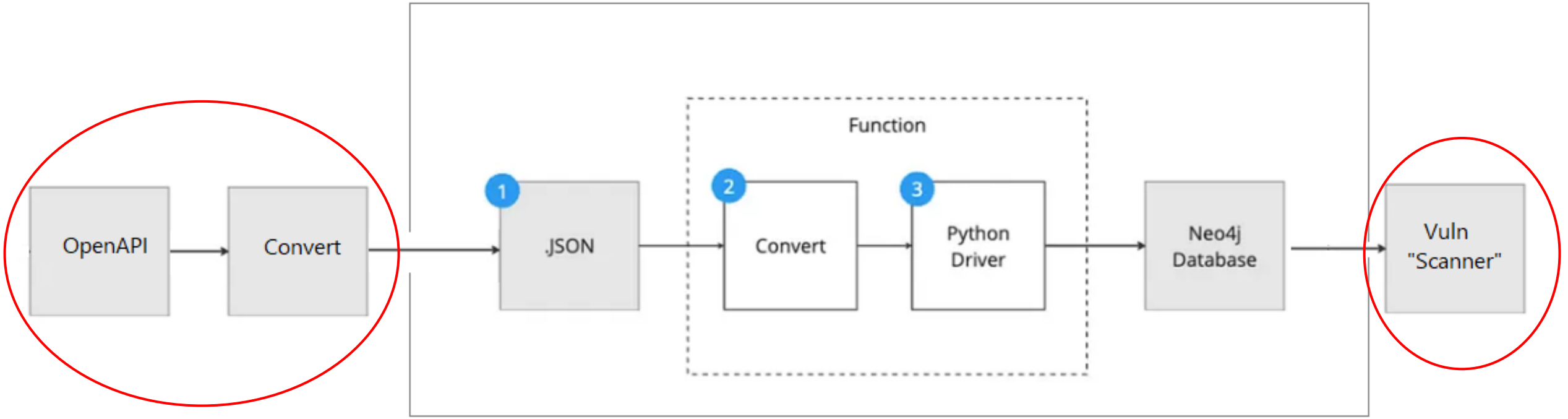

neo4j-uploader
Public
Watch 1
Fork 0
Starred 2
main
Go to file
Code
jalakoo Docs updated
d748db1 · 2 months ago
docs Docs updated 2 months ago
neo4j_uploader Docs updated 2 months ago
samples 0.5.0 (#5) 9 months ago
tests Dev (#8) 2 months ago
.gitignore 0.3.0 (#1) 10 months ago
LICENSE.txt Initial last year
README.md Dev (#8) 2 months ago
poetry.lock Dev (#8) 2 months ago
pyproject.toml Dev (#8) 2 months ago
README
MIT license
Neo4j-uploader
For uploading specially formatted dictionary data to a Neo4j database instance.
About
No description, website, or topics provided.
Readme
MIT license
Activity
2 stars
1 watching
0 forks
Report repository
Releases 5
0.5.3 Latest
on Feb 27
+ 4 releases
Packages
No packages published
Deployments 4
github-pages 8 months ago
+ 3 deployments

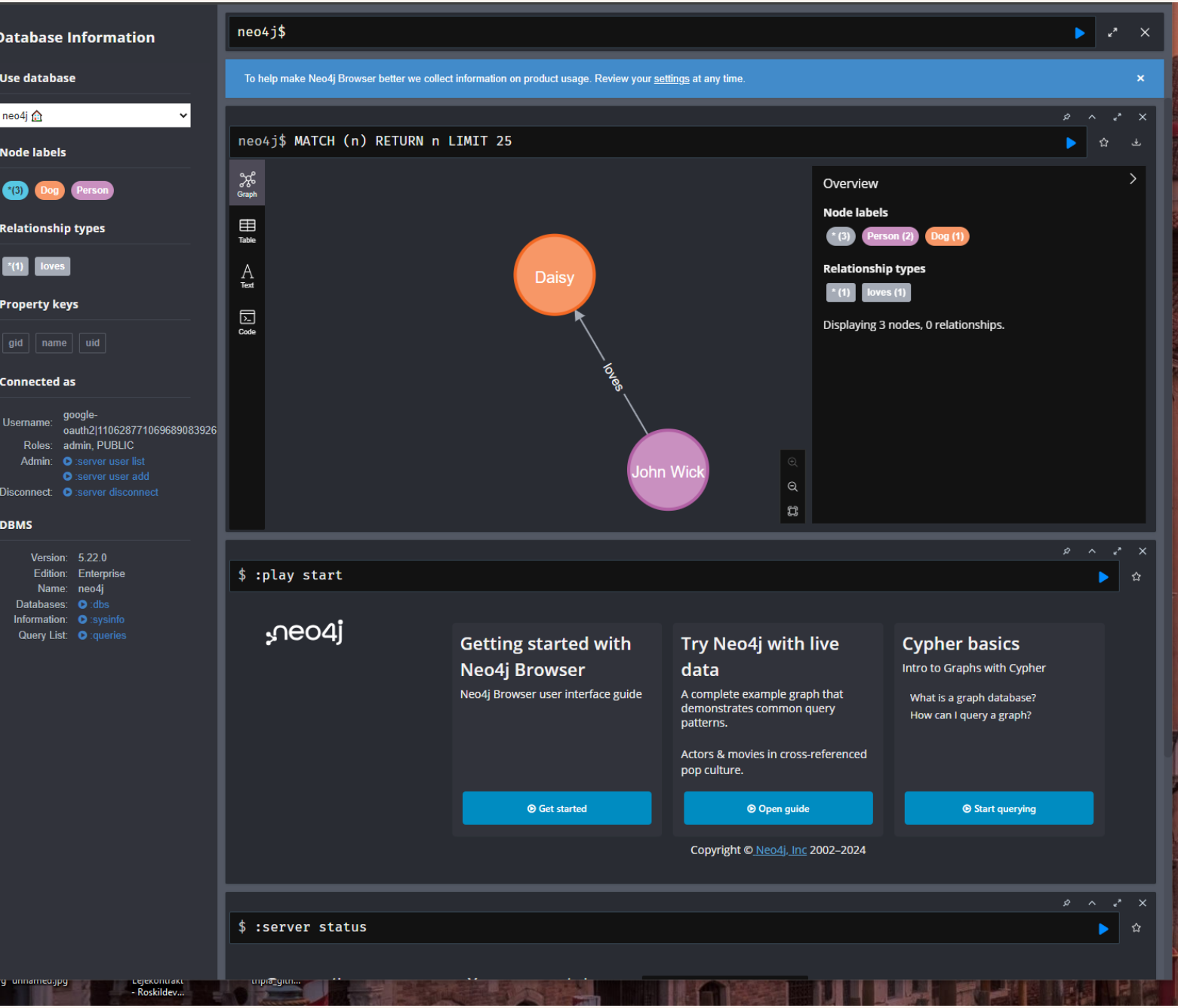

MY OWASP PRESENTATION
A RANDOM ARTICLE ON MEDIUM

There are 3 main parts to this:

1. An opinionated .json schema that the source data needs to conforms to.
2. Convert the data for use by the official Neo4j Python driver.
3. Upload the converted data via the driver




OpenAPI
Convert
1
JSON
Function
2
Convert
3
Python
Driver
Neo4j
Database
Vuln
"Scanner"

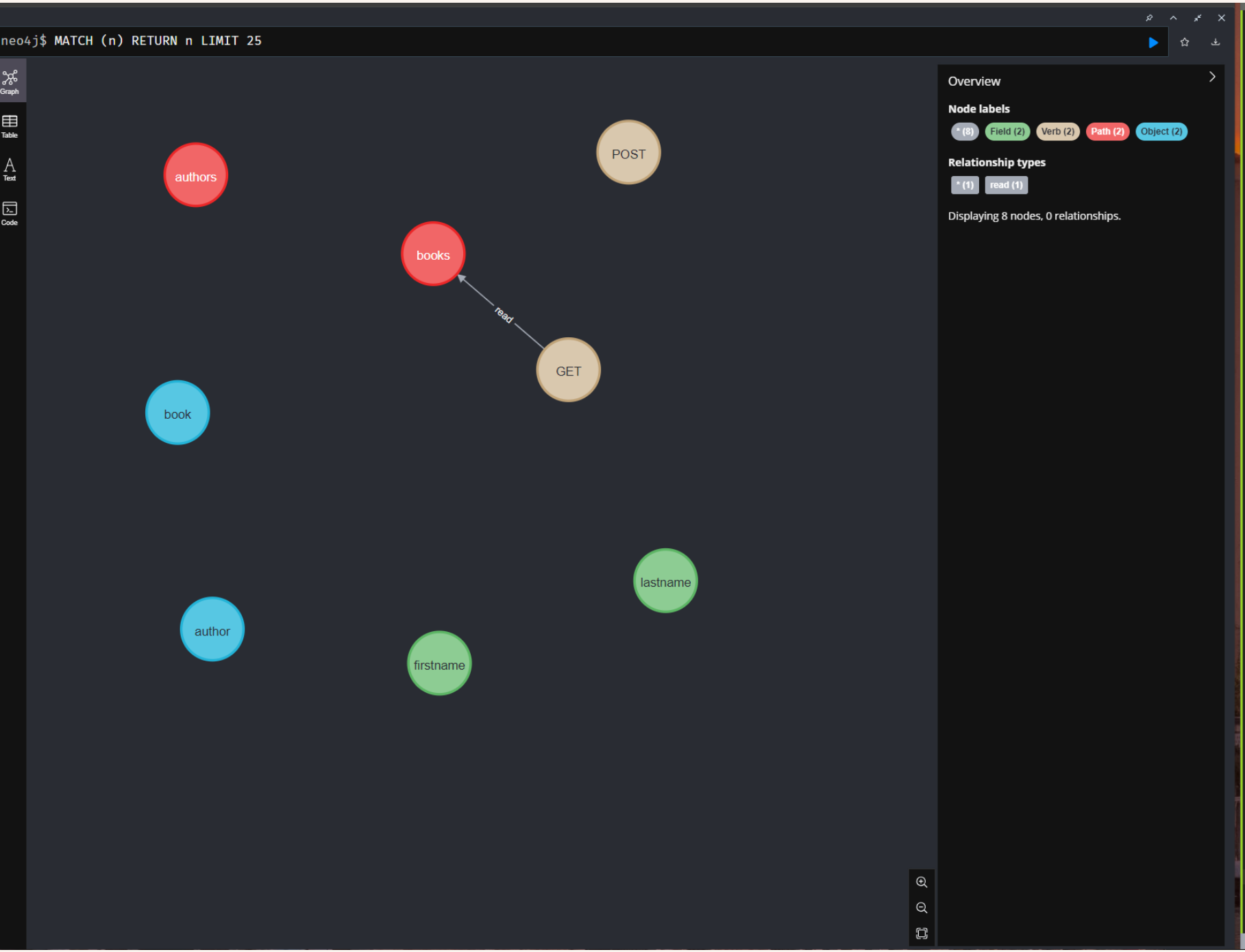



```
13              "key":"uid",
14              "records":[
15                  {
16                      "uid":"abc",
17                      "name": "John Wick"
18                  },
19                  {
20                      "uid":"bcd",
21                      "name":"Cane"
22                  }
23              ]
24          },
25          {
26              "labels":["Dog"],
27              "key": "gid",
28              "records":[
29                  {
30                      "gid":"abc",
31                      "name": "Daisy"
32                  }
33              ]
34          }
35      ],
36      "relationships": [
37          {
38              "type":"loves",
39              "from_node": {
40                  "record_key":"_from_uid",
41                  "node_key":"uid",
42                  "node_label":"Person"
43              },
44              "to_node": {
45                  "record_key":"_to_gid",
46                  "node_key":"gid",
47                  "node_label": "Dog"
```

Line 41, Column 34 Spaces: 4

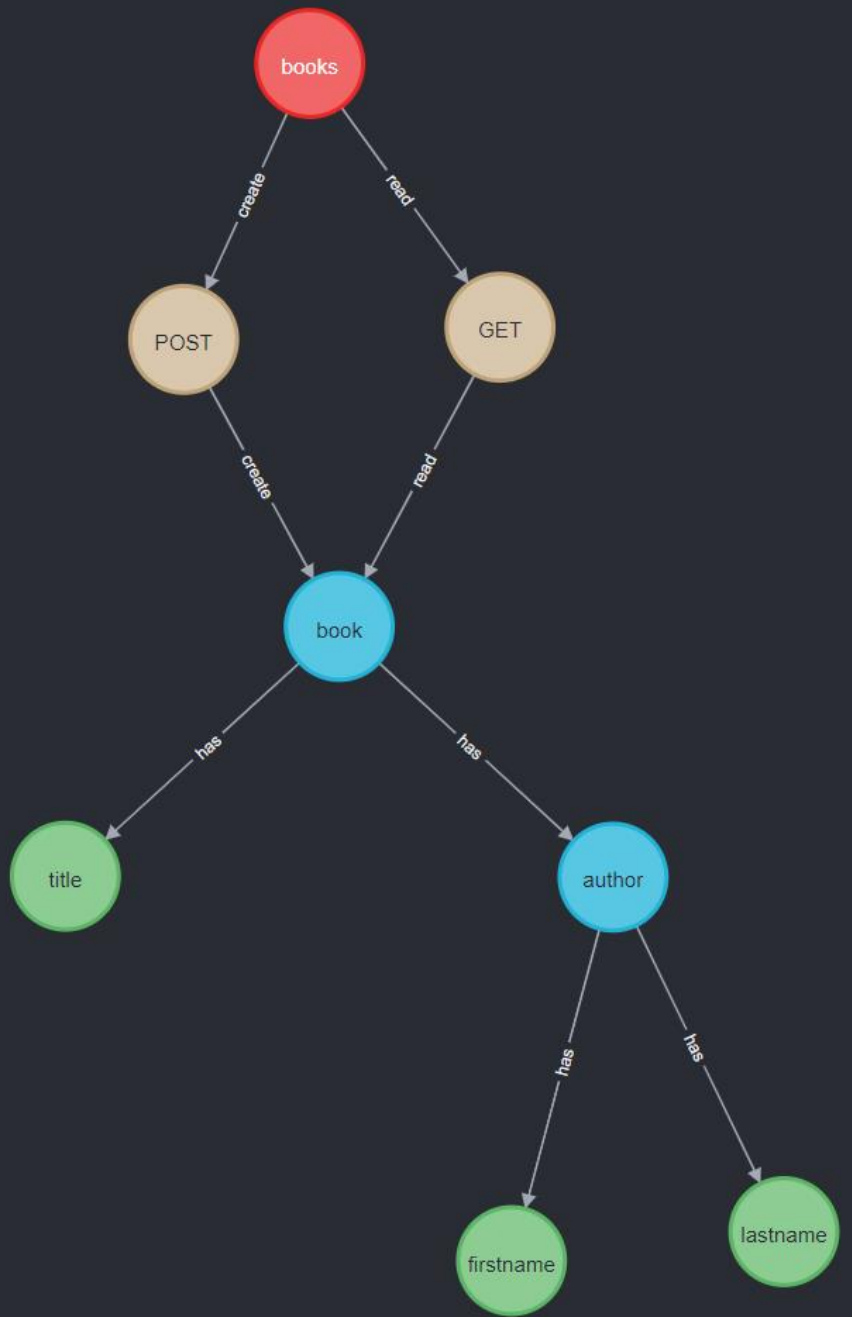
Command Prompt

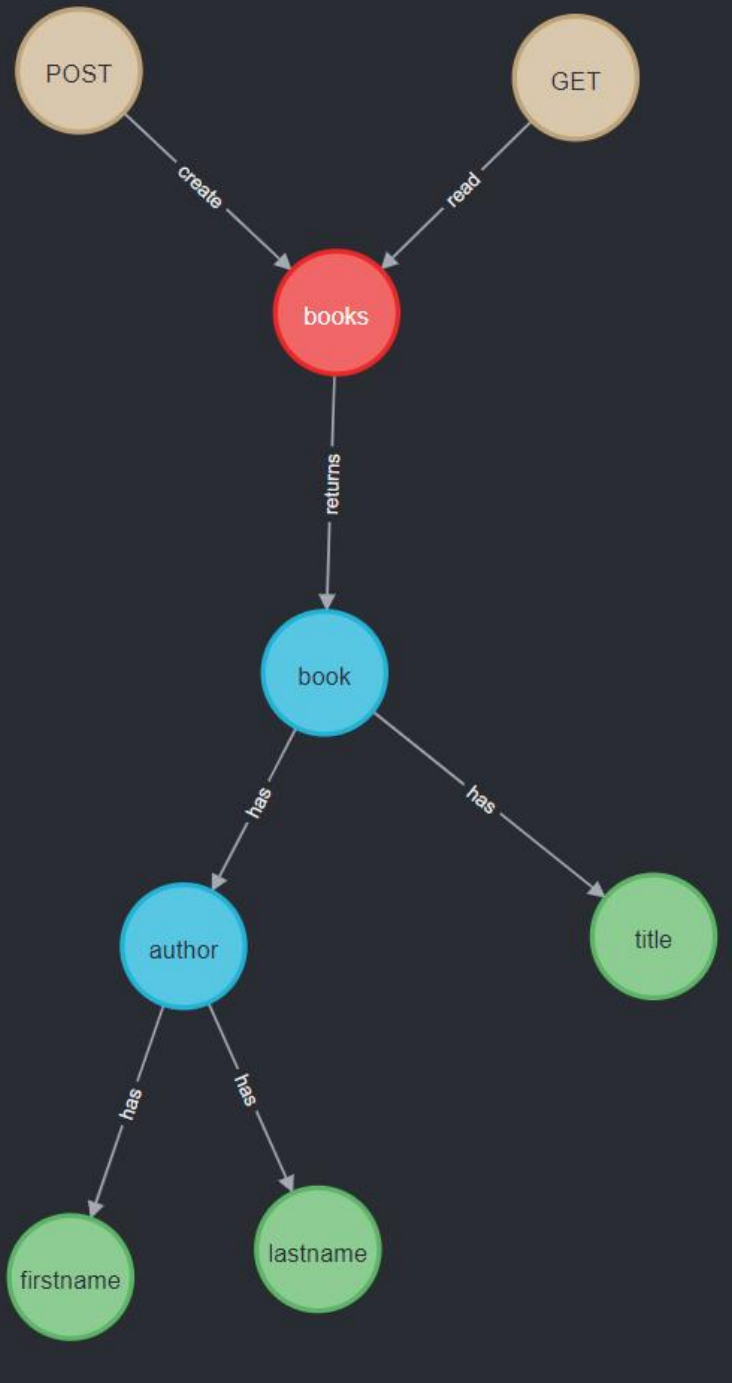
```
    For further information visit https://errors.pydantic.dev/2.9/v/list_type
relationships
  Input should be a valid list [type=list_type, input_value={'KNOWS': [{'_from_name':...'Jane', 'since': 2015}]}, i
type=dict]
    For further information visit https://errors.pydantic.dev/2.9/v/list_type

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "C:\Users\andrei\Work\Talks\Scan 700,000 APIs\Sesame\sesame_neo4j.py", line 25, in <module>
    result = batch_upload(config, data)
  File "C:\Users\andrei\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.10_qbz5n2kfra8p0\LocalCache\local-
ges\Python310\site-packages\neo4j_uploader\__init__.py", line 178, in batch_upload
    for result in gen:
  File "C:\Users\andrei\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.10_qbz5n2kfra8p0\LocalCache\local-
ges\Python310\site-packages\neo4j_uploader\__init__.py", line 92, in batch_upload_generator
    raise InvalidPayloadError(e)
neo4j_uploader.errors.InvalidPayloadError: 2 validation errors for GraphData
nodes
  Input should be a valid list [type=list_type, input_value={'Person': [{'name': 'Bob...e': 'Jane', 'age': 25}]}, i
type=dict]
    For further information visit https://errors.pydantic.dev/2.9/v/list_type
relationships
  Input should be a valid list [type=list_type, input_value={'KNOWS': [{'_from_name':...'Jane', 'since': 2015}]}, i
type=dict]
    For further information visit https://errors.pydantic.dev/2.9/v/list_type

C:\Users\andrei\Work\Talks\Scan 700,000 APIs\Sesame>python3 sesame_neo4j.py

C:\Users\andrei\Work\Talks\Scan 700,000 APIs\Sesame>
```



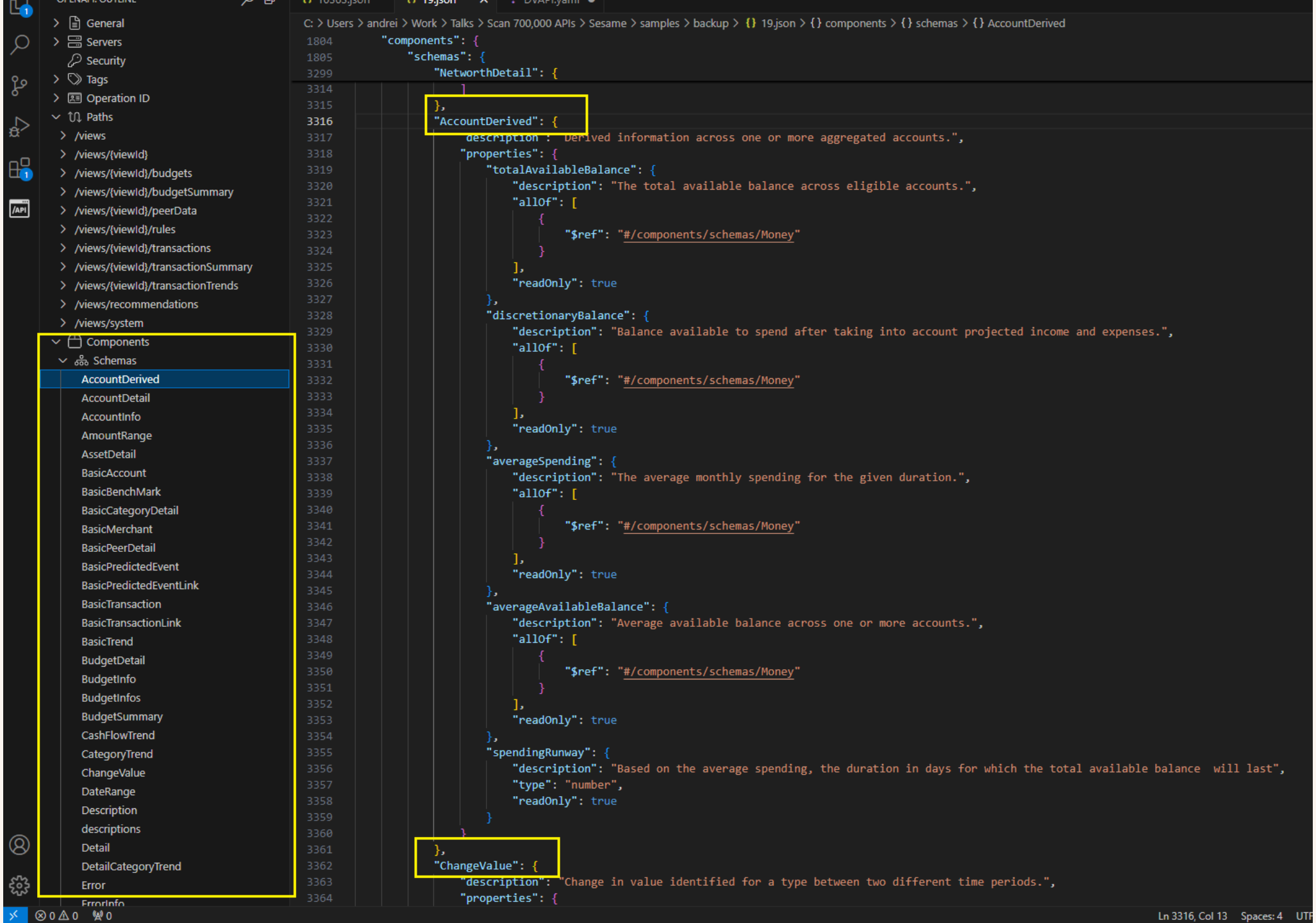

```
            "records":[
                {
                    "uid":"path_book",
                    "name": "books"
                },
                {
                    "uid":"path_author",
                    "name":"authors"
                }
            ]
        },
        {
            "labels":["Object"],
            "key": "gid",
            "records":[
                {
                    "gid":"obj_book",
                    "name": "book"
                },
                {
                    "gid":"obj_author",
                    "name": "author"
                }
            ]
        },
        {
            "labels":["Field"],
            "key": "gid",
            "records":[
                {
                    "gid":"field_fname",
                    "name": "firstname"
                },
                {
                    "gid":"field_lname",
                    "name": "lastname"
                }
            ]
        }
    ],
    "relationships": [
        {
            "type":"read",
            "from_node": {
                "record_key":"_from_uid",
                "node_key":"uid",
                "node_label":"Verb"
            },
            "to_node": {
                "record_key":"_to_uid",
                "node_key":"uid",
                "node_label": "Path"
            },
            "exclude_keys":["_from_uid", "_to_uid"],
            "records":[
                {
                    "_from_uid":"verb_get",
                    "_to_uid":"path_book"
                }
            ]
        }
```

Line 85, Column 27

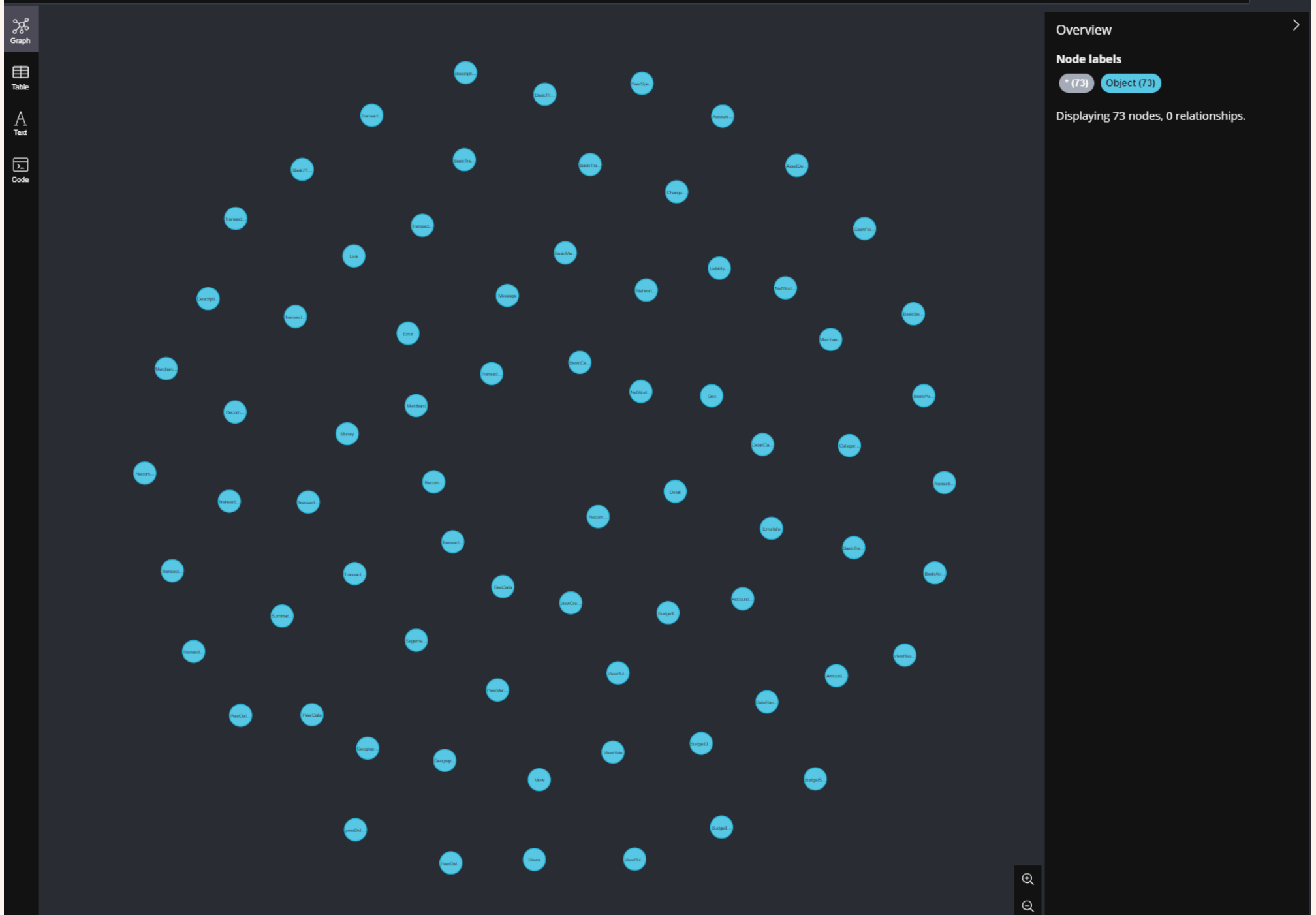

books
create
read
POST
GET
create
read
book
has
has
title
author
has
has
firstname
lastname

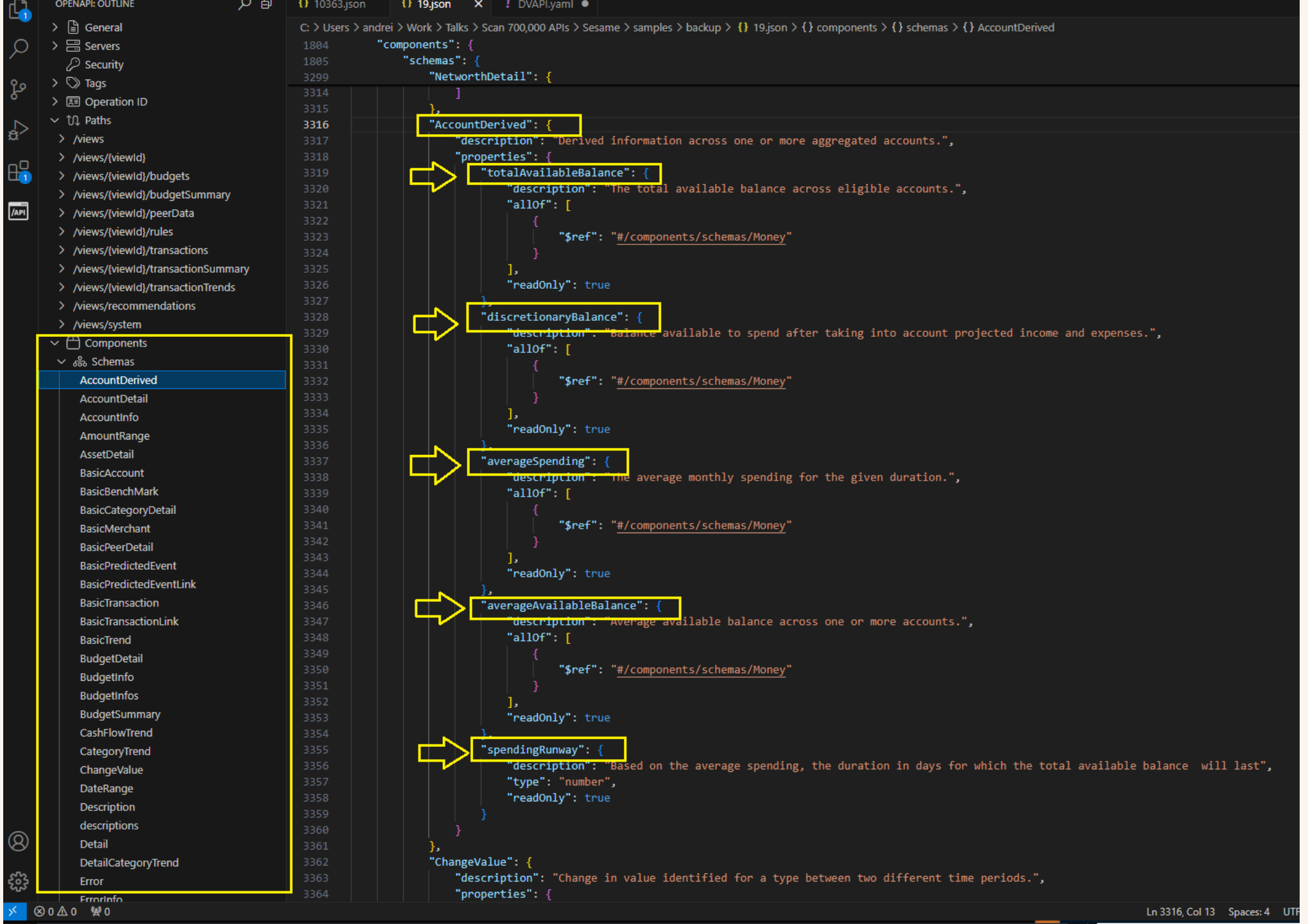

POST
GET
create
read
books
returns
book
has
has
author
title
has
has
firstname
lastname

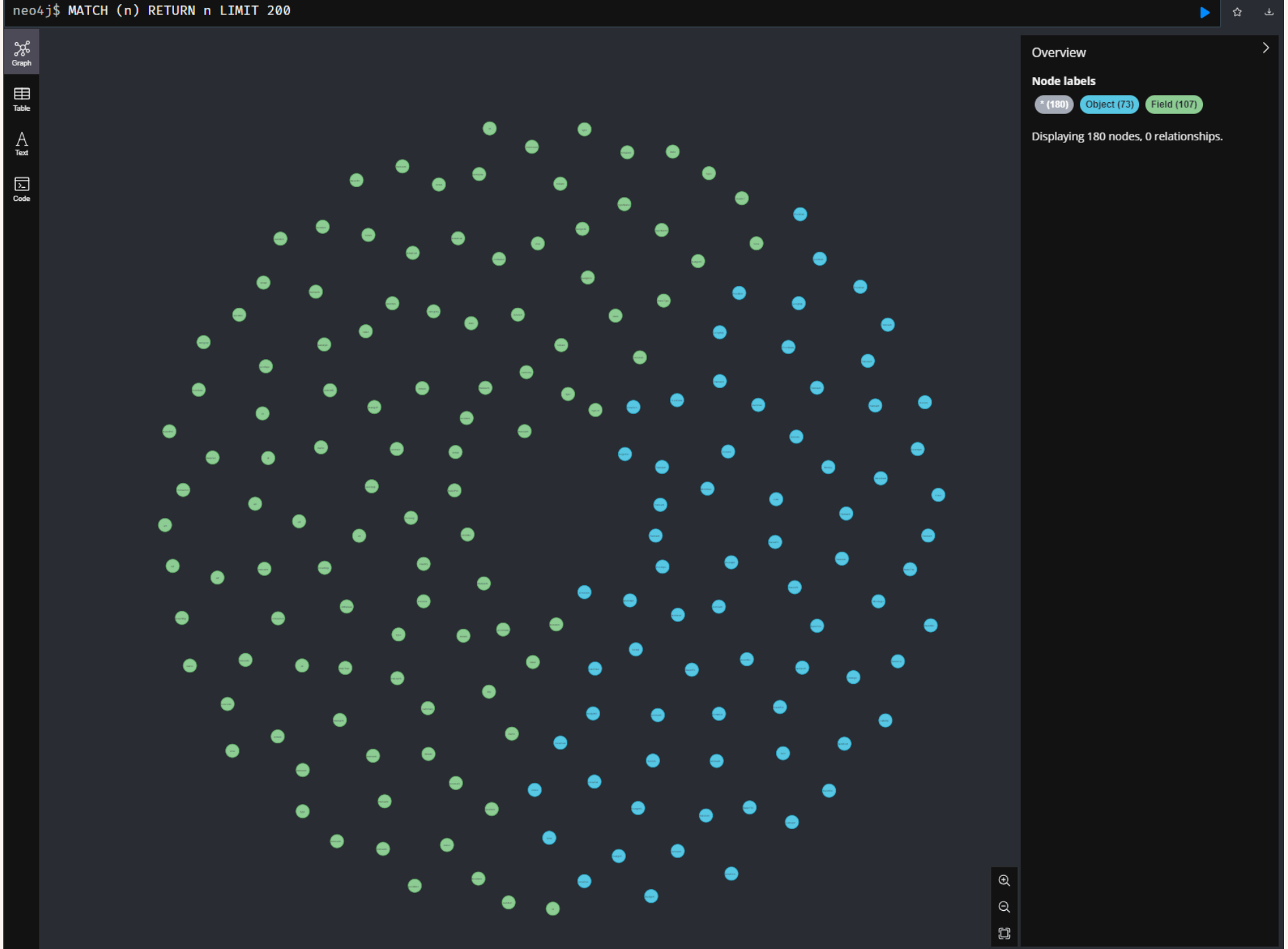
Field Name	Type	Description
openapi	`string`	**REQUIRED**. This string MUST be the semantic version number of the OpenAPI Specification version that the OpenAPI document uses. The `openapi` field SHOULD be used by tooling specifications and clients to interpret the OpenAPI document. This is *not* related to the API `info.version` string.
info	Info Object	**REQUIRED**. Provides metadata about the API. The metadata MAY be used by tooling as required.
servers	[Server Object]	An array of Server Objects, which provide connectivity information to a target server. If the `servers` property is not provided, or is an empty array, the default value would be a Server Object with a url value of `/`.
paths	Paths Object	**REQUIRED**. The available paths and operations for the API.
components	Components Object	An element to hold various schemas for the specification.
security	[Security Requirement Object]	A declaration of which security mechanisms can be used across the API. The list of values includes alternative security requirement objects that can be used. Only one of the security requirement objects need to be satisfied to authorize a request. Individual operations can override this definition. To make security optional, an empty security requirement (`{}`) can be included in the array.

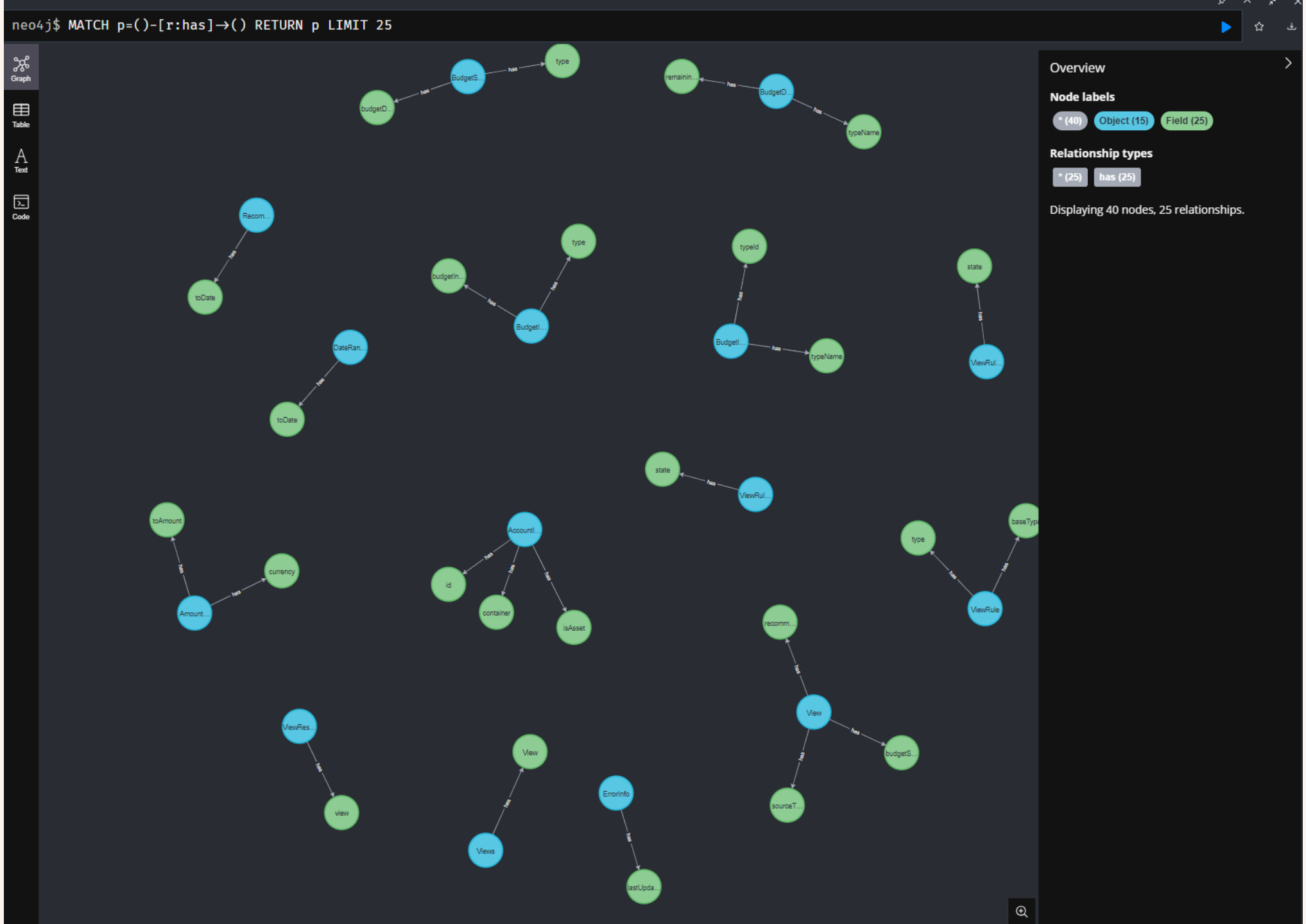

General
Servers
Security
Tags
Operation ID
Paths
/views
/views/{viewId}
/views/{viewId}/budgets
/views/{viewId}/budgetSummary
/views/{viewId}/peerData
/views/{viewId}/rules
/views/{viewId}/transactions
/views/{viewId}/transactionSummary
/views/{viewId}/transactionTrends
/views/recommendations
/views/system
Components
Schemas
AccountDerived
AccountDetail
AccountInfo
AmountRange
AssetDetail
BasicAccount
BasicBenchMark
BasicCategoryDetail
BasicMerchant
BasicPeerDetail
BasicPredictedEvent
BasicPredictedEventLink
BasicTransaction
BasicTransactionLink
BasicTrend
BudgetDetail
BudgetInfo
BudgetInfos
BudgetSummary
CashFlowTrend
CategoryTrend
ChangeValue
DateRange
Description
descriptions
Detail
DetailCategoryTrend
Error
C: > Users > andrei > Work > Talks > Scan 700,000 APIs > Sesame > samples > backup > {} 19.json > {} components > {} schemas > {} AccountDerived
"components": {
"schemas": {
"NetworthDetail": {
},
"AccountDerived": {
"description": "Derived information across one or more aggregated accounts.",
"properties": {
"totalAvailableBalance": {
"description": "The total available balance across eligible accounts.",
"allOf": [
{
"$ref": "#/components/schemas/Money"
}
],
"readOnly": true
},
"discretionaryBalance": {
"description": "Balance available to spend after taking into account projected income and expenses.",
"allOf": [
{
"$ref": "#/components/schemas/Money"
}
],
"readOnly": true
},
"averageSpending": {
"description": "The average monthly spending for the given duration.",
"allOf": [
{
"$ref": "#/components/schemas/Money"
}
],
"readOnly": true
},
"averageAvailableBalance": {
"description": "Average available balance across one or more accounts.",
"allOf": [
{
"$ref": "#/components/schemas/Money"
}
],
"readOnly": true
},
"spendingRunway": {
"description": "Based on the average spending, the duration in days for which the total available balance will last",
"type": "number",
"readOnly": true
}
}
},
"ChangeValue": {
"description": "Change in value identified for a type between two different time periods.",
"properties": {
Ln 3316, Col 13
Spaces: 4

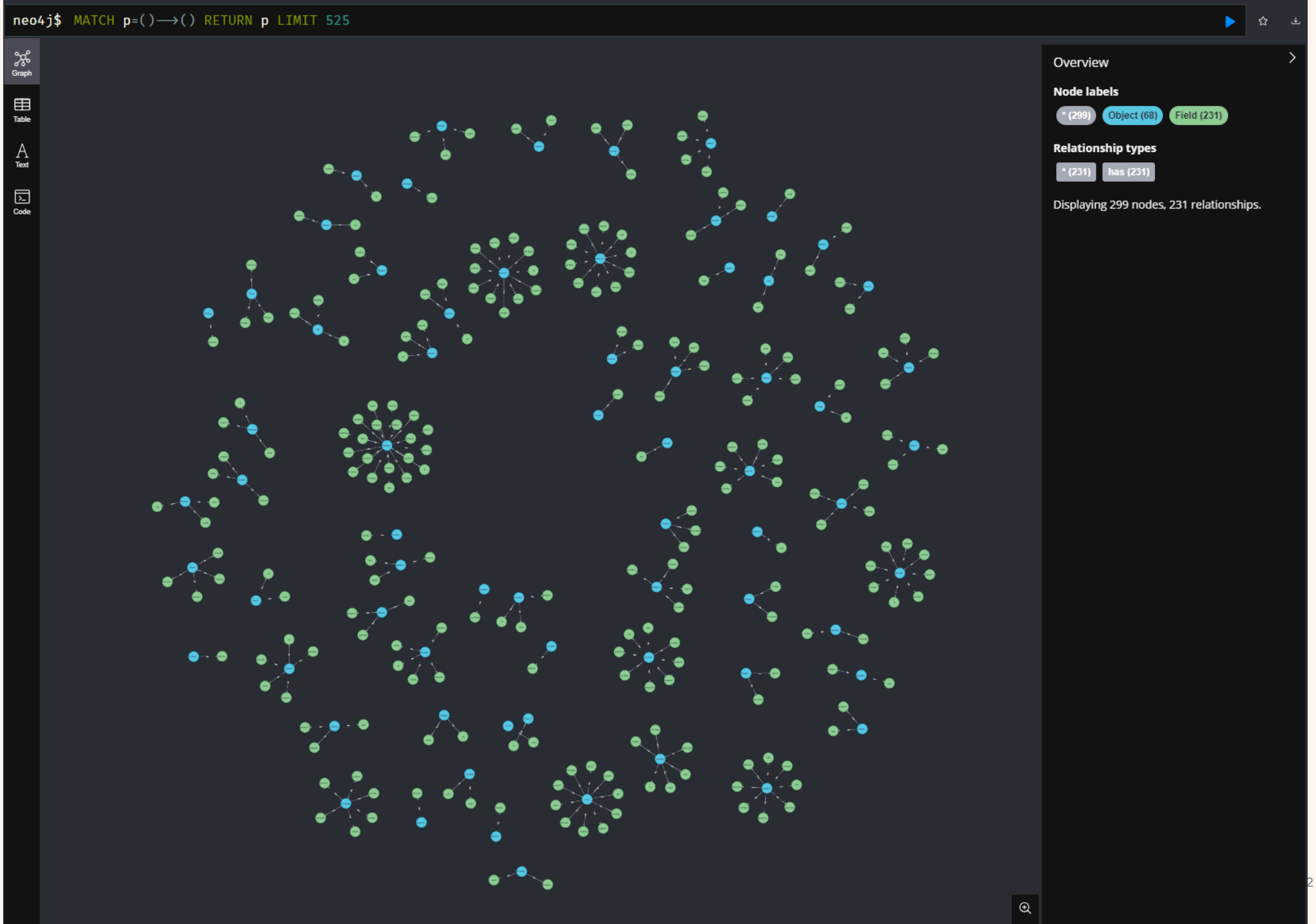
```
neo4j$ MATCH (n:Object) RETURN n LIMIT 100
```

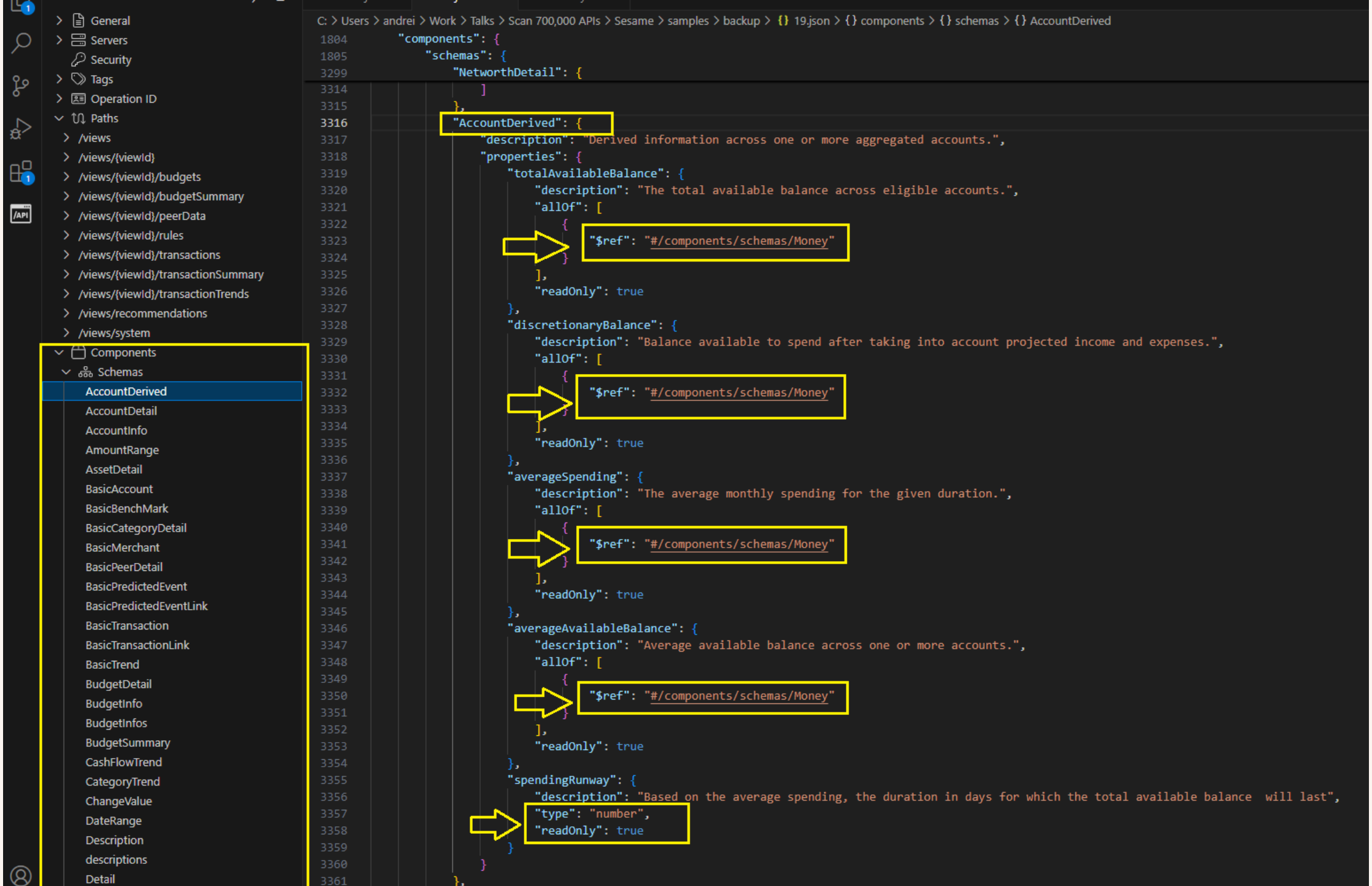


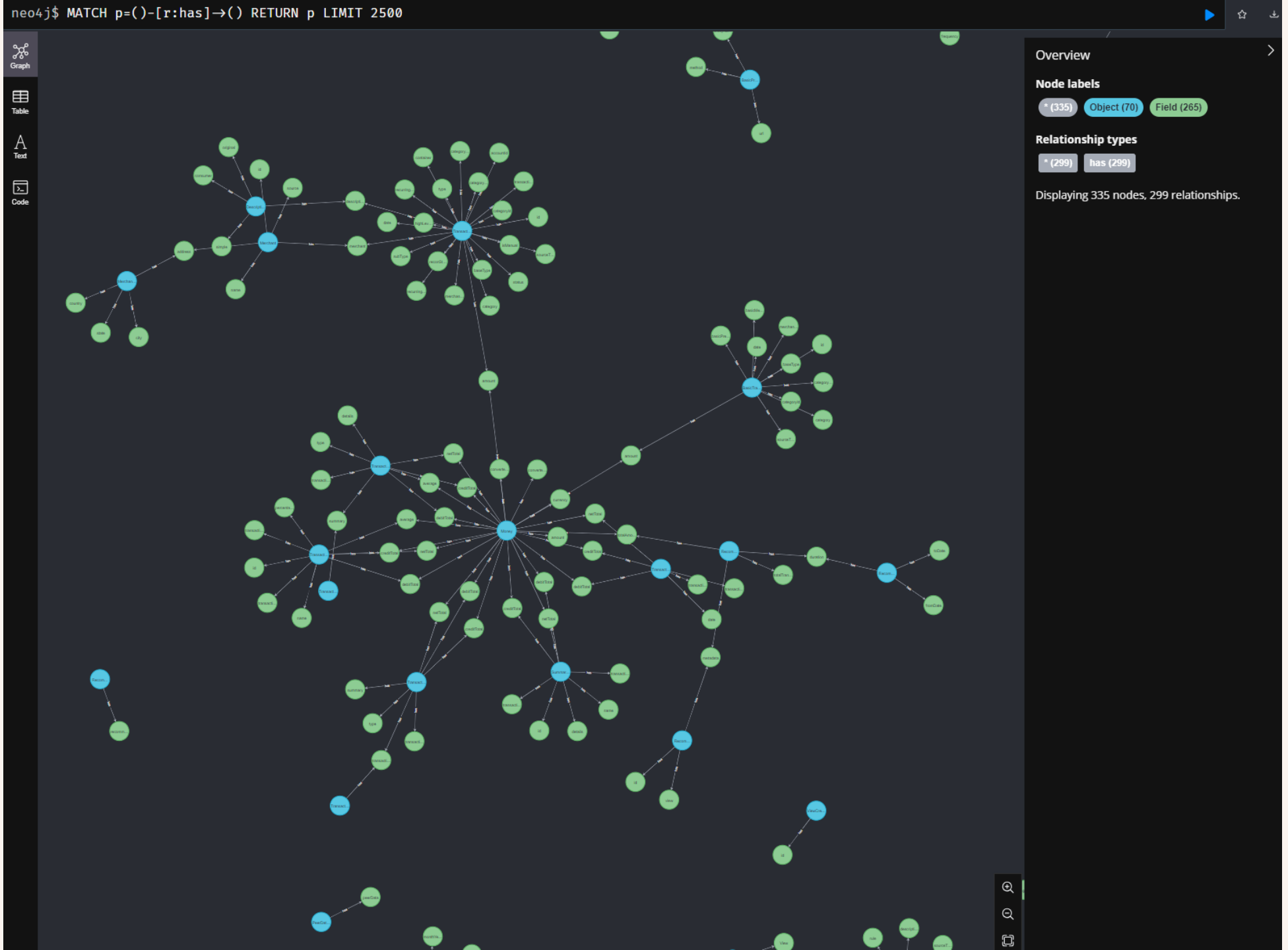

OPENAPI: OUTLINE
General
Servers
Security
Tags
Operation ID
Paths
/views
/views/{viewId}
/views/{viewId}/budgets
/views/{viewId}/budgetSummary
/views/{viewId}/peerData
/views/{viewId}/rules
/views/{viewId}/transactions
/views/{viewId}/transactionSummary
/views/{viewId}/transactionTrends
/views/recommendations
/views/system
Components
Schemas
AccountDerived
AccountDetail
AccountInfo
AmountRange
AssetDetail
BasicAccount
BasicBenchMark
BasicCategoryDetail
BasicMerchant
BasicPeerDetail
BasicPredictedEvent
BasicPredictedEventLink
BasicTransaction
BasicTransactionLink
BasicTrend
BudgetDetail
BudgetInfo
BudgetInfos
BudgetSummary
CashFlowTrend
CategoryTrend
ChangeValue
DateRange
Description
descriptions
Detail
DetailCategoryTrend
Error
10363.json
19.json
DVAPI.yaml
C: > Users > andrei > Work > Talks > Scan 700,000 APIs > Sesame > samples > backup > {} 19.json > {} components > {} schemas > {} AccountDerived
"components": {
"schemas": {
"NetworthDetail": {
"AccountDerived": {
"description": "Derived information across one or more aggregated accounts.",
"properties": {
"totalAvailableBalance": {
"description": "The total available balance across eligible accounts.",
"allOf": [
{
"$ref": "#/components/schemas/Money"
}
],
"readOnly": true
},
"discretionaryBalance": {
"description": "Balance available to spend after taking into account projected income and expenses.",
"allOf": [
{
"$ref": "#/components/schemas/Money"
}
],
"readOnly": true
},
"averageSpending": {
"description": "The average monthly spending for the given duration.",
"allOf": [
{
"$ref": "#/components/schemas/Money"
}
],
"readOnly": true
},
"averageAvailableBalance": {
"description": "Average available balance across one or more accounts.",
"allOf": [
{
"$ref": "#/components/schemas/Money"
}
],
"readOnly": true
},
"spendingRunway": {
"description": "Based on the average spending, the duration in days for which the total available balance will last",
"type": "number",
"readOnly": true
}
}
},
"ChangeValue": {
"description": "Change in value identified for a type between two different time periods.",
"properties": {
Ln 3316, Col 13
Spaces: 4
UTF


neo4j$ MATCH (n) RETURN n LIMIT 200
Graph
Table
Text
Code
Overview
Node labels
* (180)
Object (73)
Field (107)
Displaying 180 nodes, 0 relationships.

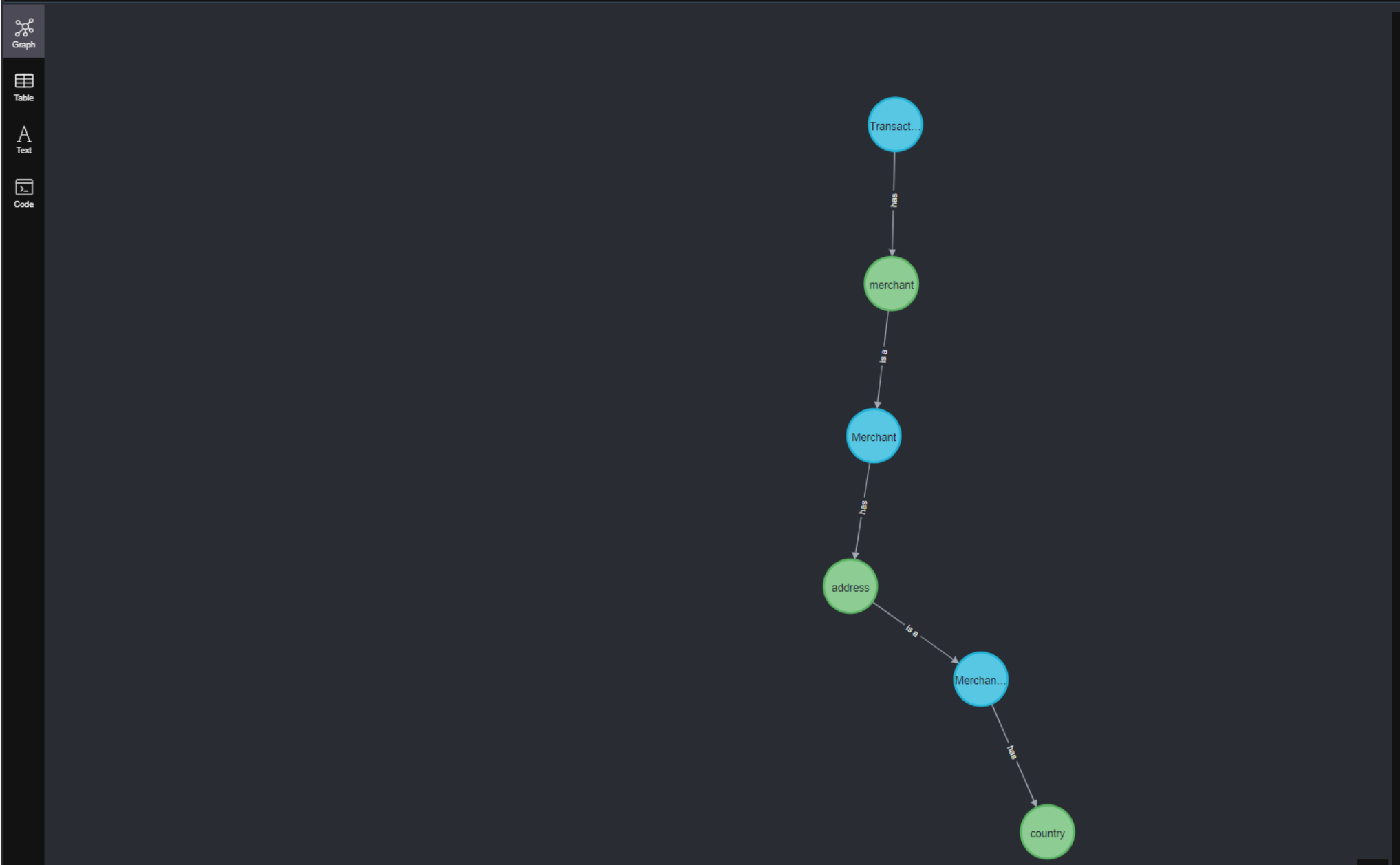

neo4j$ MATCH p=()-[r:has]→() RETURN p LIMIT 25
Graph
Table
Text
Code
Overview
Node labels
* (40)
Object (15)
Field (25)
Relationship types
* (25)
has (25)
Displaying 40 nodes, 25 relationships.
type
BudgetS...
budgetD...
has
remainin...
BudgetD...
typeName
Recom...
toDate
budgetIn...
BudgetI...
typeId
state
ViewRul...
DateRan...
toAmount
currency
Amount...
AccountI...
id
container
isAsset
baseType
type
ViewRule
recomm...
View
budgetS...
sourceT...
ViewRes...
view
Views
ErrorInfo
lastUpda...

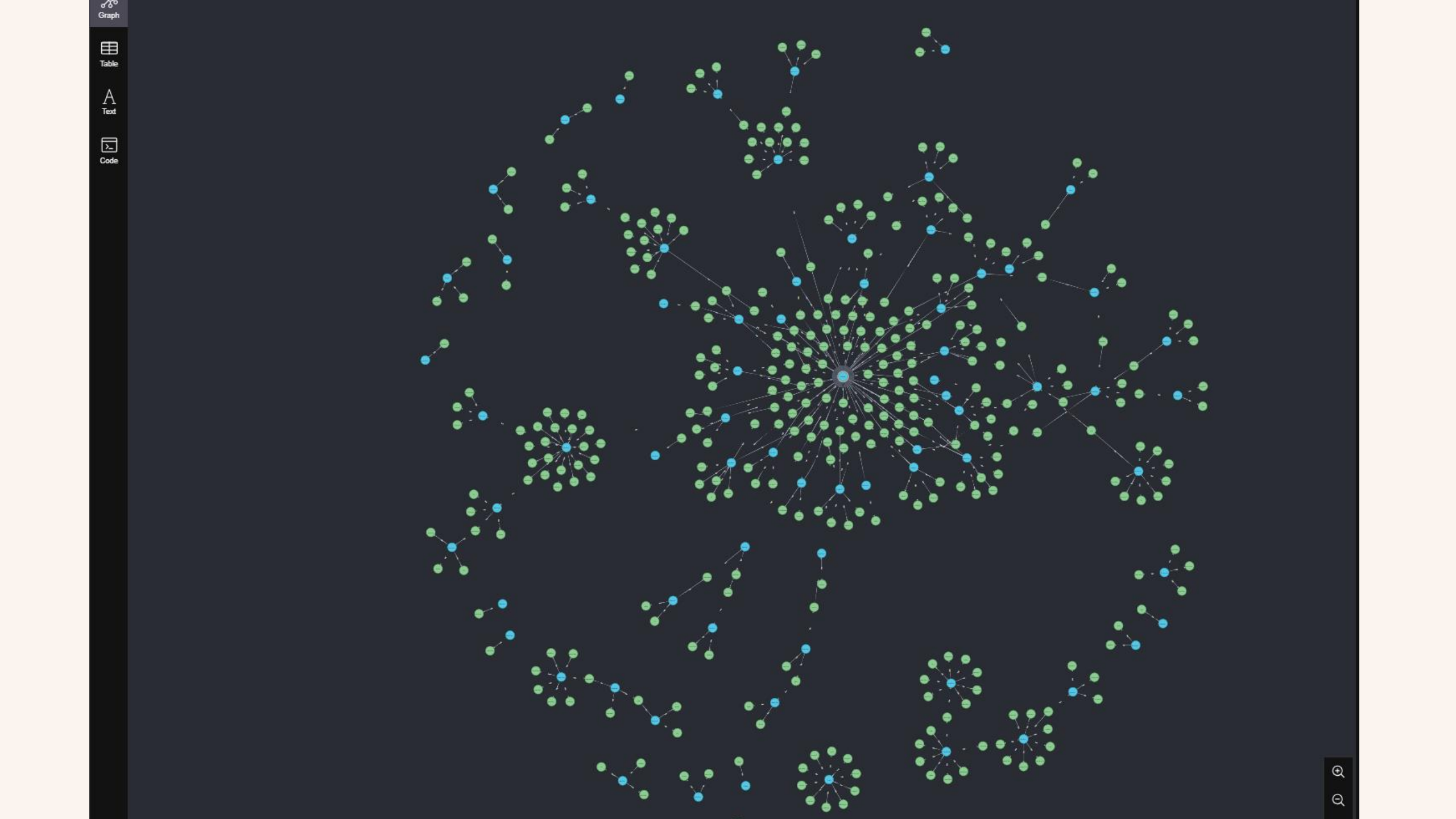

neo4j$ MATCH p=()⟶() RETURN p LIMIT 525
Graph
Table
Text
Code
Overview
Node labels
* (299)
Object (68)
Field (231)
Relationship types
* (231)
has (231)
Displaying 299 nodes, 231 relationships.


General
Servers
Security
Tags
Operation ID
Paths
/views
/views/{viewId}
/views/{viewId}/budgets
/views/{viewId}/budgetSummary
/views/{viewId}/peerData
/views/{viewId}/rules
/views/{viewId}/transactions
/views/{viewId}/transactionSummary
/views/{viewId}/transactionTrends
/views/recommendations
/views/system
Components
Schemas
AccountDerived
AccountDetail
AccountInfo
AmountRange
AssetDetail
BasicAccount
BasicBenchMark
BasicCategoryDetail
BasicMerchant
BasicPeerDetail
BasicPredictedEvent
BasicPredictedEventLink
BasicTransaction
BasicTransactionLink
BasicTrend
BudgetDetail
BudgetInfo
BudgetInfos
BudgetSummary
CashFlowTrend
CategoryTrend
ChangeValue
DateRange
Description
descriptions
Detail
C: > Users > andrei > Work > Talks > Scan 700,000 APIs > Sesame > samples > backup > {} 19.json > {} components > {} schemas > {} AccountDerived
"components": {
"schemas": {
"NetworthDetail": {
"AccountDerived": {
"description": "Derived information across one or more aggregated accounts.",
"properties": {
"totalAvailableBalance": {
"description": "The total available balance across eligible accounts.",
"allOf": [
"$ref": "#/components/schemas/Money"
"readOnly": true
"discretionaryBalance": {
"description": "Balance available to spend after taking into account projected income and expenses.",
"allOf": [
"$ref": "#/components/schemas/Money"
"readOnly": true
"averageSpending": {
"description": "The average monthly spending for the given duration.",
"allOf": [
"$ref": "#/components/schemas/Money"
"readOnly": true
"averageAvailableBalance": {
"description": "Average available balance across one or more accounts.",
"allOf": [
"$ref": "#/components/schemas/Money"
"readOnly": true
"spendingRunway": {
"description": "Based on the average spending, the duration in days for which the total available balance will last",
"type": "number",
"readOnly": true

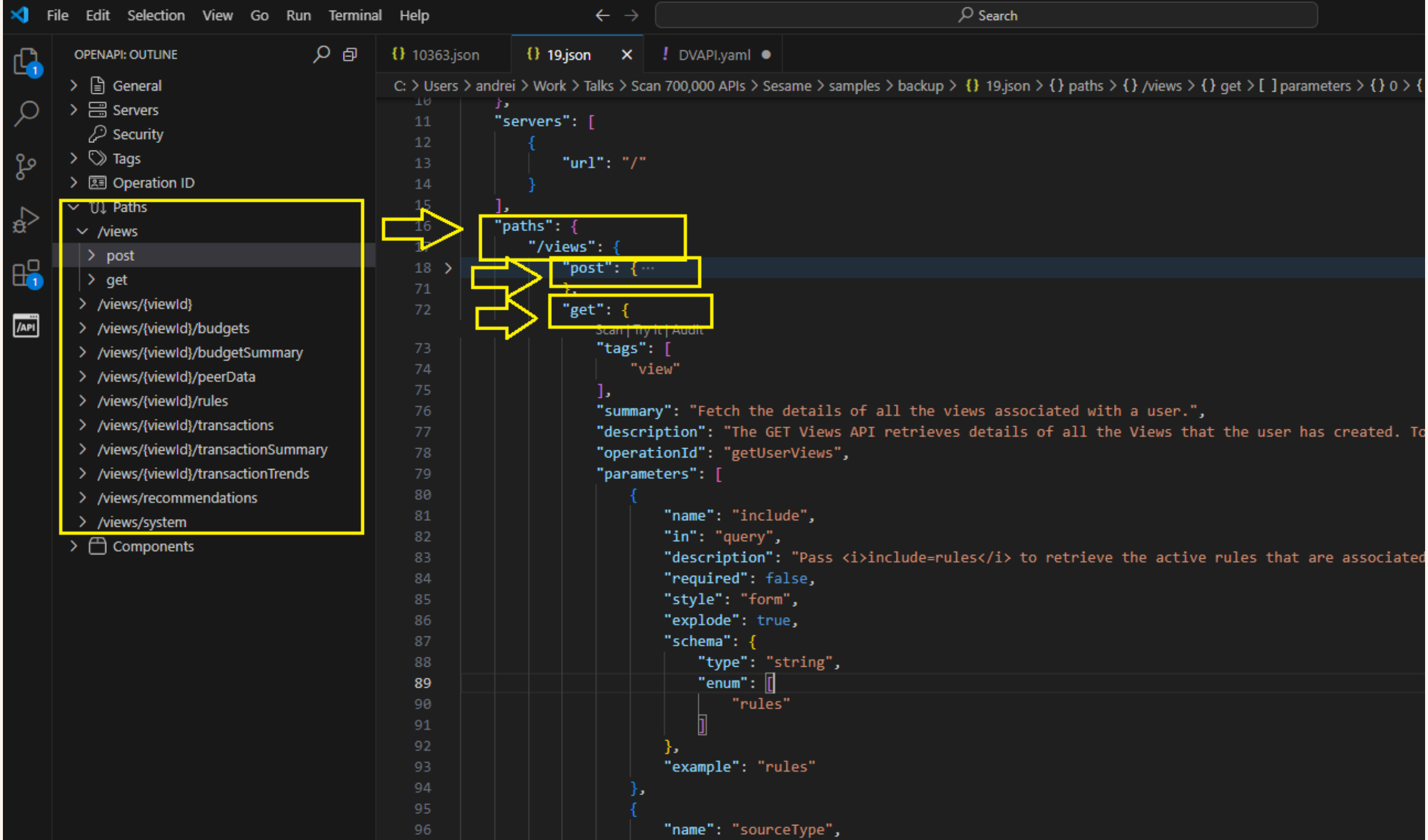

neo4j$ MATCH p=()-[r:has]→() RETURN p LIMIT 2500
Graph
Table
Text
Code
Overview
Node labels
* (335)
Object (70)
Field (265)
Relationship types
* (299)
has (299)
Displaying 335 nodes, 299 relationships.

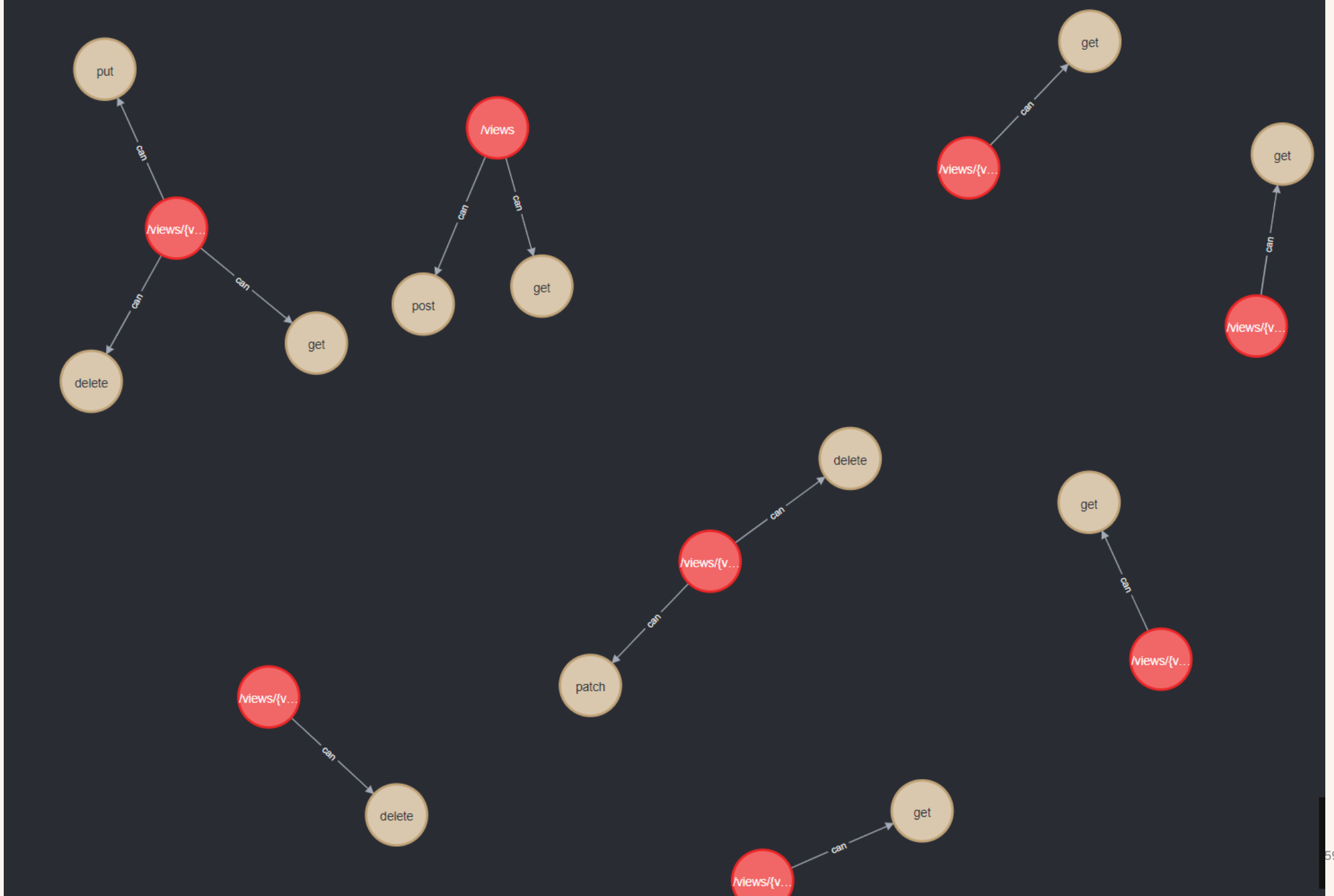
```
neo4j$ MATCH path = (start {uid: "Transaction_object"})-[*1..5]-(end {uid: "MerchantAddress_string_country"}) RETURN path
```



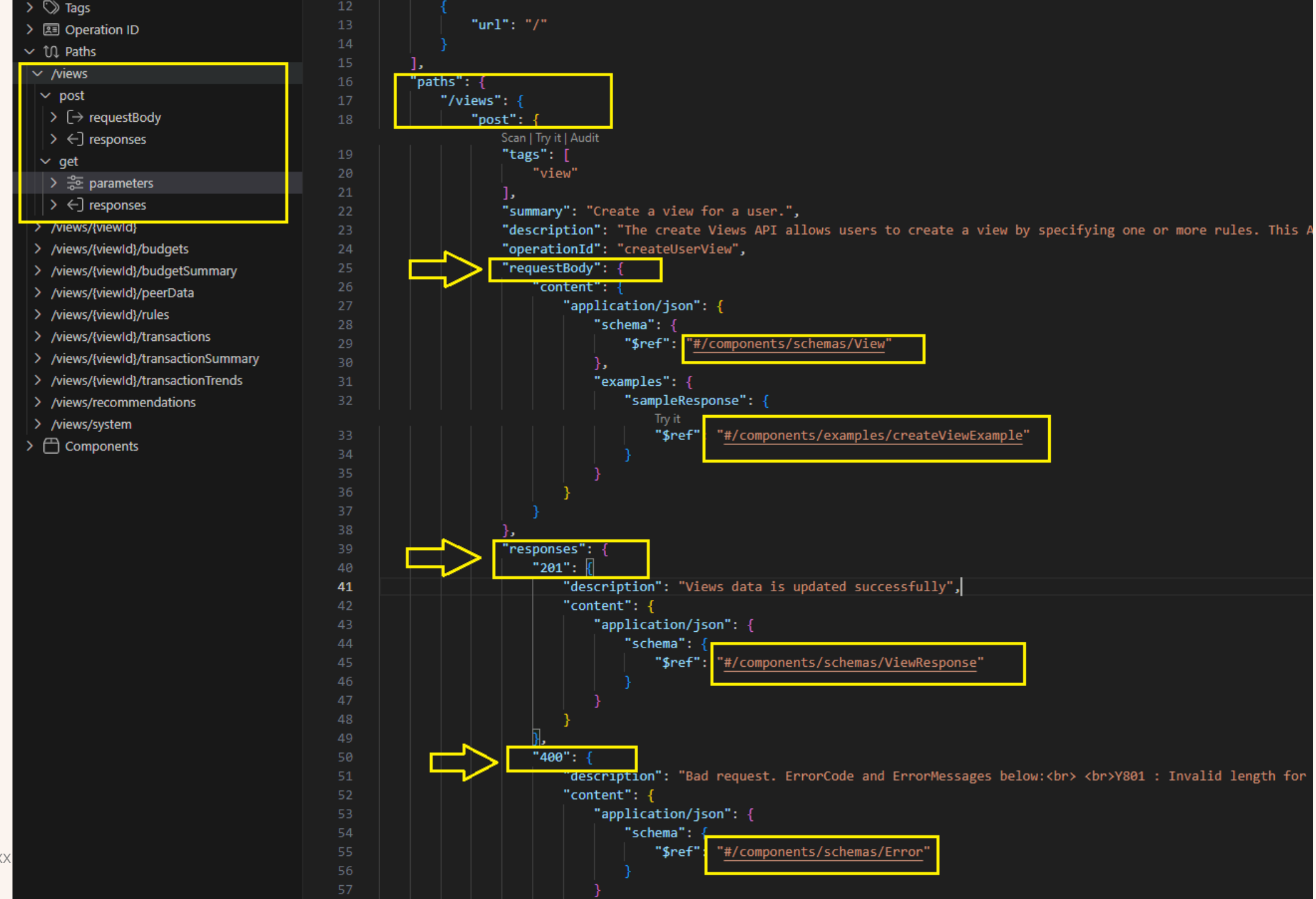

Graph
Table
Text
Code

Field Name	Type	Description
openapi	`string`	**REQUIRED**. This string MUST be the semantic version number of the OpenAPI Specification version that the OpenAPI document uses. The `openapi` field SHOULD be used by tooling specifications and clients to interpret the OpenAPI document. This is *not* related to the API `info.version` string.
info	Info Object	**REQUIRED**. Provides metadata about the API. The metadata MAY be used by tooling as required.
servers	[Server Object]	An array of Server Objects, which provide connectivity information to a target server. If the `servers` property is not provided, or is an empty array, the default value would be a Server Object with a url value of `/`.
paths	Paths Object	**REQUIRED**. The available paths and operations for the API.
components	Components Object	An element to hold various schemas for the specification.
security	[Security Requirement Object]	A declaration of which security mechanisms can be used across the API. The list of values includes alternative security requirement objects that can be used. Only one of the security requirement objects need to be satisfied to authorize a request. Individual operations can override this definition. To make security optional, an empty security requirement (`{}`) can be included in the array.


File Edit Selection View Go Run Terminal Help
Search
OPENAPI: OUTLINE
General
Servers
Security
Tags
Operation ID
Paths
/views
post
get
/views/{viewId}
/views/{viewId}/budgets
/views/{viewId}/budgetSummary
/views/{viewId}/peerData
/views/{viewId}/rules
/views/{viewId}/transactions
/views/{viewId}/transactionSummary
/views/{viewId}/transactionTrends
/views/recommendations
/views/system
Components
10363.json
19.json
DVAPI.yaml
C: > Users > andrei > Work > Talks > Scan 700,000 APIs > Sesame > samples > backup > 19.json > {} paths > {} /views > {} get > [ ] parameters > {} 0 > {
"servers": [
"url": "/"
"paths": {
"/views": {
"post": {
"get": {
Scan | Try it | Audit
"tags": [
"view"
"summary": "Fetch the details of all the views associated with a user.",
"description": "The GET Views API retrieves details of all the Views that the user has created. To
"operationId": "getUserViews",
"parameters": [
"name": "include",
"in": "query",
"description": "Pass include=rules to retrieve the active rules that are associated
"required": false,
"style": "form",
"explode": true,
"schema": {
"type": "string",
"enum": [
"rules"
"example": "rules"
"name": "sourceType",


put
can
/views/{v...
can
can
delete
get
/views
can
can
post
get
get
can
/views/{v...
get
can
/views/{v...
delete
can
/views/{v...
patch
get
can
/views/{v...
/views/{v...
can
delete
get
can
/views/{v...

```
65822.json  44383.json  44394.json  44497.json  44545.json  51171.json  rest-api.yml  sesame_iboa.py  parse_paths.py  sesame_neo4j.py  sesame_ididor.py  - algolia_api_ke

            if True:
                json_content = json.load(my_file)
                schemas = json_content["components"]["schemas"]
                paths = json_content["paths"]
                ## Parse the paths
                for path in paths:
                    path_record = {}
                    path_record["uid"] = "path_" + path
                    path_record["name"] = path
                    path_node["records"].append(path_record)
                    for path_verb in paths[path]:
                        print("--------")
                        print(path_verb + " " + path)
                        verb_record = {}
                        verb_record["uid"] = path_verb + "_" + path
                        verb_record["name"] = path_verb
                        verb_node["records"].append(verb_record)
                        #link paths with verbs
                        relationship = {}
                        relationship["_from_uid"] = path_record["uid"]
                        relationship["_to_uid"] = verb_record["uid"]
                        path_verb_relationship["records"].append(relationship)
                        if "parameters" in paths[path][path_verb]:
                            for parameter in paths[path][path_verb]["parameters"]:
                                if "type" in parameter:
                                    type_t = parameter["type"]
                                    parameter_record = {}
                                    parameter_record["uid"] = path_verb + "_" + path + "_parameter_" + type_t + "_" + parameter["name"]
                                    parameter_record["name"] =  type_t + "_" + parameter["name"]
                                    print(parameter_record)
                                elif "schema" in parameter:
                                    type_t = parameter["schema"]["type"]
                                    parameter_record = {}
                                    parameter_record["uid"] = path_verb + "_" + path + "_parameter_" + type_t + "_" + parameter["name"]
                                    parameter_record["name"] =  type_t + " " + parameter["name"]
                                    print(parameter_record)
                                else:
                                    print("Can't find the type of parameter")
                                    print(parameter["schema"]["type"] + " " + parameter["name"])
                        if "responses" in paths[path][path_verb]:
                            print("response exists")
                            for response in paths[path][path_verb]["responses"]:
                                print("response: " + response)
                                if "content" in paths[path][path_verb]["responses"][response]:
                                    for c in paths[path][path_verb]["responses"][response]["content"]:
                                        cnt = paths[path][path_verb]["responses"][response]["content"][c]
                                        if "$ref" in cnt["schema"]:
                                            print("response type" + cnt["schema"]["$ref"])
                                        elif "type" in cnt["schema"]:
                                            print("response type" + cnt["schema"]["type"])
                                        else:
                                            print("Can't find the type of response")
                        if "requestBody" in paths[path][path_verb]:
                            print("requestBody exists")
                            for requestBody in paths[path][path_verb]["requestBody"]:
                                print("requestBody: " + requestBody)
                                if "content" in paths[path][path_verb]["requestBody"]:
                                    for c in paths[path][path_verb]["requestBody"]["content"]:
                                        cnt = paths[path][path_verb]["requestBody"]["content"][c]
                                        if "$ref" in cnt["schema"]:
                                            print("request body type" + cnt["schema"]["$ref"])
                                        elif "type" in cnt["schema"]:
                                            print("request body type" + cnt["schema"]["type"])
                                        else:
                                            print("Can't find the type of request body")
```

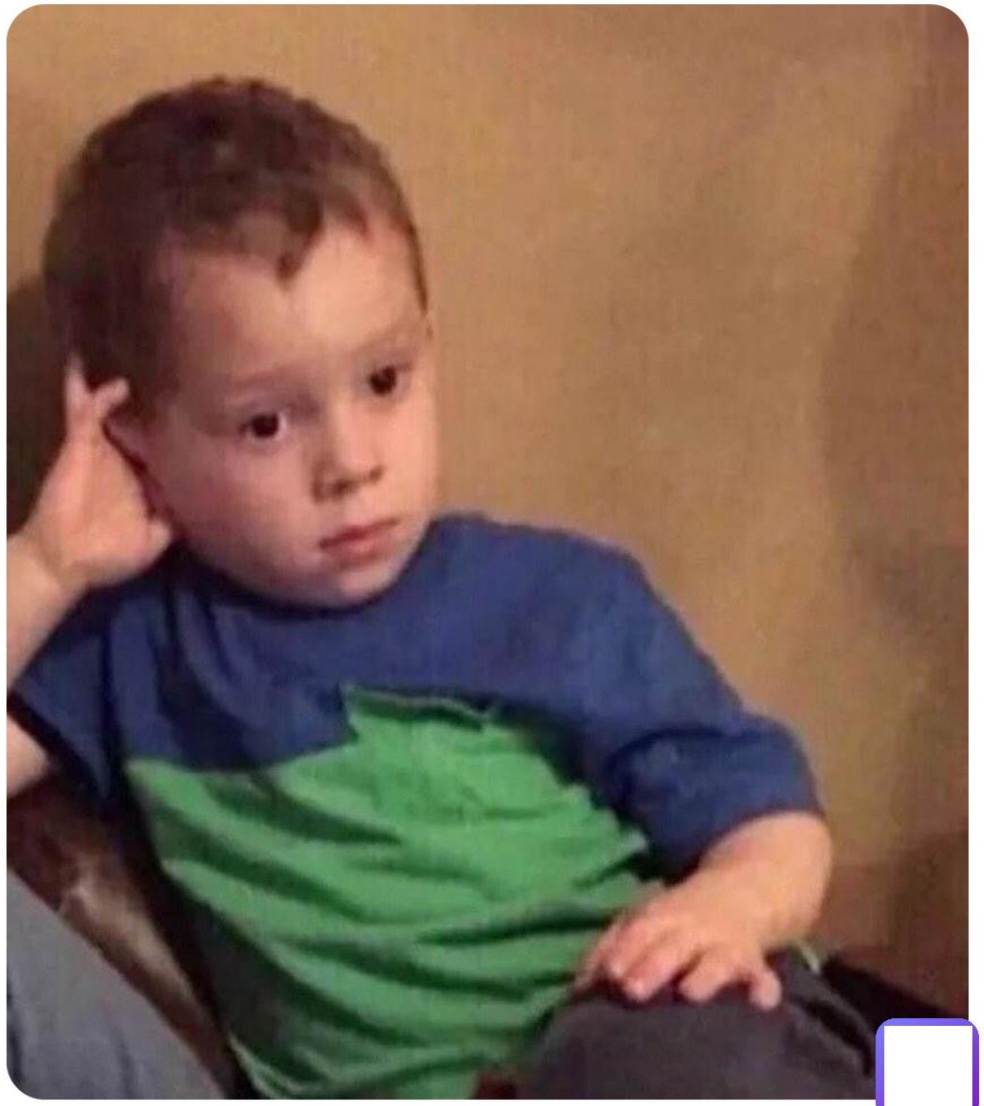

Tags
Operation ID
Paths
/views
post
requestBody
responses
get
parameters
responses
/views/{viewId}
/views/{viewId}/budgets
/views/{viewId}/budgetSummary
/views/{viewId}/peerData
/views/{viewId}/rules
/views/{viewId}/transactions
/views/{viewId}/transactionSummary
/views/{viewId}/transactionTrends
/views/recommendations
/views/system
Components
{
"url": "/"
}
],
"paths": {
"/views": {
"post": {
Scan | Try it | Audit
"tags": [
"view"
],
"summary": "Create a view for a user.",
"description": "The create Views API allows users to create a view by specifying one or more rules. This A
"operationId": "createUserView",
"requestBody": {
"content": {
"application/json": {
"schema": {
"$ref": "#/components/schemas/View"
},
"examples": {
"sampleResponse": {
Try it
"$ref": "#/components/examples/createViewExample"
}
}
}
}
},
"responses": {
"201": {
"description": "Views data is updated successfully",
"content": {
"application/json": {
"schema": {
"$ref": "#/components/schemas/ViewResponse"
}
}
}
},
"400": {
"description": "Bad request. ErrorCode and ErrorMessages below:

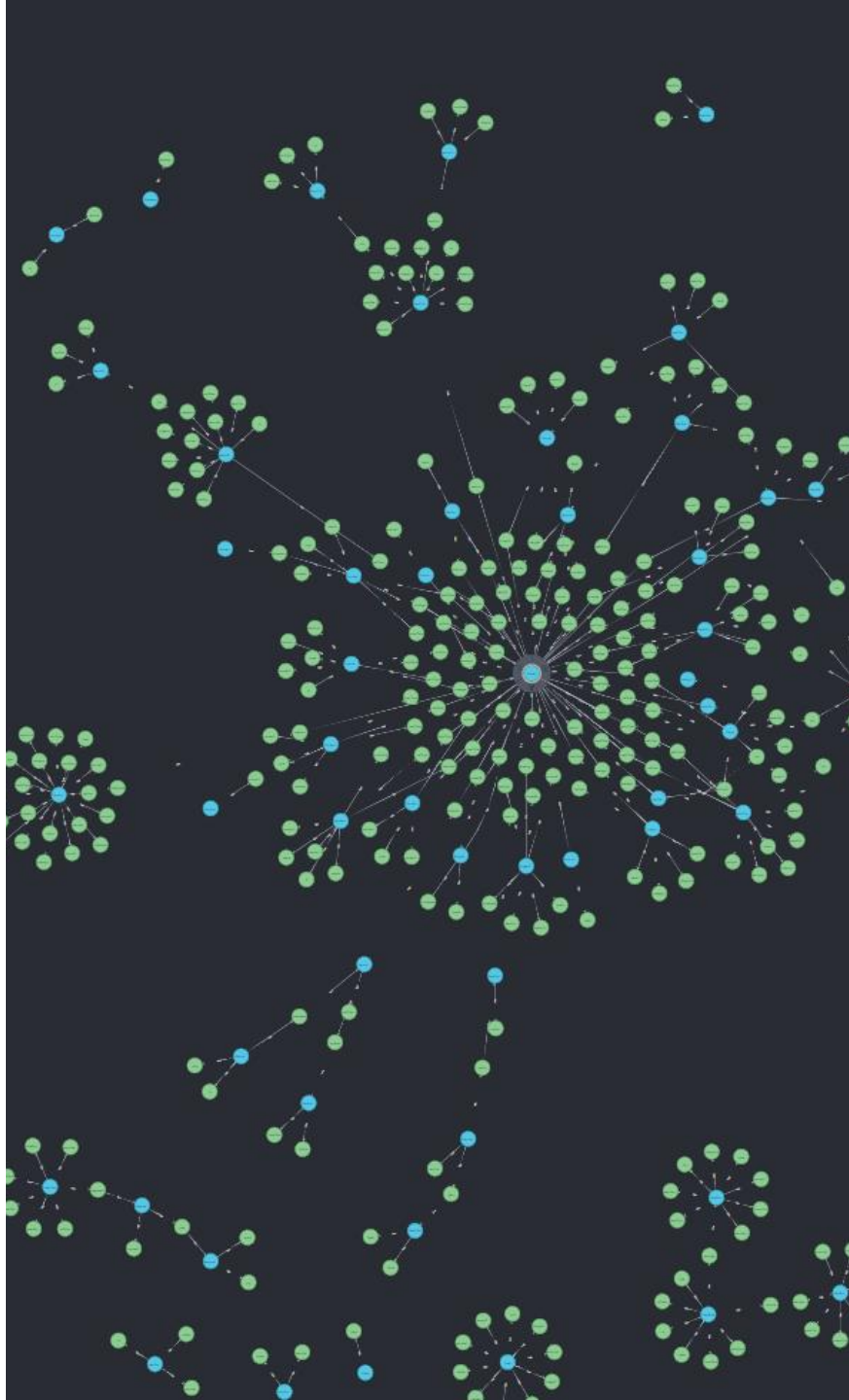
Y801 : Invalid length for
"content": {
"application/json": {
"schema": {
"$ref": "#/components/schemas/Error"
}
}

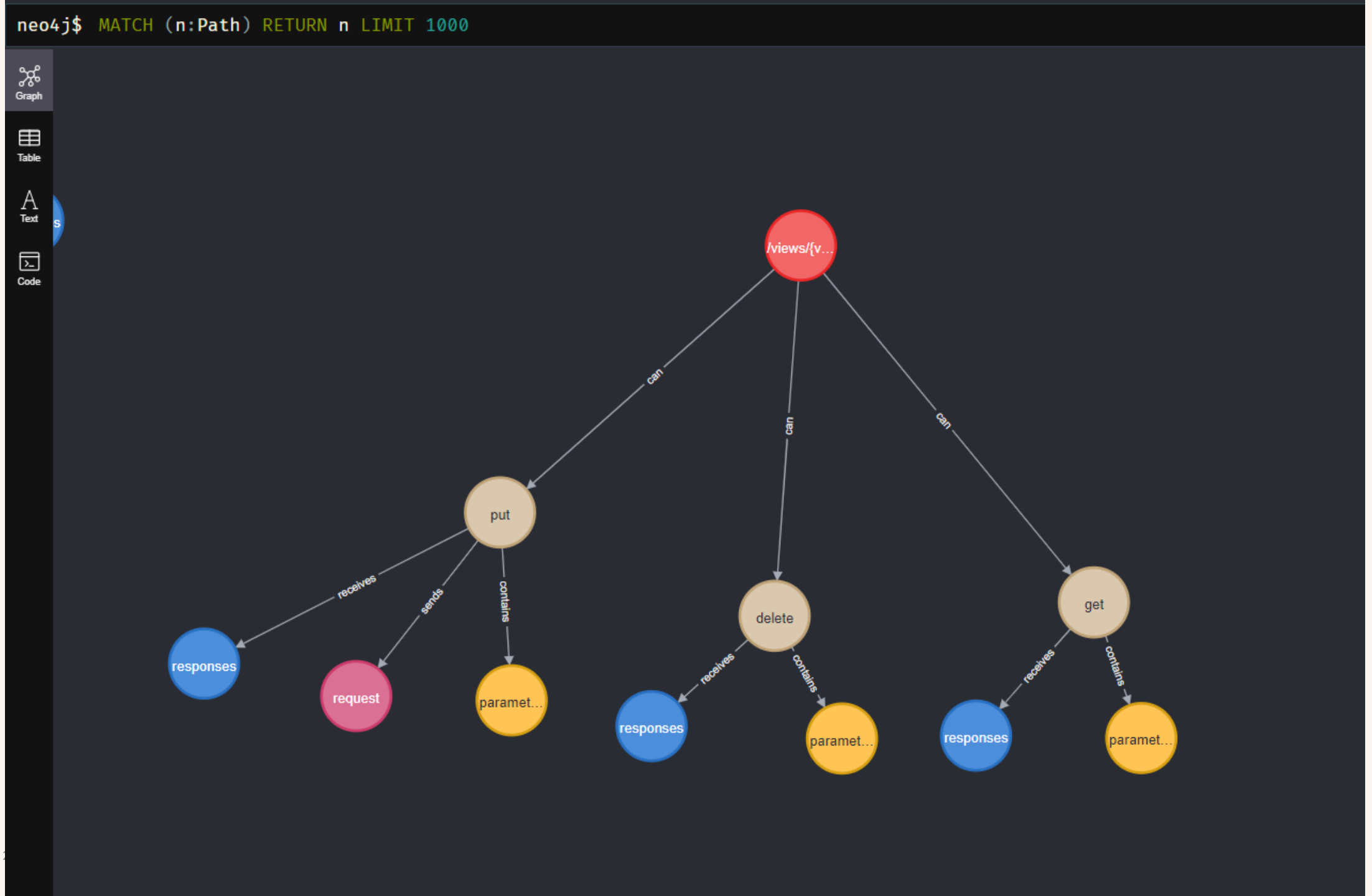
394.json × 44497.json × 44545.json × 51171.json × rest-api.yml × sesame_iboa

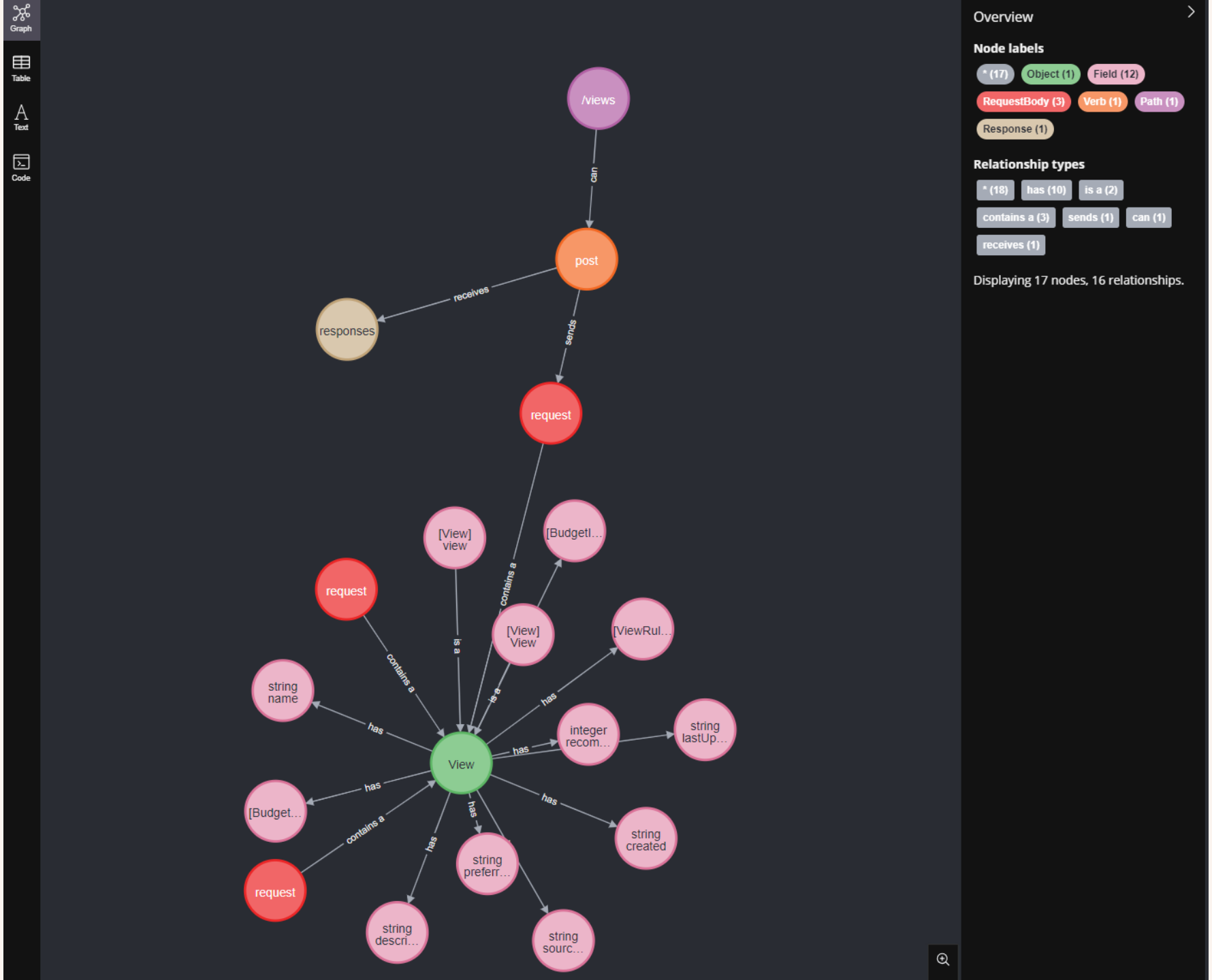
```
content = json.load(my_file)
as = json_content["components"]["schemas"]
 = json_content["paths"]
rse the paths
ath in paths:
ath_record = {}
ath_record["uid"] = "path_" + path
ath_record["name"] = path
ath_node["records"].append(path_record)
or path_verb in paths[path]:
    print("--------")
    print(path_verb + " " + path)
    verb_record = {}
    verb_record["uid"] = path_verb + "_" + path
    verb_record["name"] = path_verb
    verb_node["records"].append(verb_record)
    #link paths with verbs
    relationship = {}
    relationship["_from_uid"] = path_record["uid"]
    relationship["_to_uid"] = verb_record["uid"]
    path_verb_relationship["records"].append(relationship)
    if "parameters" in paths[path][path_verb]:
        for parameter in paths[path][path_verb]["parameters"]:
            if "type" in parameter:
                type_t = parameter["type"]
                parameter_record = {}
                parameter_record["uid"] = path_verb + "_" + path +
                parameter_record["name"] =  type_t + "_" + paramet
                print(parameter_record)
            elif "schema" in parameter:
                type_t = parameter["schema"]["type"]
                parameter_record = {}
                parameter_record["uid"] = path_verb + "_" + path +
                parameter_record["name"] =  type_t + " " + paramet
                print(parameter_record)
            else:
                print("Can't find the type of parameter")
            print(parameter["schema"]["type"] + " " + parameter["n
    if "responses" in paths[path][path_verb]:
        print("response exists")
        for response in paths[path][path_verb]["responses"]:
            print("response: " + response)
            if "content" in paths[path][path_verb]["responses"][re
                for c in paths[path][path_verb]["responses"][respo
                    cnt = paths[path][path_verb]["responses"][resp
                    if "$ref" in cnt["schema"]:
                        print("response type" + cnt["schema"]["$re
                    elif "type" in cnt["schema"]:
                        print("response type" + cnt["schema"]["typ
                    else:
                        print("Can't find the type of response")
    if "requestBody" in paths[path][path_verb]:
        print("requestBody exists")
        for requestBody in paths[path][path_verb]["requestBody"]:
            print("requestBody: " + requestBody)
            if "content" in paths[path][path_verb]["requestBody"]:
                for c in paths[path][path_verb]["requestBody"]["co
                    cnt = paths[path][path_verb]["requestBody"]["c
                    if "$ref" in cnt["schema"]:
                        print("request body type" + cnt["schema"][
                    elif "type" in cnt["schema"]:
                        print("request body type" + cnt["schema"][
                    else:
                        print("Can't find the type of request body
```

Me contemplating my life choices






neo4j$ MATCH (n:Path) RETURN n LIMIT 1000
Graph
Table
Text
Code
/views/{v...
can
can
can
put
delete
get
receives
sends
contains
receives
contains
receives
contains
responses
request
paramet...
responses
paramet...
responses
paramet...


Graph
Table
Text
Code
/views
can
post
receives
responses
sends
request
[View] view
[BudgetI...
request
contains a
[View] View
[ViewRul...
is a
string name
has
integer recom...
string lastUp...
View
has
[Budget...
contains a
request
has
string preferr...
string descri...
string sourc...
string created
Overview
Node labels
* (17)
Object (1)
Field (12)
RequestBody (3)
Verb (1)
Path (1)
Response (1)
Relationship types
* (18)
has (10)
is a (2)
contains a (3)
sends (1)
can (1)
receives (1)
Displaying 17 nodes, 16 relationships.

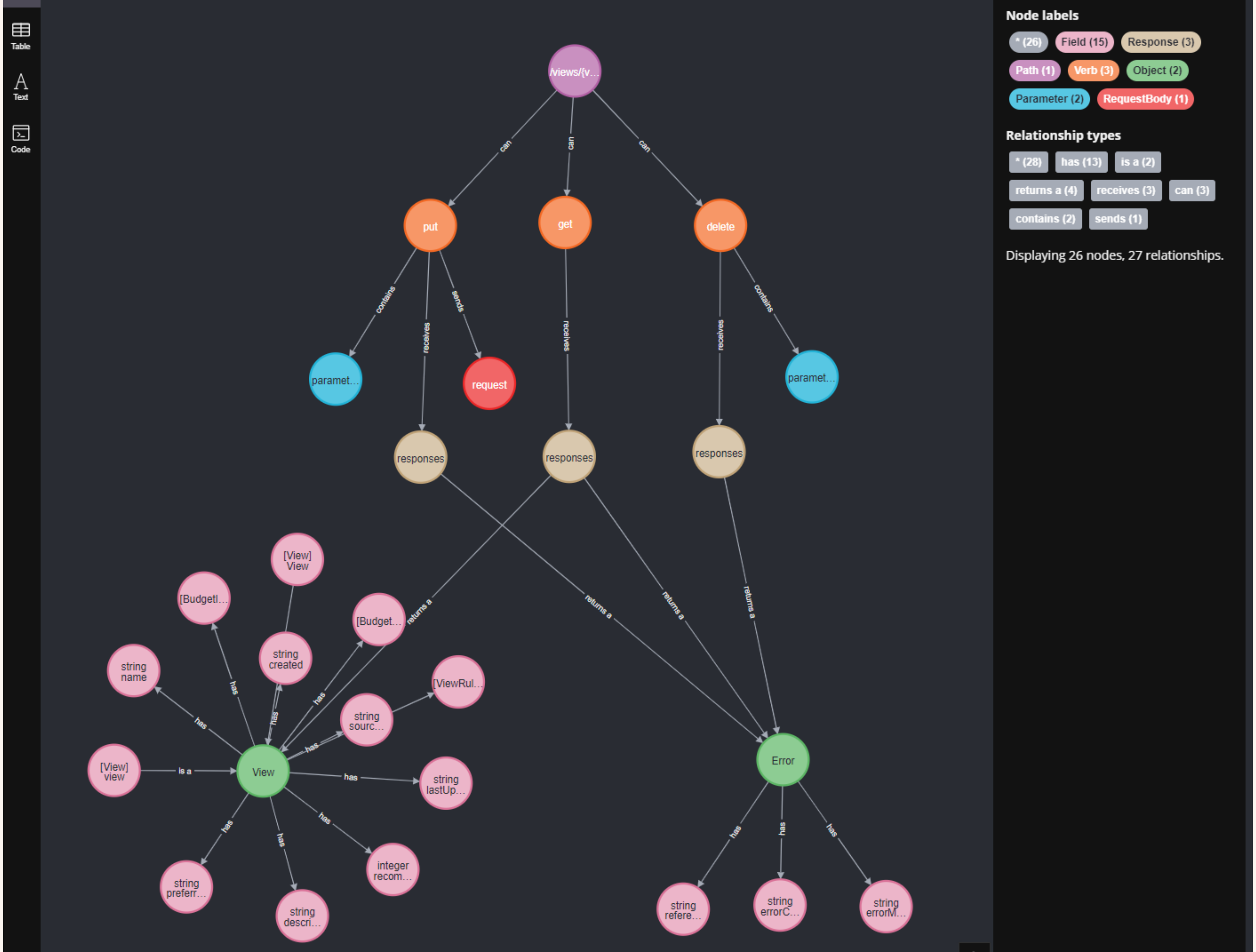


Table
Text
Code
/views/{v...
can
can
can
put
get
delete
contains
receives
sends
receives
receives
contains
paramet...
request
paramet...
responses
responses
responses
returns a
returns a
returns a
returns a
[View] View
[BudgetI...
[Budget...
string created
string name
[ViewRul...
string sourc...
[View] view
is a
View
has
string lastUp...
string preferr...
string descri...
integer recom...
Error
string refere...
string errorC...
string errorM...
Node labels
* (26)
Field (15)
Response (3)
Path (1)
Verb (3)
Object (2)
Parameter (2)
RequestBody (1)
Relationship types
* (28)
has (13)
is a (2)
returns a (4)
receives (3)
can (3)
contains (2)
sends (1)
Displaying 26 nodes, 27 relationships.

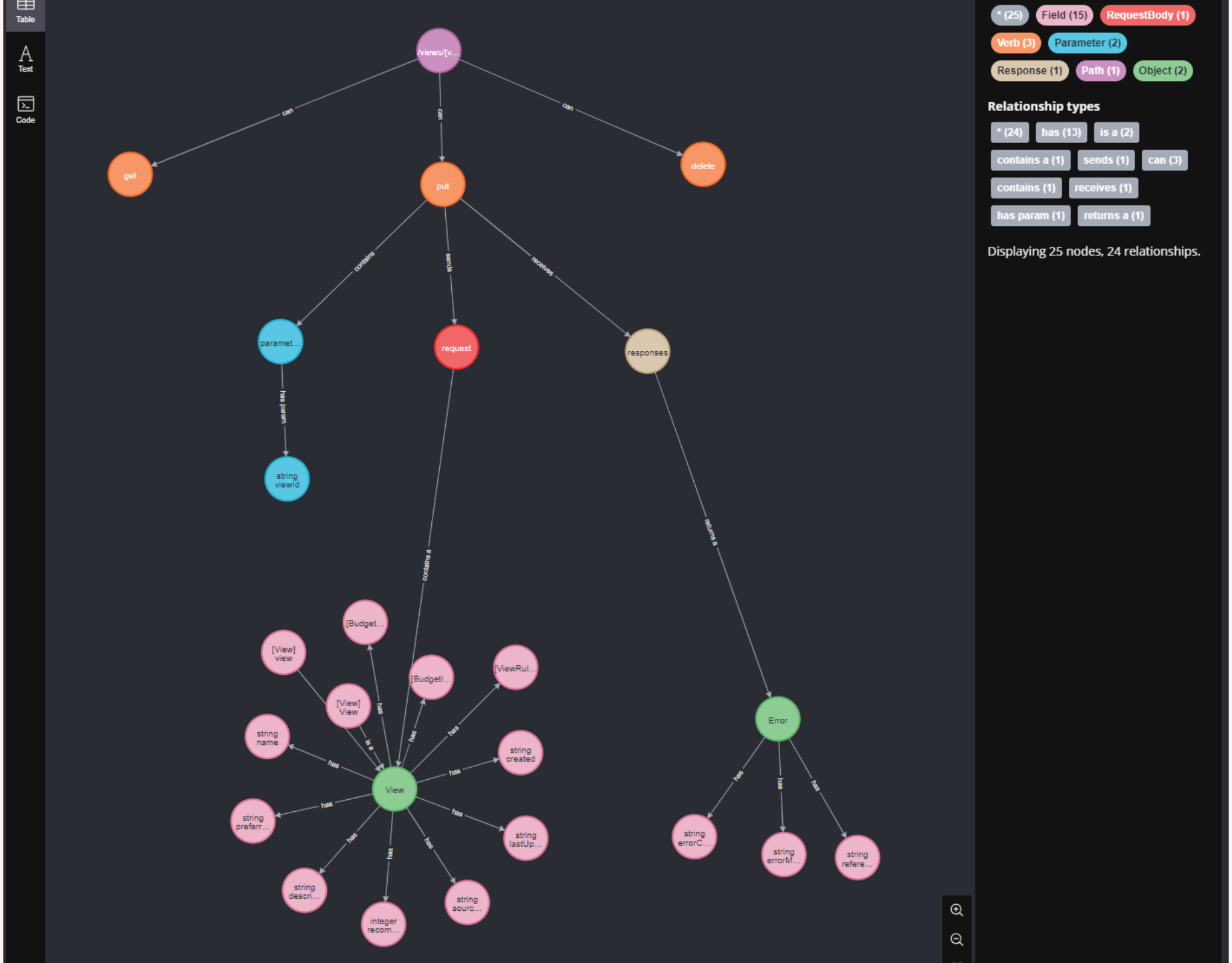
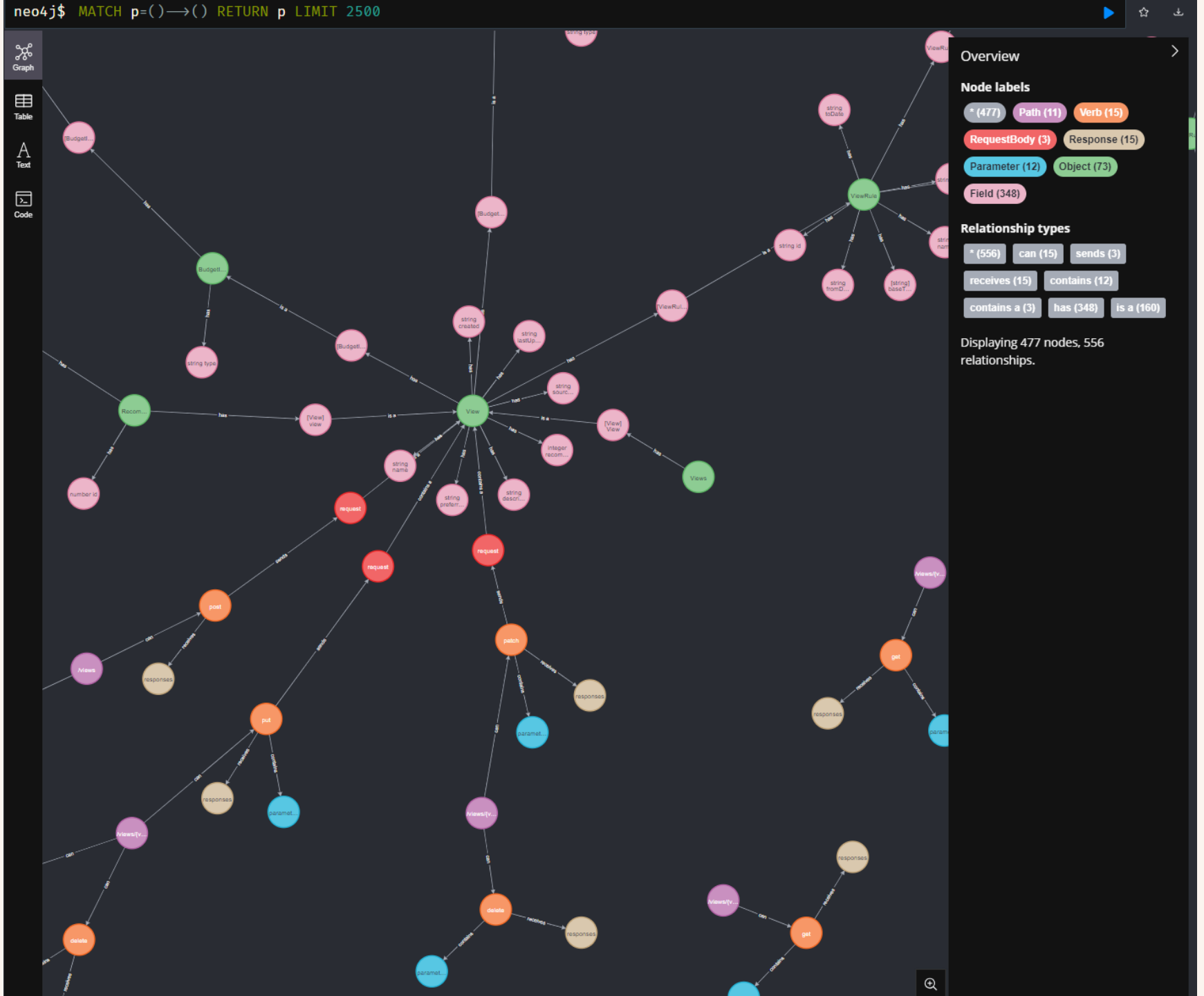
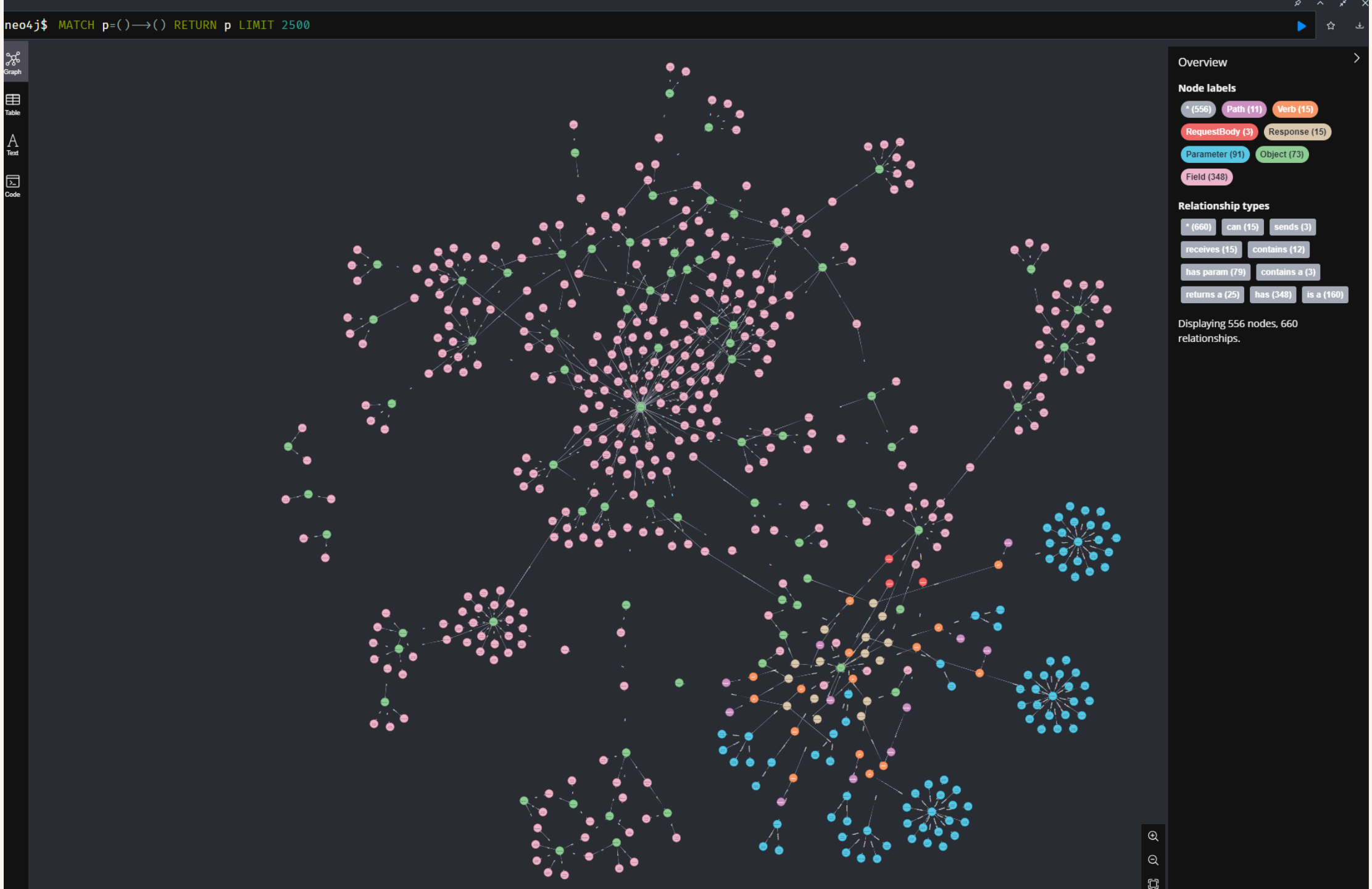


Table
Text
Code
/views/{v...
can
can
can
get
put
delete
contains
sends
receives
paramet...
request
responses
has param
string viewId
contains a
returns a
[Budget...
[View] view
[View] View
[BudgetI...
[ViewRul...
is a
has
string name
string created
View
string preferr...
string lastUp...
string descri...
integer recom...
string sourc...
Error
string errorC...
string errorM...
string refere...
* (25)
Field (15)
RequestBody (1)
Verb (3)
Parameter (2)
Response (1)
Path (1)
Object (2)
Relationship types
* (24)
has (13)
is a (2)
contains a (1)
sends (1)
can (3)
contains (1)
receives (1)
has param (1)
returns a (1)
Displaying 25 nodes, 24 relationships.


neo4j$ MATCH p=()⟶() RETURN p LIMIT 2500
Graph
Table
Text
Code
Overview
Node labels
* (477)
Path (11)
Verb (15)
RequestBody (3)
Response (15)
Parameter (12)
Object (73)
Field (348)
Relationship types
* (556)
can (15)
sends (3)
receives (15)
contains (12)
contains a (3)
has (348)
is a (160)
Displaying 477 nodes, 556 relationships.


neo4j$ MATCH p=()⟶() RETURN p LIMIT 2500
Graph
Table
Text
Code
Overview
Node labels
* (556)
Path (11)
Verb (15)
RequestBody (3)
Response (15)
Parameter (91)
Object (73)
Field (348)
Relationship types
* (660)
can (15)
sends (3)
receives (15)
contains (12)
has param (79)
contains a (3)
returns a (25)
has (348)
is a (160)
Displaying 556 nodes, 660 relationships.

Given a field/object - get all paths to there starting from endpoint

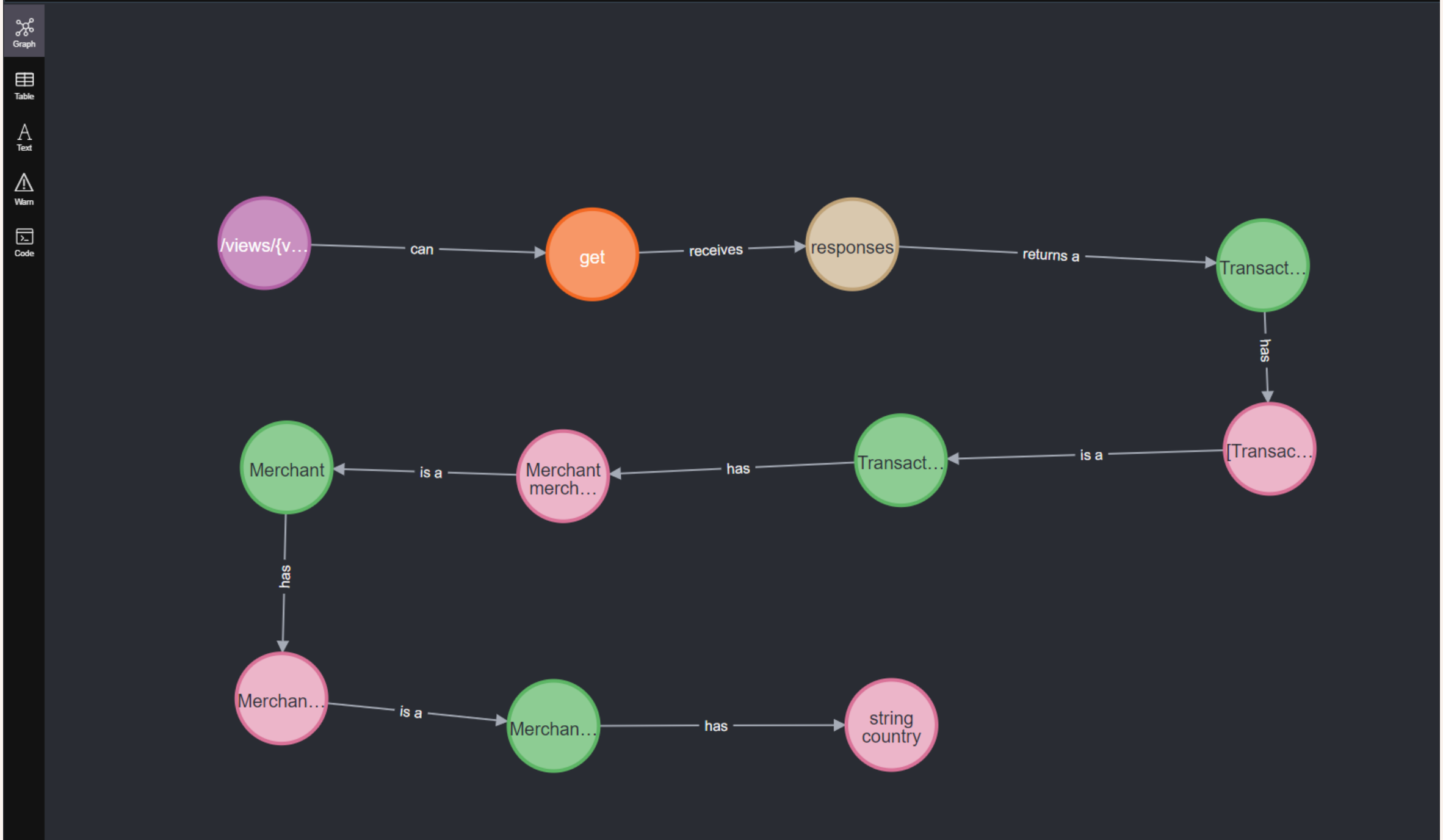
```
MATCH (start), (end {uid: "MerchantAddress_string_country"})
WHERE start.uid STARTS WITH "path_"
MATCH path = (start)-[*..10]->(end)
RETURN path

```

Find path between 2 nodes

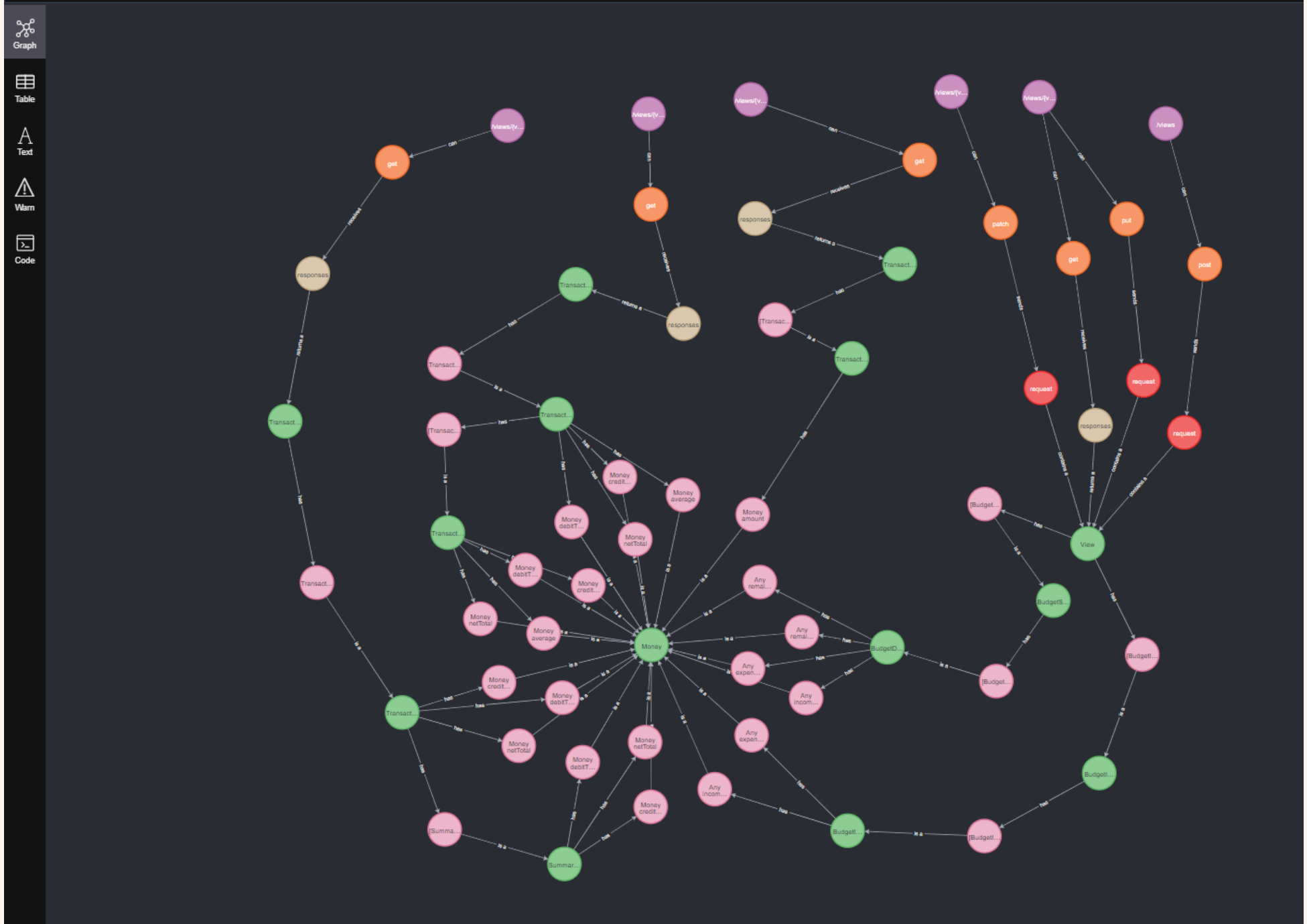
```
MATCH path = (start {uid: "Transaction_object"})-[*1..5]-(end {uid: "MerchantAddress_string_country"})
RETURN path
```

Find a specific object and it's links
```
MATCH (n {uid: "BasicAccount_object"})-[r]-(connected)
RETURN n, r, connected

```

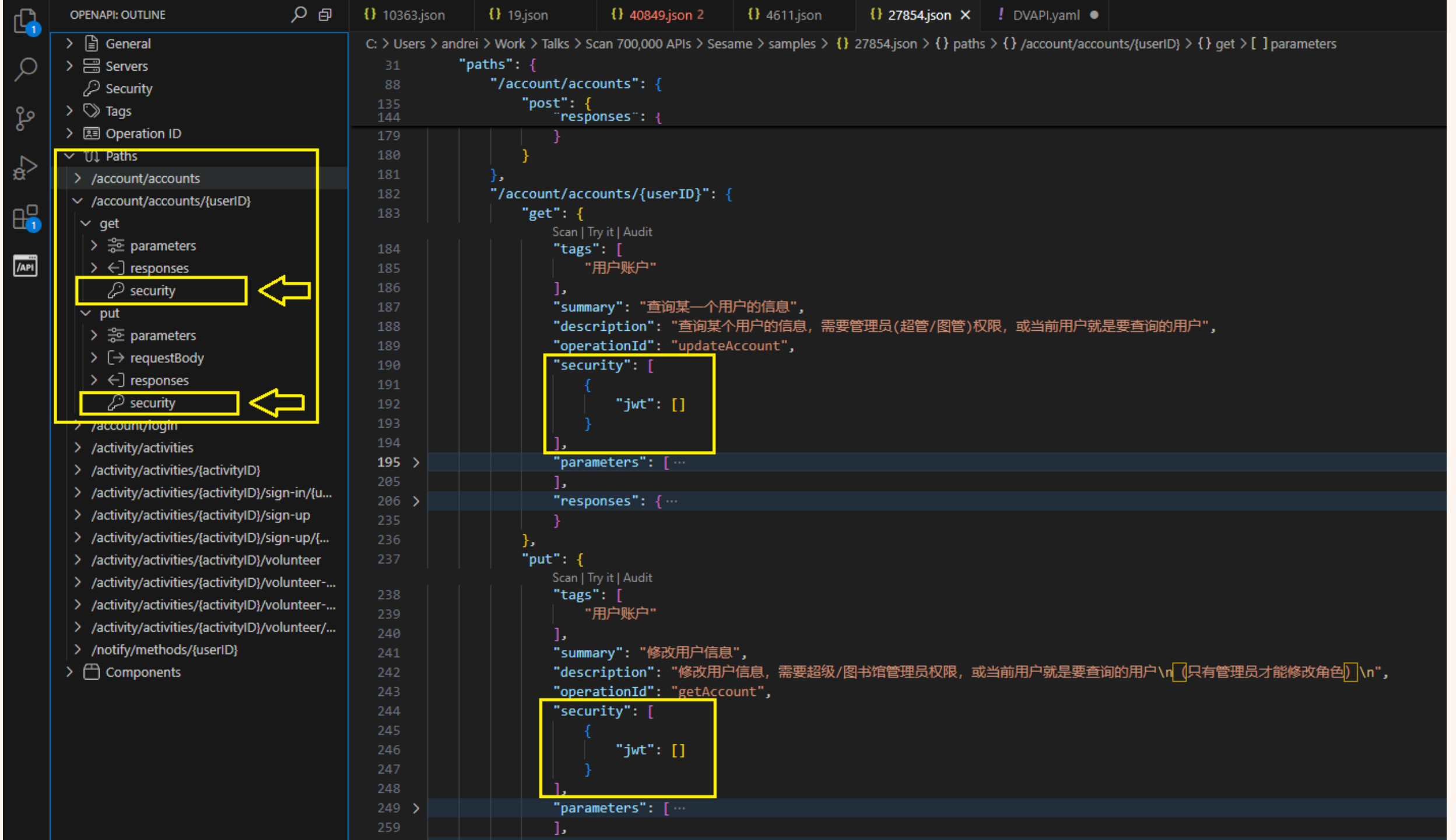
```
neo4j$ MATCH (start), (end {uid: "MerchantAddress_string_country"}) WHERE start.uid STARTS WITH "path_" MATCH path = (start)-[*..10]→(end) RETURN path
```



```
neo4j$ MATCH (start), (end {uid: "Money_object"}) WHERE start.uid STARTS WITH "path_" MATCH path = (start)-[*..10]→(end) RETURN path
```

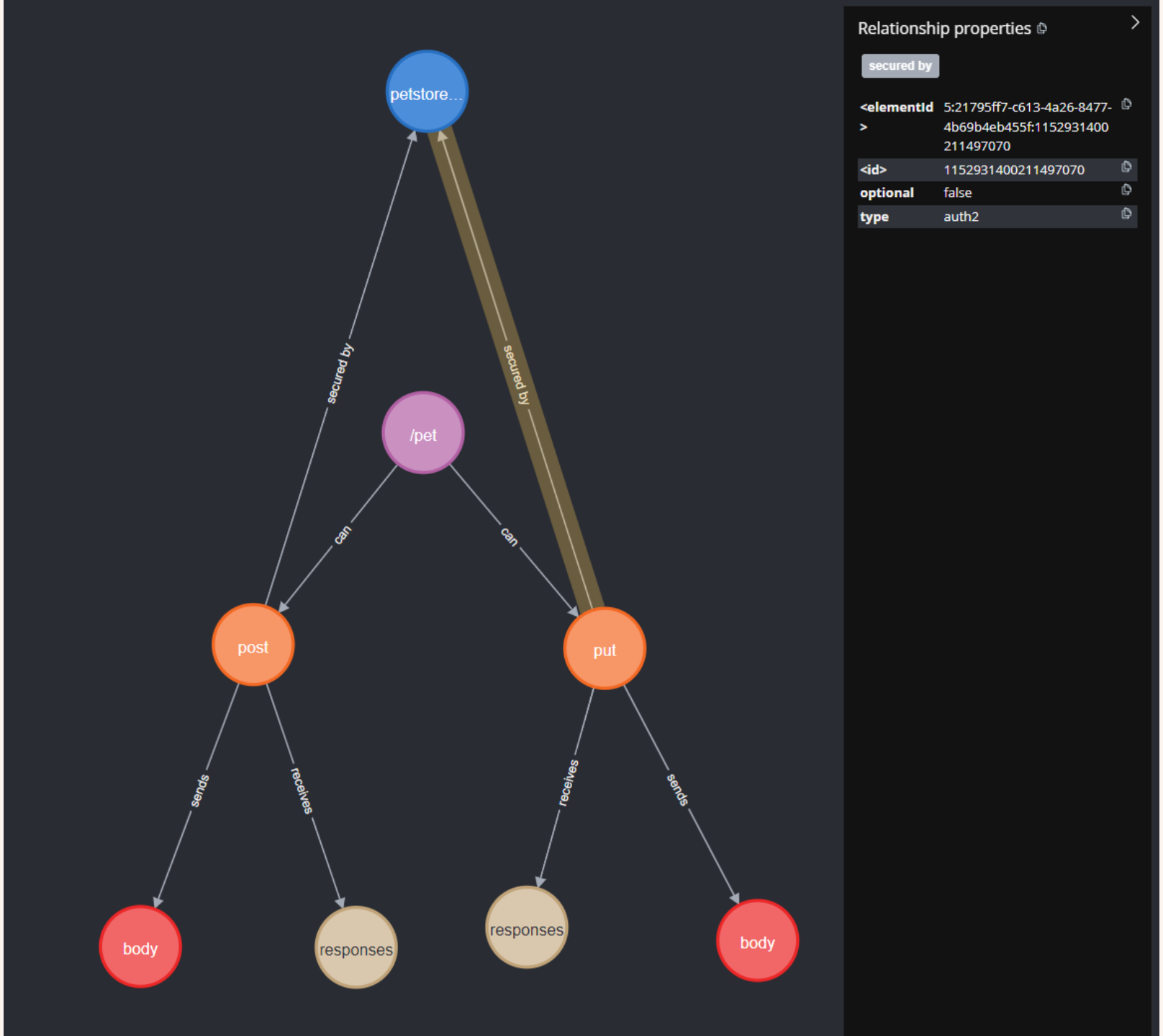


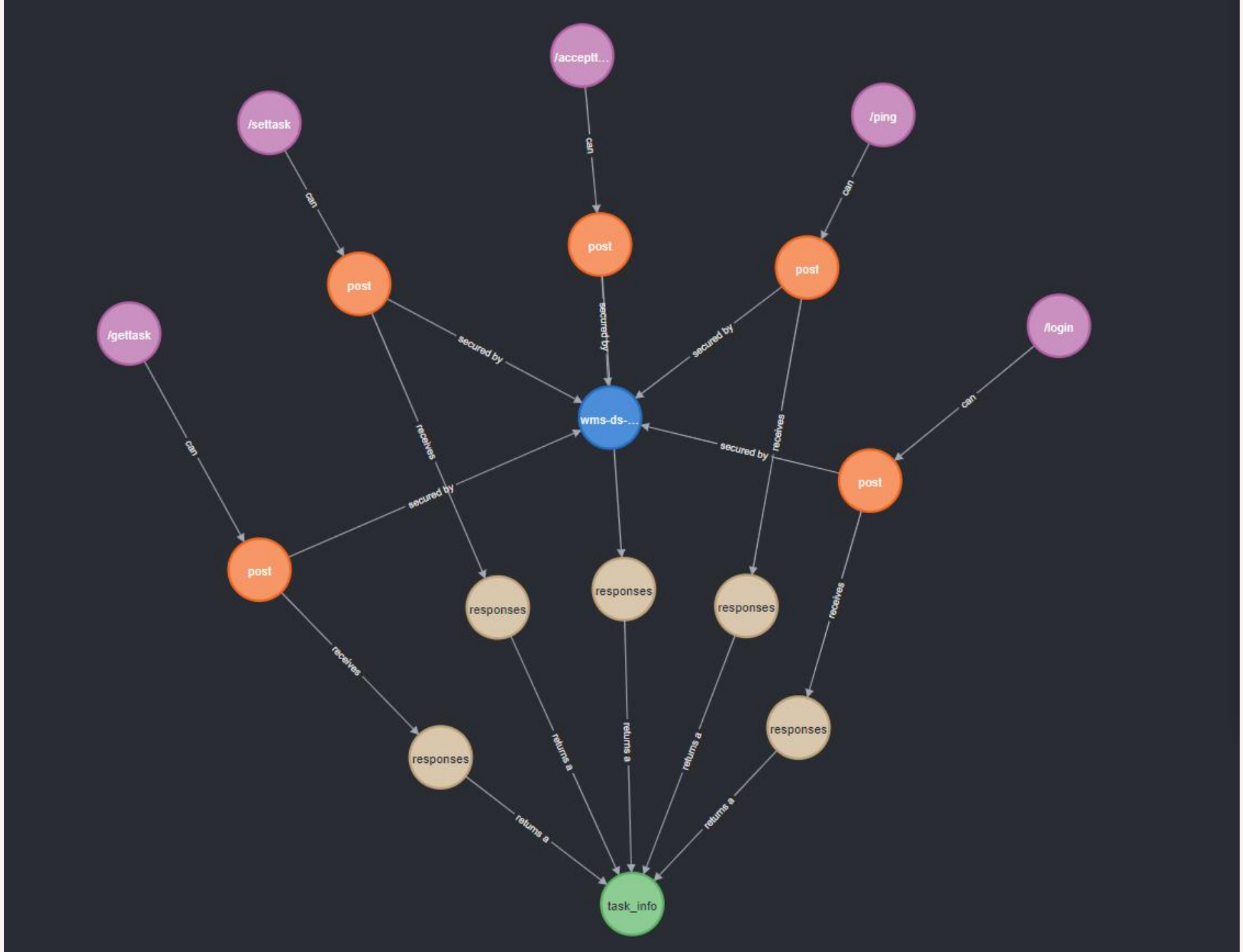
Field Name	Type	Description
openapi	`string`	**REQUIRED**. This string MUST be the semantic version number of the OpenAPI Specification version that the OpenAPI document uses. The `openapi` field SHOULD be used by tooling specifications and clients to interpret the OpenAPI document. This is *not* related to the API `info.version` string.
info	Info Object	**REQUIRED**. Provides metadata about the API. The metadata MAY be used by tooling as required.
servers	[Server Object]	An array of Server Objects, which provide connectivity information to a target server. If the `servers` property is not provided, or is an empty array, the default value would be a Server Object with a url value of `/`.
paths	Paths Object	**REQUIRED**. The available paths and operations for the API.
components	Components Object	An element to hold various schemas for the specification.
security	[Security Requirement Object]	A declaration of which security mechanisms can be used across the API. The list of values includes alternative security requirement objects that can be used. Only one of the security requirement objects need to be satisfied to authorize a request. Individual operations can override this definition. To make security optional, an empty security requirement (`{}`) can be included in the array.

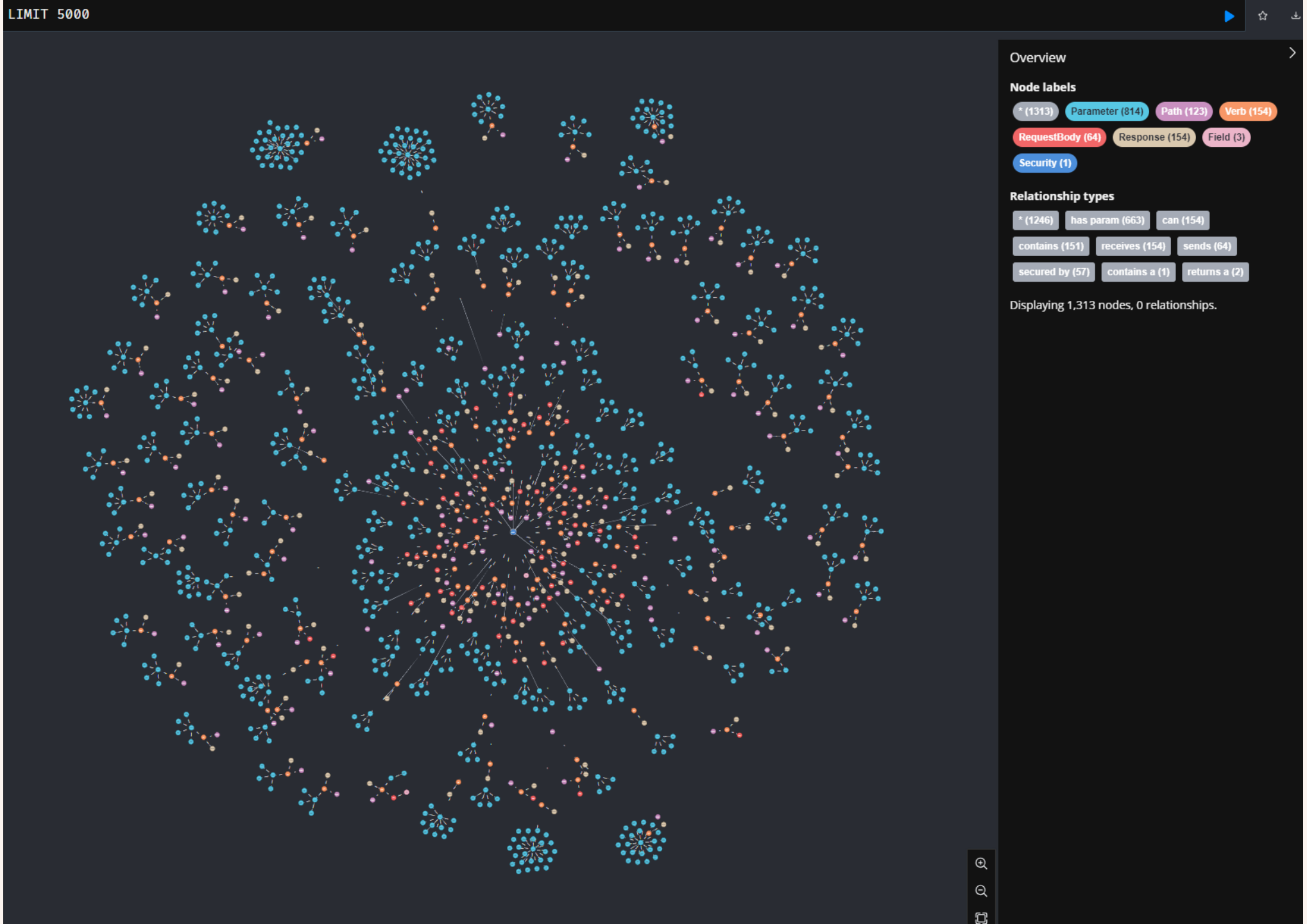

OPENAPI: OUTLINE
General
Servers
Security
Tags
Operation ID
Paths
/account/accounts
/account/accounts/{userID}
get
parameters
responses
security
put
parameters
requestBody
responses
security
/activity/activities
/activity/activities/{activityID}
/activity/activities/{activityID}/sign-in/{u...
/activity/activities/{activityID}/sign-up
/activity/activities/{activityID}/sign-up/{...
/activity/activities/{activityID}/volunteer
/activity/activities/{activityID}/volunteer-...
/activity/activities/{activityID}/volunteer-...
/activity/activities/{activityID}/volunteer/...
/notify/methods/{userID}
Components
10363.json
19.json
40849.json 2
4611.json
27854.json
DVAPI.yaml
C: > Users > andrei > Work > Talks > Scan 700,000 APIs > Sesame > samples > 27854.json > paths > /account/accounts/{userID} > get > parameters
"paths": {
"/account/accounts": {
"post": {
"responses": {
}
}
},
"/account/accounts/{userID}": {
"get": {
Scan | Try it | Audit
"tags": [
"用户账户"
],
"summary": "查询某一个用户的信息",
"description": "查询某个用户的信息，需要管理员(超管/图管)权限，或当前用户就是要查询的用户",
"operationId": "updateAccount",
"security": [
{
"jwt": []
}
],
"parameters": [
],
"responses": {
}
},
"put": {
Scan | Try it | Audit
"tags": [
"用户账户"
],
"summary": "修改用户信息",
"description": "修改用户信息，需要超级/图书馆管理员权限，或当前用户就是要查询的用户\n(只有管理员才能修改角色)\n",
"operationId": "getAccount",
"security": [
{
"jwt": []
}
],
"parameters": [
],

In an OpenAPI document, `security`, `security schemes within components`, and `security schemes within paths` serve different purposes when defining and applying security requirements. Here's a table comparing these concepts:

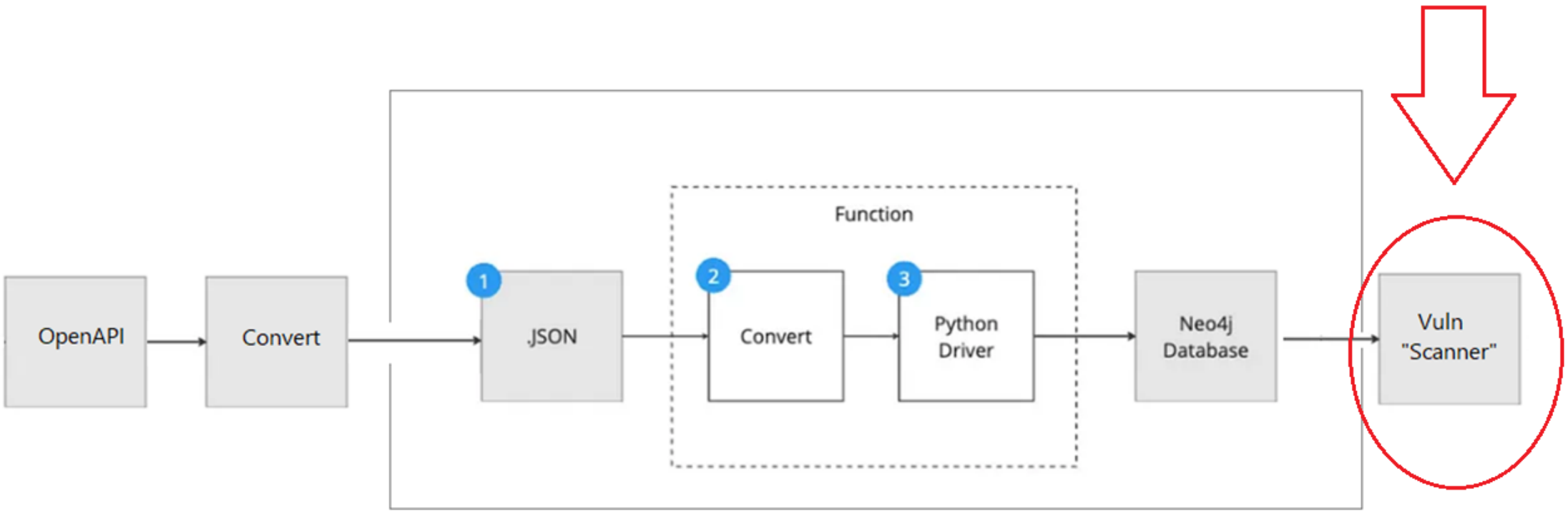
Aspect	`security`	`securitySchemes` (within `components`)	`security` (within `paths`)
Purpose	Specifies the security requirements that apply globally to the entire API or operation.	Defines the available authentication methods or mechanisms, such as OAuth2, API keys, or HTTP authentication.	Specifies security requirements for specific paths or operations, overriding the global `security` requirements.
Location in OpenAPI Document	At the root level of the document or under specific paths/operations.	Under `components` in the document.	Inside a path or operation definition.
Impact	Sets security for all operations if defined globally, unless overridden by path-specific security settings.	Provides definitions for authentication methods, but does not apply them directly.	Overrides global security settings for specific paths or operations.

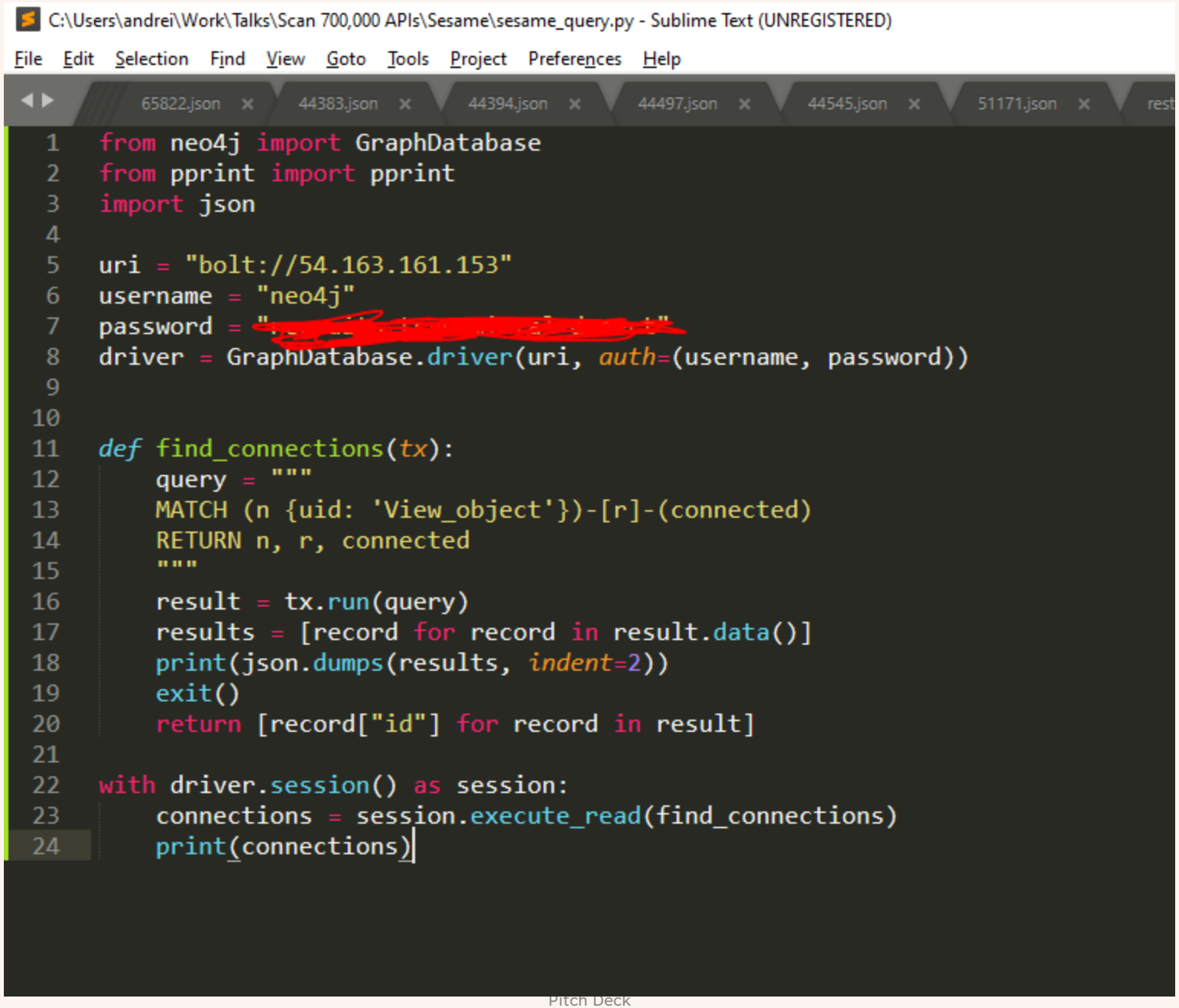

petstore...
secured by
secured by
/pet
can
can
post
put
sends
receives
receives
sends
body
responses
responses
body
Relationship properties
secured by
<elementId> 5:21795ff7-c613-4a26-8477-4b69b4eb455f:1152931400211497070
<id> 1152931400211497070
optional false
type auth2


/acceptt...
/settask
/ping
can
can
can
post
post
post
secured by
secured by
secured by
/gettask
/login
wms-ds-...
receives
receives
can
can
secured by
secured by
post
post
responses
responses
responses
responses
receives
receives
returns a
returns a
returns a
returns a
task_info

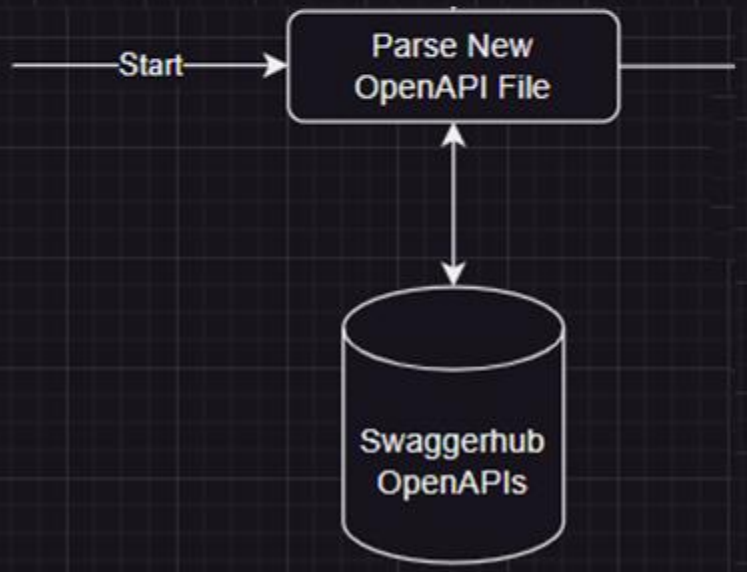

LIMIT 5000
Overview
Node labels
* (1313)
Parameter (814)
Path (123)
Verb (154)
RequestBody (64)
Response (154)
Field (3)
Security (1)
Relationship types
* (1246)
has param (663)
can (154)
contains (151)
receives (154)
sends (64)
secured by (57)
contains a (1)
returns a (2)
Displaying 1,313 nodes, 0 relationships.

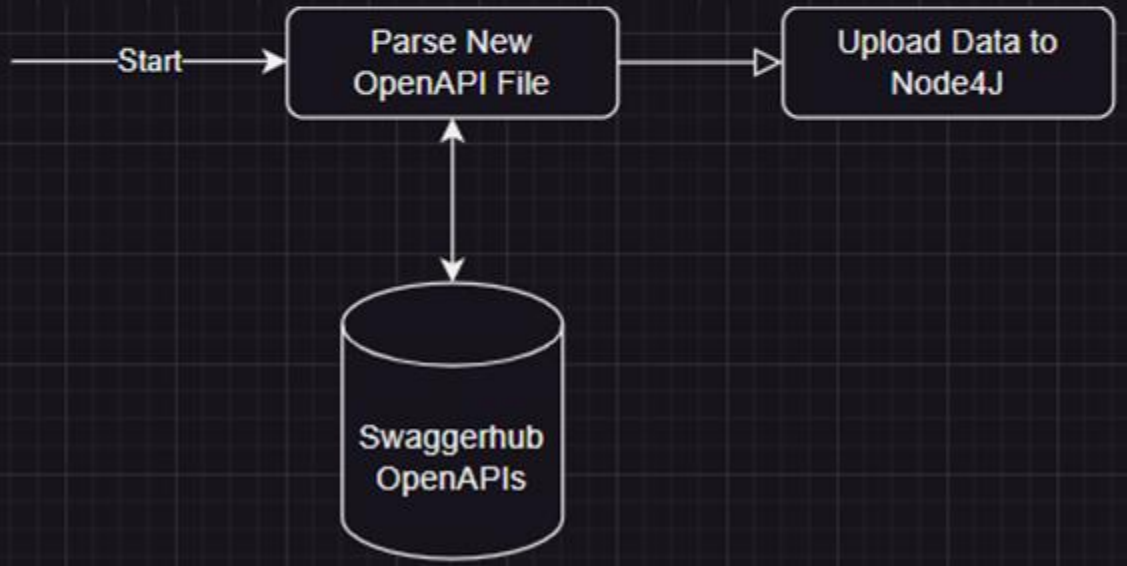
5. RESULTS

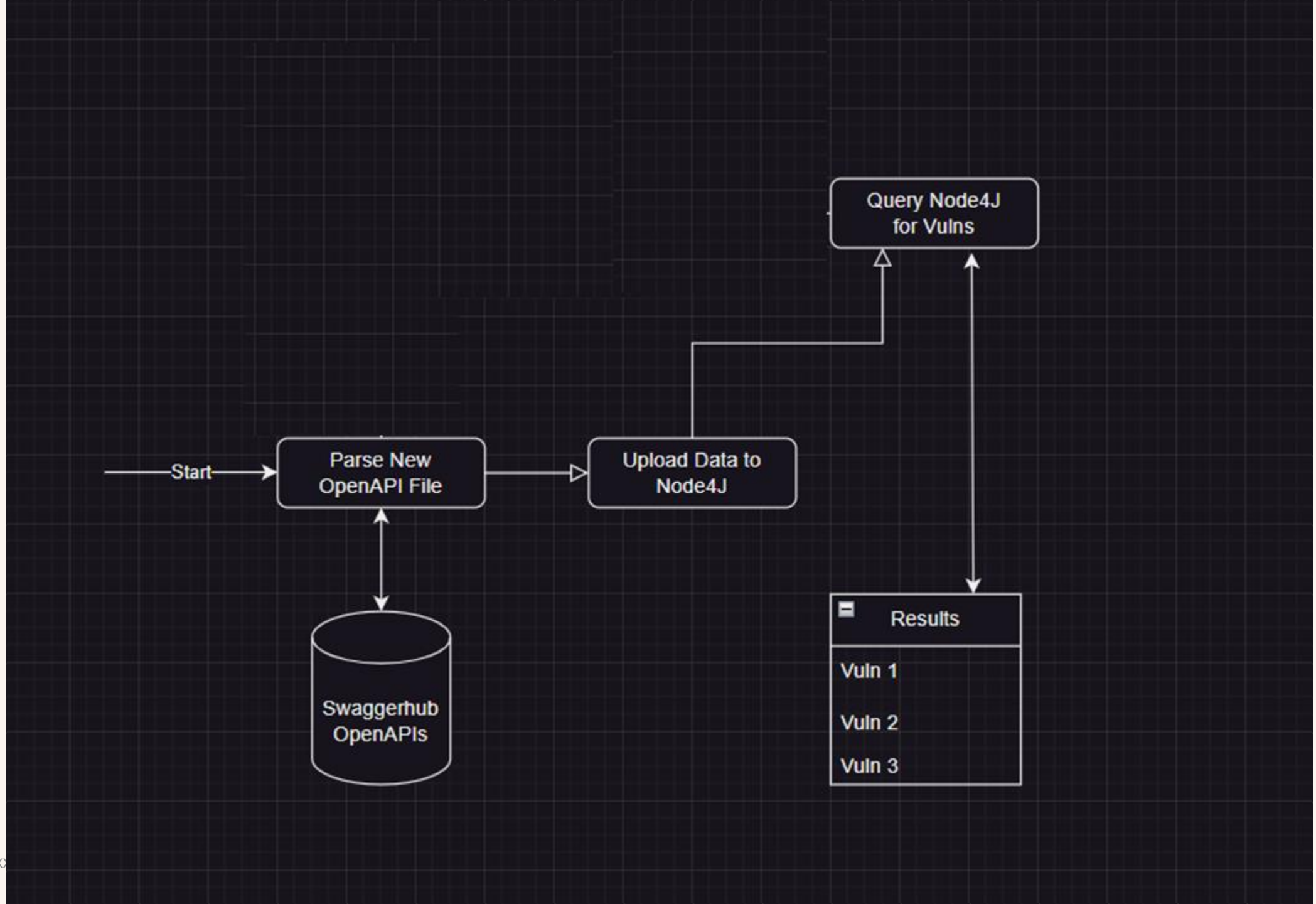

OpenAPI
Convert
1
JSON
Function
2
Convert
3
Python
Driver
Neo4j
Database
Vuln
"Scanner"

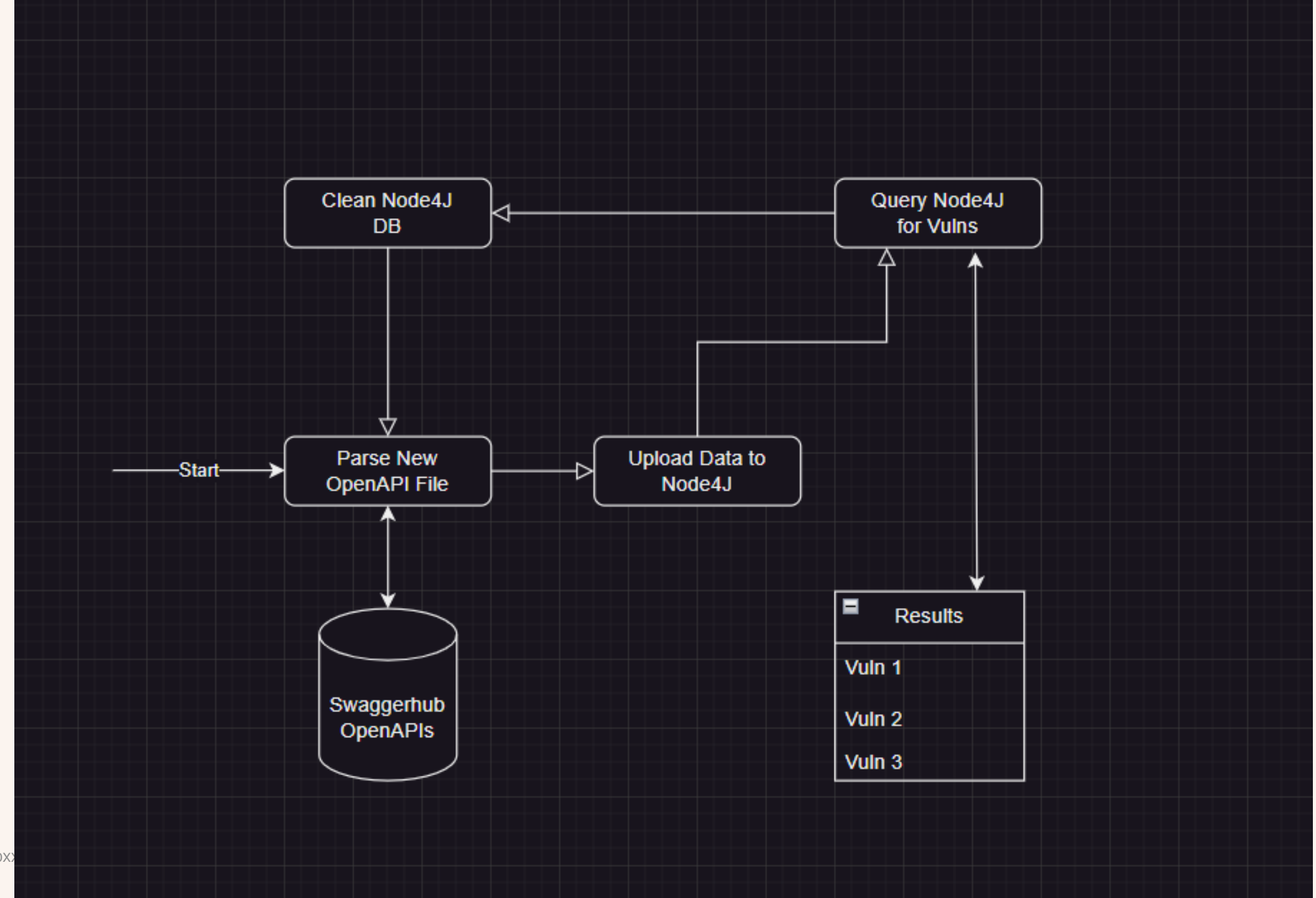

C:\Users\andrei\Work\Talks\Scan 700,000 APIs\Sesame\sesame_query.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
65822.json
44383.json
44394.json
44497.json
44545.json
51171.json
from neo4j import GraphDatabase
from pprint import pprint
import json
uri = "bolt://54.163.161.153"
username = "neo4j"
password = "
driver = GraphDatabase.driver(uri, auth=(username, password))
def find_connections(tx):
query = """
MATCH (n {uid: 'View_object'})-[r]-(connected)
RETURN n, r, connected
"""
result = tx.run(query)
results = [record for record in result.data()]
print(json.dumps(results, indent=2))
exit()
return [record["id"] for record in result]
with driver.session() as session:
connections = session.execute_read(find_connections)
print(connections)

```
C:\Users\andrei\Work\Talks\Scan 700,000 APIs\Sesame>python3 sesame_query.py
{'n': {'uid': 'device_info_object', 'name': 'device_info', 'type': 'object'}}
{'n': {'uid': 'device_info_short_object', 'name': 'device_info_short', 'type': 'object'}}
{'n': {'uid': 'config_res_object', 'name': 'config_res', 'type': 'object'}}
{'n': {'uid': 'task_info_object', 'name': 'task_info', 'type': 'object'}}
{'n': {'uid': 'task_goods_object', 'name': 'task_goods', 'type': 'object'}}
{'n': {'uid': 'task_cells_object', 'name': 'task_cells', 'type': 'object'}}
{'n': {'uid': 'task_series_object', 'name': 'task_series', 'type': 'object'}}
{'n': {'uid': 'task_put_cells_object', 'name': 'task_put_cells', 'type': 'object'}}
{'n': {'uid': 'token_req_object', 'name': 'token_req', 'type': 'object'}}
{'n': {'uid': 'error_resp_object', 'name': 'error_resp', 'type': 'object'}}
{'n': {'uid': 'text_resp_object', 'name': 'text_resp', 'type': 'object'}}
-------
{'path': [{'uid': 'path_/login', 'name': '/login', 'type': 'path'}, 'can', {'uid': 'post_/login', 'name': 'post', 'type': 'verb'
}, 'receives', {'uid': 'post_/login_responses', 'name': 'responses', 'type': 'responses'}, 'returns a', {'uid': 'task_info_objec
t', 'name': 'task_info', 'type': 'object'}]}
...
{'path': [{'uid': 'path_/ping', 'name': '/ping', 'type': 'path'}, 'can', {'uid': 'post_/ping', 'name': 'post', 'type': 'verb'},
'receives', {'uid': 'post_/ping_responses', 'name': 'responses', 'type': 'responses'}, 'returns a', {'uid': 'task_info_object',
'name': 'task_info', 'type': 'object'}]}
...
{'path': [{'uid': 'path_/accepttask', 'name': '/accepttask', 'type': 'path'}, 'can', {'uid': 'post_/accepttask', 'name': 'post',
 'type': 'verb'}, 'receives', {'uid': 'post_/accepttask_responses', 'name': 'responses', 'type': 'responses'}, 'returns a', {'ui
d': 'task_info_object', 'name': 'task_info', 'type': 'object'}]}
...
{'path': [{'uid': 'path_/gettask', 'name': '/gettask', 'type': 'path'}, 'can', {'uid': 'post_/gettask', 'name': 'post', 'type':
'verb'}, 'receives', {'uid': 'post_/gettask_responses', 'name': 'responses', 'type': 'responses'}, 'returns a', {'uid': 'task_in
fo_object', 'name': 'task_info', 'type': 'object'}]}
...
{'path': [{'uid': 'path_/settask', 'name': '/settask', 'type': 'path'}, 'can', {'uid': 'post_/settask', 'name': 'post', 'type':
'verb'}, 'receives', {'uid': 'post_/settask_responses', 'name': 'responses', 'type': 'responses'}, 'returns a', {'uid': 'task_in
fo_object', 'name': 'task_info', 'type': 'object'}]}
...

C:\Users\andrei\Work\Talks\Scan 700,000 APIs\Sesame>
```


Start
Parse New
OpenAPI File
Swaggerhub
OpenAPIs

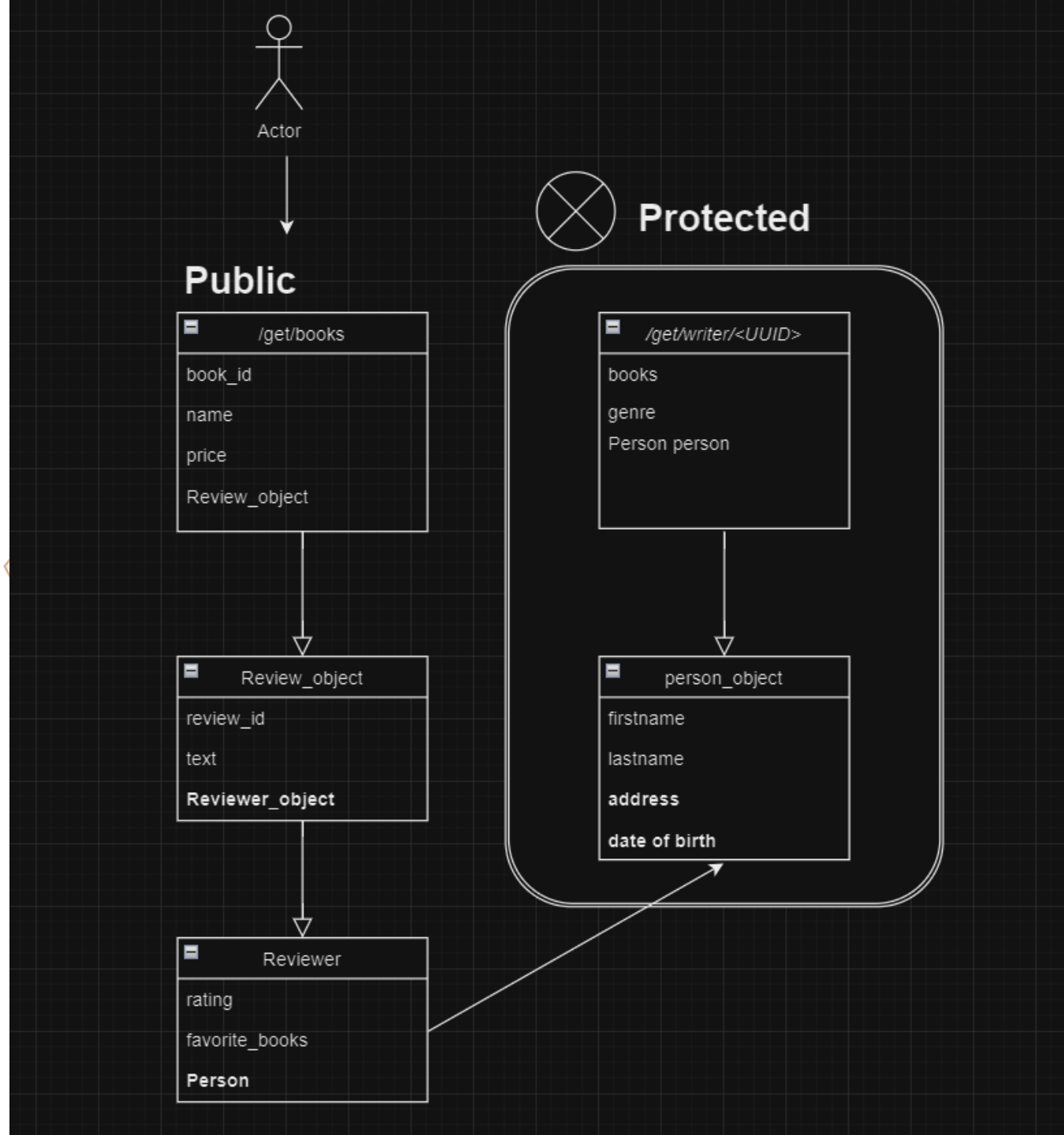

Start
Parse New OpenAPI File
Upload Data to Node4J
Swaggerhub OpenAPIs


Query Node4J
for Vulns
Start
Parse New
OpenAPI File
Upload Data to
Node4J
Swaggerhub
OpenAPIs
Results
Vuln 1
Vuln 2
Vuln 3


Clean Node4J DB
Query Node4J for Vulns
Start
Parse New OpenAPI File
Upload Data to Node4J
Swaggerhub OpenAPIs
Results
Vuln 1
Vuln 2
Vuln 3

1. INDIRECT BROKEN ACCESS CONTROL

1. FIND OBJECT WITH SENSITIVE INFO
2. CHECK ALL PATHS TO READ/MODIFY IT
3. CHECK FOR DIRECT AND INDIRECT ACCESS



```
#for each object
    #get all paths go get to the object
        #for each path
            #check if path require security
                #if we find 1 path that requires and 1 path that doesn't require auth
                    # -> possible indirect broken access control?
```

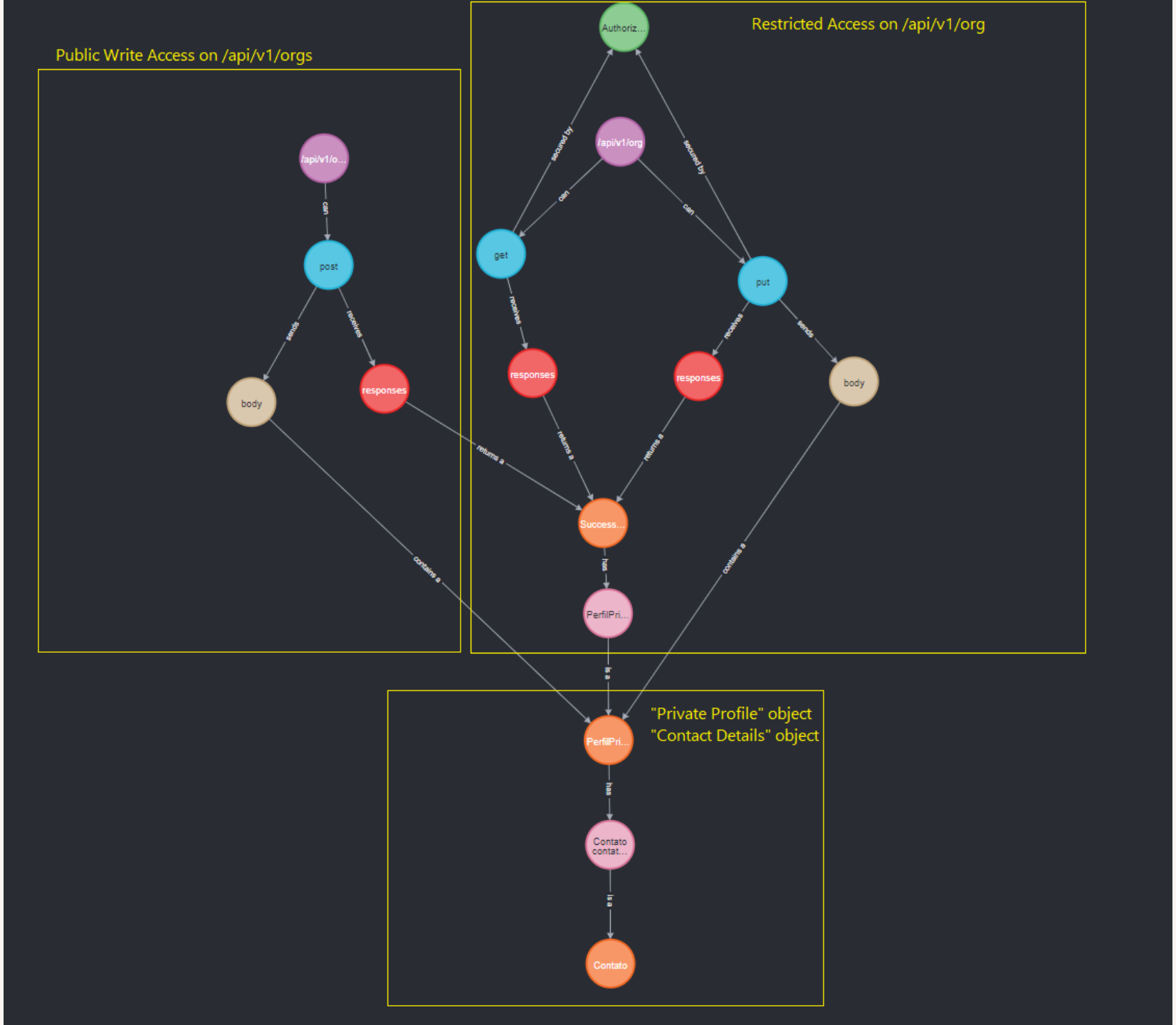
```
C:\Users\andrei\Work\Talks\Scan 700,000 APIs\Sesame>python3 sesame_wrapper.py
[------------------------------]
[+] Analyzing samples\10032.json
[+] Parsing complete!
[+] Uploading data to Neo4j server..
[+] Data uploaded succesfully!
[+] Checking vulnerable paths..
[+] Check complete!
[+] Cleaning Neo4j DB..
[+] Neo4j DB deleted!
[------------------------------]
[+] Analyzing samples\10043.json
[+] Parsing complete!
[+] Uploading data to Neo4j server..
[+] Data uploaded succesfully!
[+] Checking vulnerable paths..
------
[+] Check complete!
[+] Cleaning Neo4j DB..
[+] Neo4j DB deleted!
[------------------------------]
[+] Analyzing samples\10056.json
[+] Parsing complete!
[+] Uploading data to Neo4j server..
[+] Data uploaded succesfully!
[+] Checking vulnerable paths..
------
------
------
------
------
------
------
[+] Check complete!
[+] Cleaning Neo4j DB..
[+] Neo4j DB deleted!
[------------------------------]
```

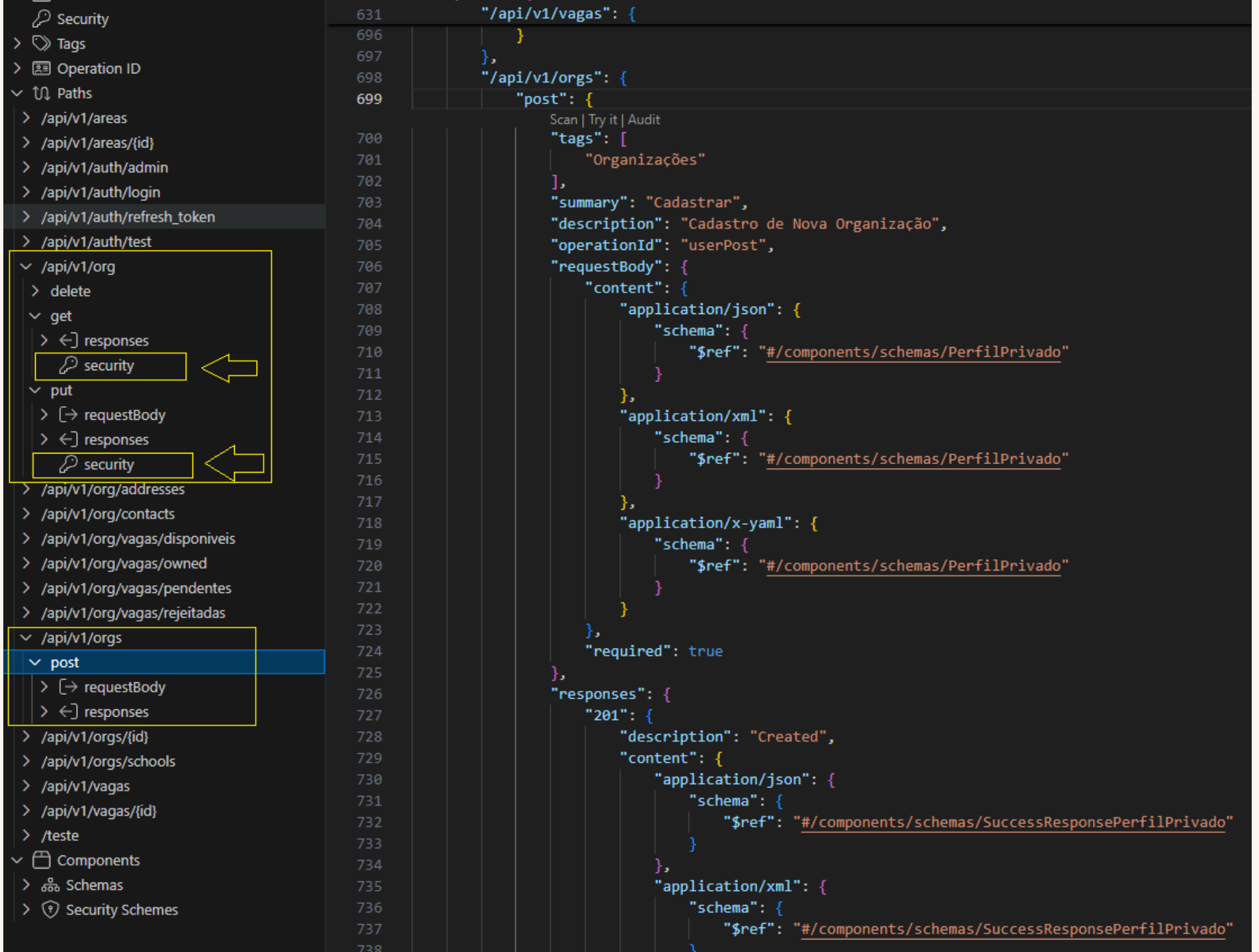
```
[+] Analyzing samples\10113.json
[+] Parsing complete!
[+] Uploading data to Neo4j server..
[+] Data uploaded succesfully!
[+] Checking vulnerable paths..
------
------
------
[+++++] Object CharityProjectDBcan be reached from public & protected endpoints
------
------
------
------
------
[+++++] Object ErrorModelcan be reached from public & protected endpoints
------
[+++++] Object HTTPValidationErrorcan be reached from public & protected endpoints
------
[+++++] Object Usercan be reached from public & protected endpoints
------
------
------
[+++++] Object ValidationErrorcan be reached from public & protected endpoints
```

```
[+] Checking vulnerable paths..
------
------
------
------
------
[+++++] Object AccountsDTOcan be reached from public & protected endpo
------
------
------
[+++++] Object Usercan be reached from public & protected endpoints
------
[+++++] Object Accountcan be reached from public & protected endpoints
------
------
[+++++] Object Transactioncan be reached from public & protected endpo
------
[+++++] Object AccountTypecan be reached from public & protected endpo
------
[+++++] Object AccountStatuscan be reached from public & protected end
------
[+++++] Object TransactionTypecan be reached from public & protected e
------
[+++++] Object UserTypecan be reached from public & protected endpoint
------
```

```
[------------------------------]
[+] Analyzing samples\1016.json
[+] Parsing complete!
[+] Uploading data to Neo4j server..
[+] Data uploaded succesfully!
[+] Checking vulnerable paths..
[+++++] Object Usercan be reached from public & protected endpoints
------
[+++++] Object Artistcan be reached from public & protected endpoints
------
[+++++] Object Albumcan be reached from public & protected endpoints
------
[+++++] Object Trackcan be reached from public & protected endpoints
------
------
[+] Check complete!
[+] Cleaning Neo4j DB..
[+] Neo4j DB deleted!
[------------------------------]
```

```
[------------------------------]
[+] Analyzing samples\10363.json
[+] Parsing complete!
[+] Uploading data to Neo4j server..
[+] Data uploaded succesfully!
[+] Checking vulnerable paths..
[+++++] Object SKUcan be reached from public & protected endpoints
------
[+++++] Object Productcan be reached from public & protected endpoints
------
[+++++] Object Providercan be reached from public & protected endpoints
------
------
------
```


Public Write Access on /api/v1/orgs
/api/v1/o...
can
post
sends
receives
body
responses
returns a
Restricted Access on /api/v1/org
Authoriz...
secured by
/api/v1/org
can
get
receives
responses
returns a
put
receives
responses
returns a
sends
body
contains a
Success...
has
PerfilPri...
is a
"Private Profile" object
"Contact Details" object
PerfilPri...
has
Contato
contat...
is a
Contato

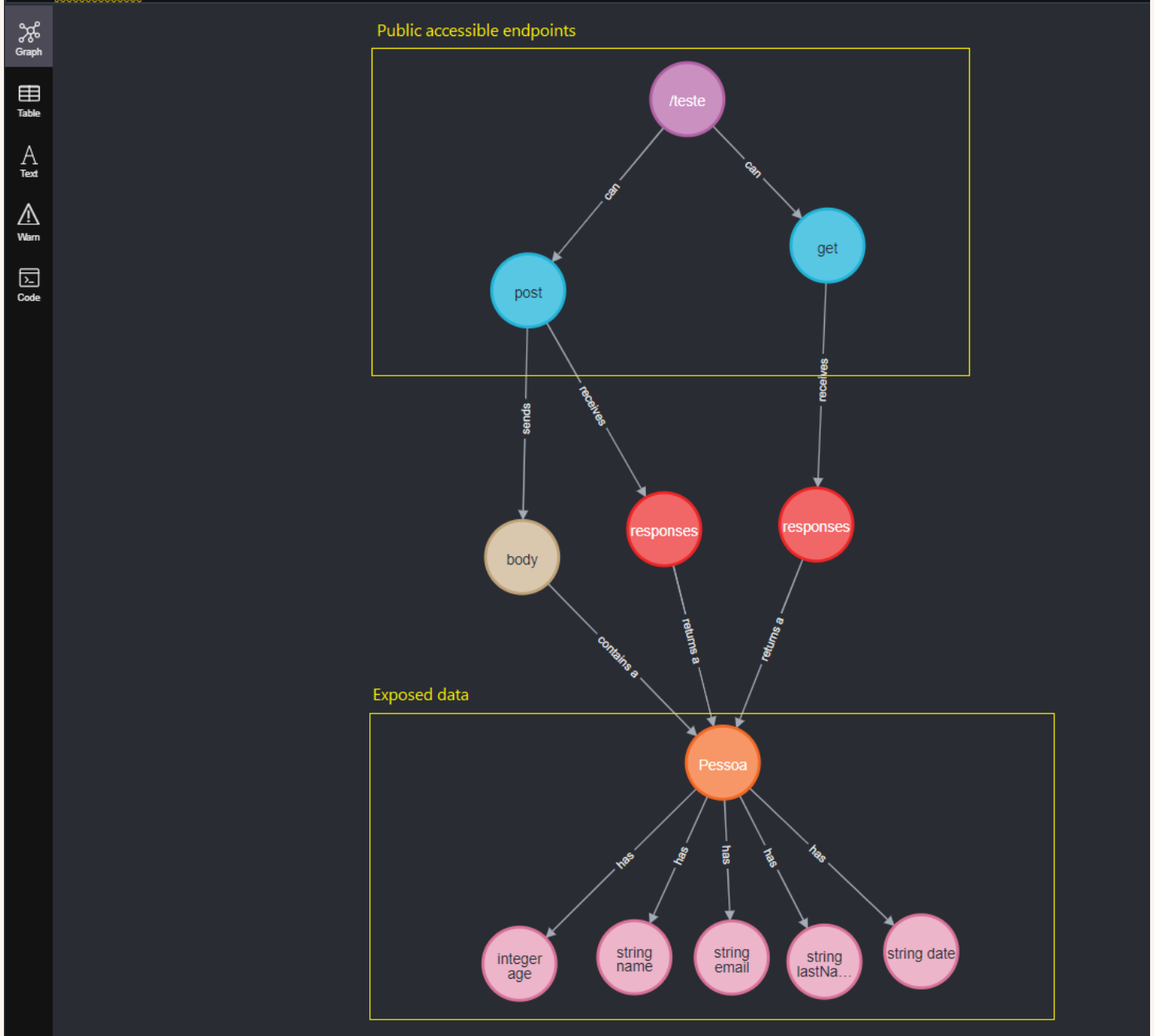

Security
Tags
Operation ID
Paths
/api/v1/areas
/api/v1/areas/{id}
/api/v1/auth/admin
/api/v1/auth/login
/api/v1/auth/refresh_token
/api/v1/auth/test
/api/v1/org
delete
get
responses
security
put
requestBody
responses
security
/api/v1/org/addresses
/api/v1/org/contacts
/api/v1/org/vagas/disponiveis
/api/v1/org/vagas/owned
/api/v1/org/vagas/pendentes
/api/v1/org/vagas/rejeitadas
/api/v1/orgs
post
requestBody
responses
/api/v1/orgs/{id}
/api/v1/orgs/schools
/api/v1/vagas
/api/v1/vagas/{id}
/teste
Components
Schemas
Security Schemes
"/api/v1/vagas": {
}
},
"/api/v1/orgs": {
"post": {
Scan | Try it | Audit
"tags": [
"Organizações"
],
"summary": "Cadastrar",
"description": "Cadastro de Nova Organização",
"operationId": "userPost",
"requestBody": {
"content": {
"application/json": {
"schema": {
"$ref": "#/components/schemas/PerfilPrivado"
}
},
"application/xml": {
"schema": {
"$ref": "#/components/schemas/PerfilPrivado"
}
},
"application/x-yaml": {
"schema": {
"$ref": "#/components/schemas/PerfilPrivado"
}
}
},
"required": true
},
"responses": {
"201": {
"description": "Created",
"content": {
"application/json": {
"schema": {
"$ref": "#/components/schemas/SuccessResponsePerfilPrivado"
}
},
"application/xml": {
"schema": {
"$ref": "#/components/schemas/SuccessResponsePerfilPrivado"

2. PATHS TO SENSITIVE DATA

1. LOOK FOR FIELDS/OBJECT CONTAINING SENSITIVE DATA
2. CHECK ALL PUBLIC PATHS TO REACH IT
3. EXFIL THE DATA THROUGH UNPROTECTED ENDPOINTS

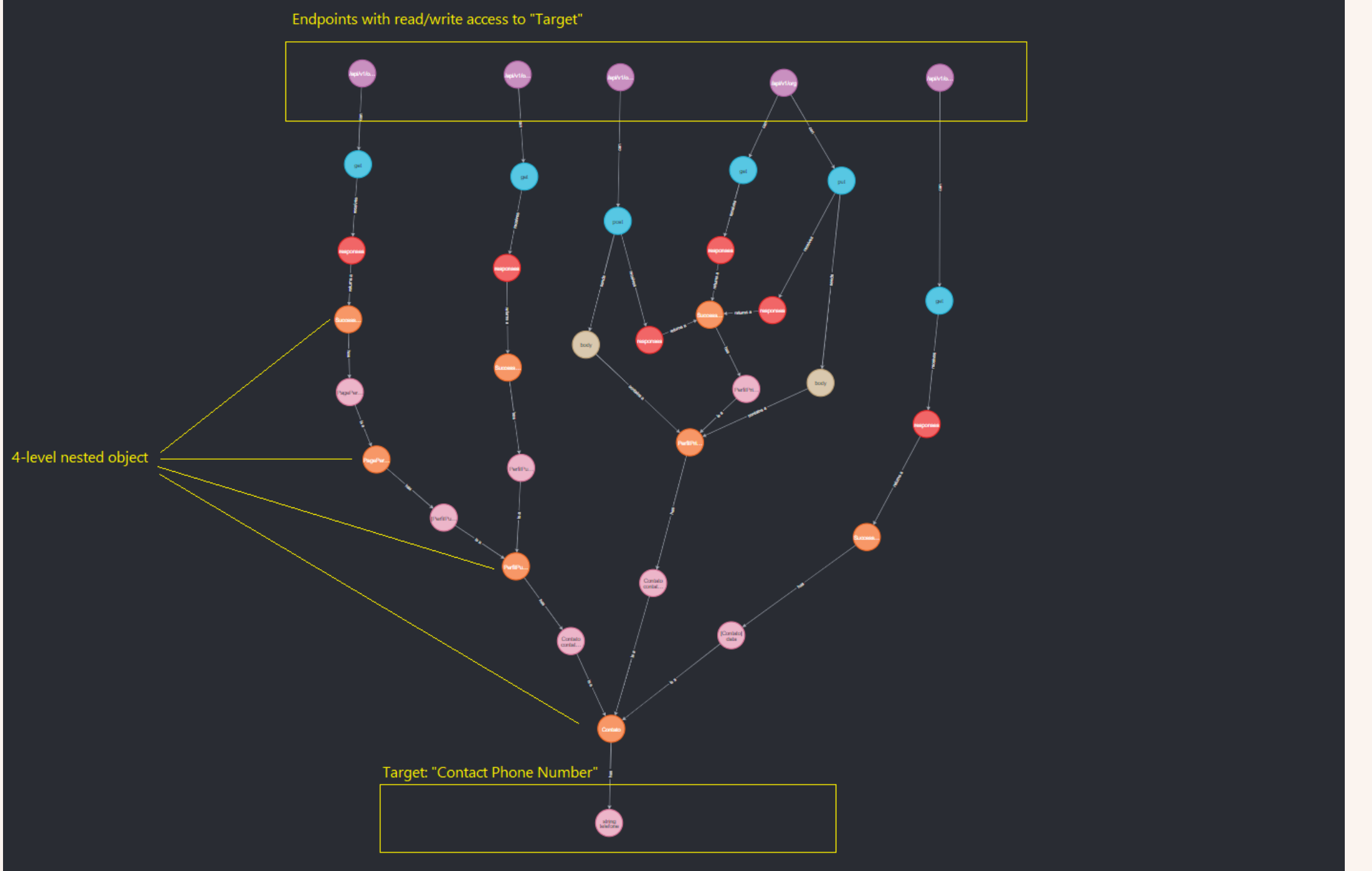
```
[Public Modify] Object Contato can be reached from post /api/v1/orgs
[Public Modify] Object Contato can be reached from post /api/v1/orgs
[Public Modify] Object PerfilPrivado can be reached from post /api/v1/orgs
[Public Modify] Object PerfilPrivado can be reached from post /api/v1/orgs
[Public Modify] Object PublicAddressResponse can be reached from post /api/v1/orgs
[Public Modify] Object PublicAddressResponse can be reached from post /api/v1/orgs
[Public Modify] Object PublicContactResponse can be reached from post /api/v1/orgs
[Public Modify] Object PublicContactResponse can be reached from post /api/v1/orgs
[Public Modify] Object UserCredentialsProjection can be reached from post /api/v1/orgs
[Public Modify] Object UserCredentialsProjection can be reached from post /api/v1/orgs
[Public Modify] Object SuccessResponsePerfilPrivado can be reached from post /api/v1/orgs
[Public Read] Object Pessoa can be reached from get /teste
[Public Modify] Object Pessoa can be reached from post /teste
[Public Modify] Object Pessoa can be reached from post /teste
[Public Modify] Object AuthRefreshTokenRequest can be reached from post /api/v1/auth/refresh_token
[Public Modify] Object AuthTokenResponse can be reached from post /api/v1/auth/refresh_token
[Public Modify] Object AuthTokenResponse can be reached from post /api/v1/auth/login
[Public Modify] Object SuccessResponseAuthTokenResponse can be reached from post /api/v1/auth/refresh_token
[Public Modify] Object SuccessResponseAuthTokenResponse can be reached from post /api/v1/auth/login
[Public Modify] Object AuthLoginRequest can be reached from post /api/v1/auth/login
```

```
MATCH (start), (end {uid: "Pessoa object"})
WHERE start.uid STARTS WITH "path "
MATCH path = (start)-[*..10]->(end)
RETURN path
```



3. BYPASS RATE LIMIT

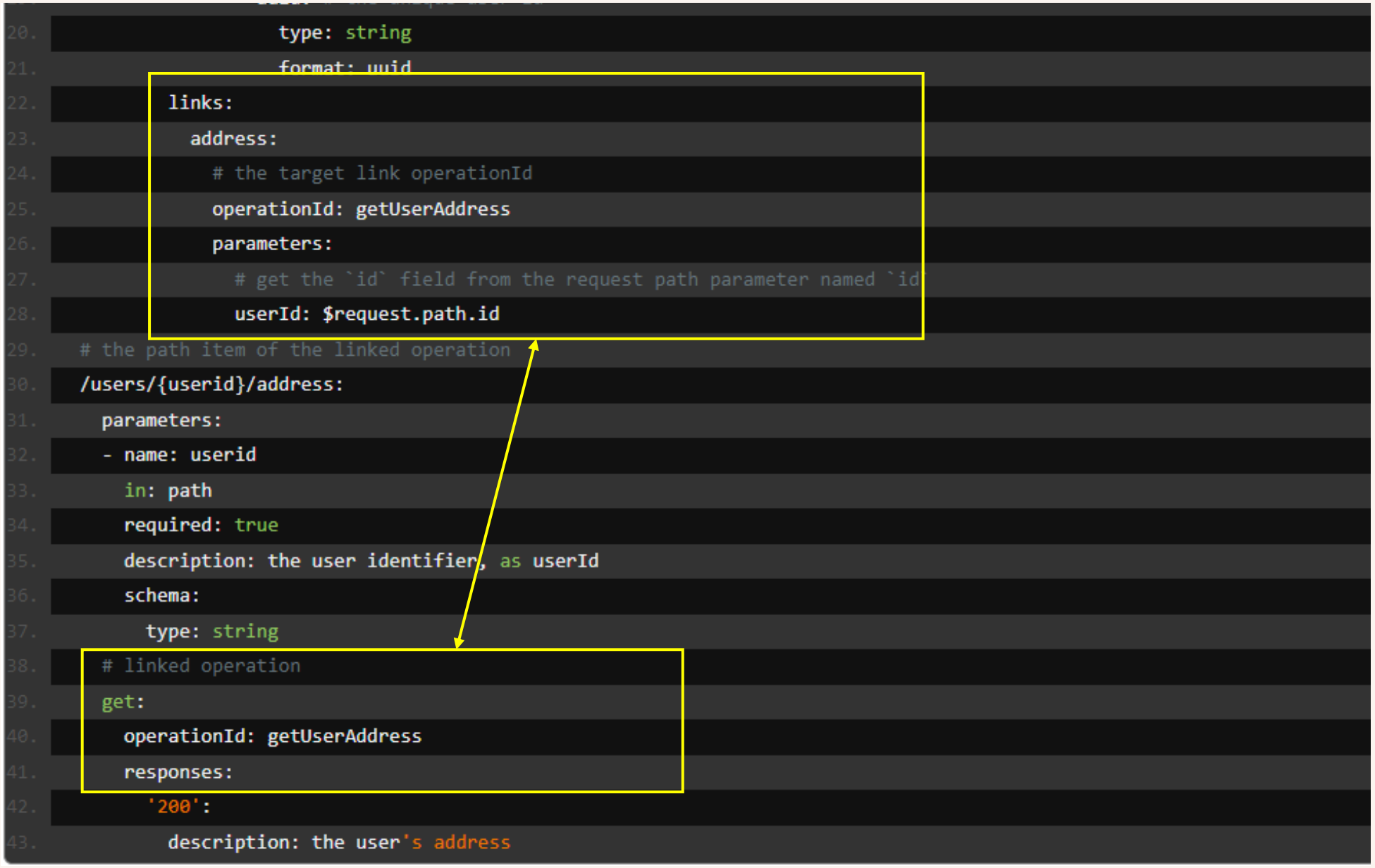
1. SET THE TARGET FIELD/OBJECT
2. LIST ALL PATHS TO REACH IT
3. BYPASS RATE LIMIT BY PARALELIZING THE REQUESTS


Endpoints with read/write access to "Target"
4-level nested object
Target: "Contact Phone Number"

4. IDOR ID FINDER

1. FIND ENDPOINTS THAT REQUIRE ID
2. DETERMINE WHERE THESE IDS ARE LEAKED
3. OBTAIN THEM AND TEST ACCESS CONTROL

4. IDOR ID FINDER



6. CONCLUSION

CONCLUSION

1. PARSING OPENAPI IS A COMPLEX TASK
2. POC TOOL IS BUGGY, BUT HAS POTENTIAL
3. NEO4J QUERIES CAN DETECT POSSIBLE INSECURE DESIGN
4. MANUAL VERIFICATION IS (STILL) NEEDED
5. PROMISING TECHNIQUE TO IMPROVE DEFENSE/ATTACK OF API