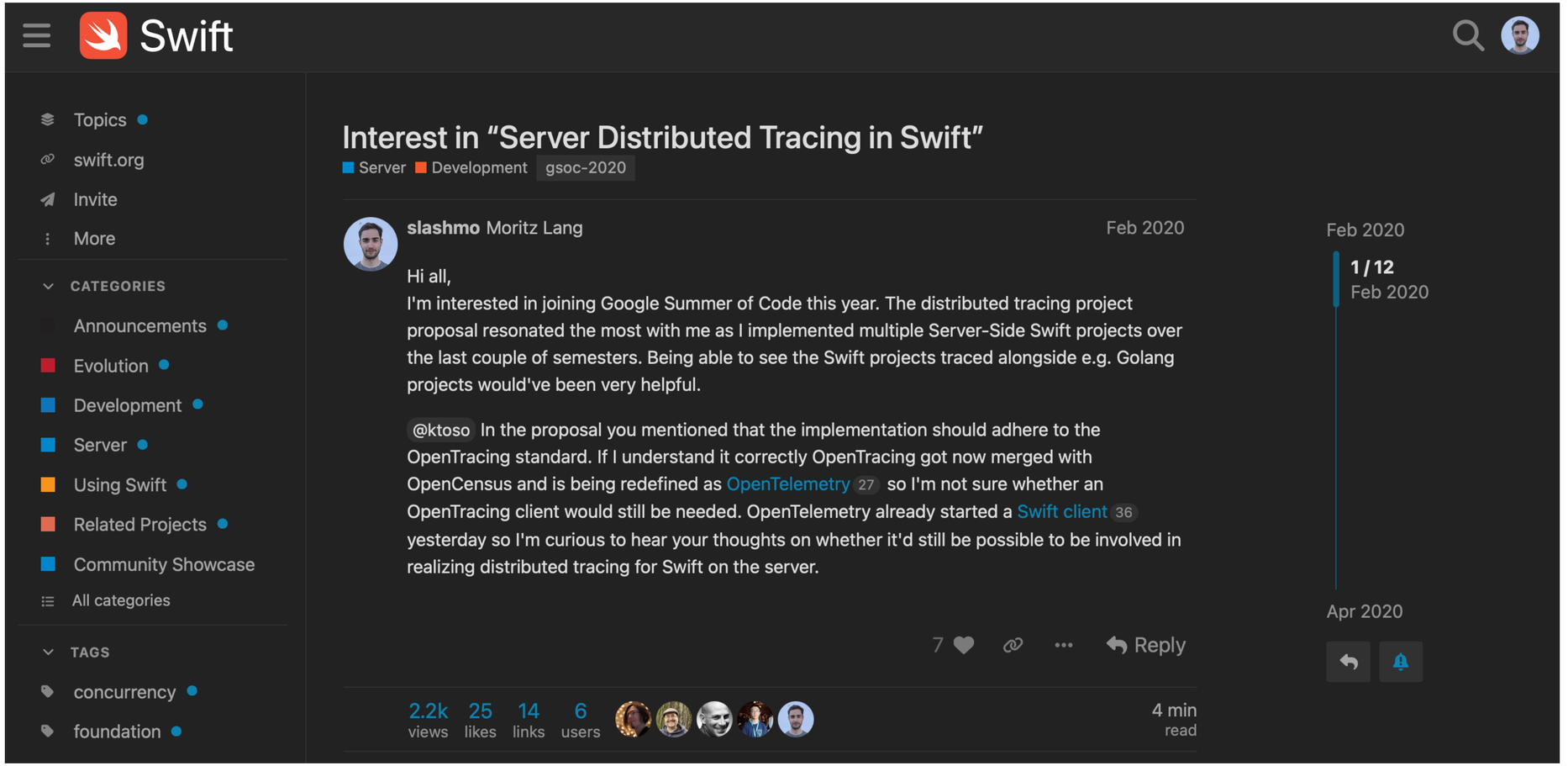
Distributed Tracing in Server-Side Swift

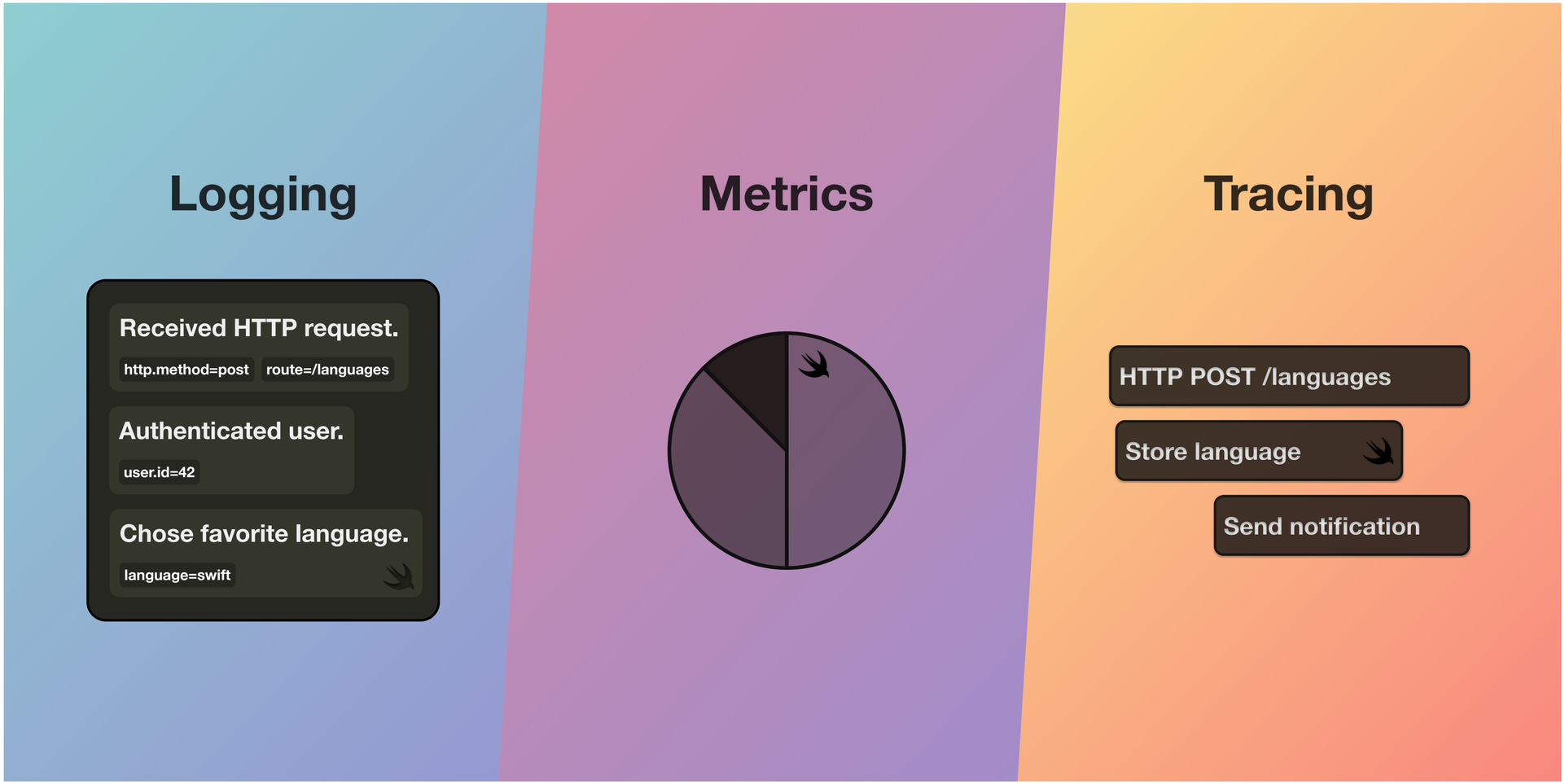
Moritz Lang @ FOSDEM 25


Swift
Topics
swift.org
Invite
More
CATEGORIES
Announcements
Evolution
Development
Server
Using Swift
Related Projects
Community Showcase
All categories
TAGS
concurrency
foundation
Interest in “Server Distributed Tracing in Swift”
Server Development gsoc-2020
slashmo Moritz Lang
Feb 2020
Hi all,
I'm interested in joining Google Summer of Code this year. The distributed tracing project proposal resonated the most with me as I implemented multiple Server-Side Swift projects over the last couple of semesters. Being able to see the Swift projects traced alongside e.g. Golang projects would've been very helpful.
@ktoso In the proposal you mentioned that the implementation should adhere to the OpenTracing standard. If I understand it correctly OpenTracing got now merged with OpenCensus and is being redefined as OpenTelemetry 27 so I'm not sure whether an OpenTracing client would still be needed. OpenTelemetry already started a Swift client 36 yesterday so I'm curious to hear your thoughts on whether it'd still be possible to be involved in realizing distributed tracing for Swift on the server.
7
Reply
2.2k views
25 likes
14 links
6 users
4 min read
Feb 2020
1 / 12
Feb 2020
Apr 2020

Agenda

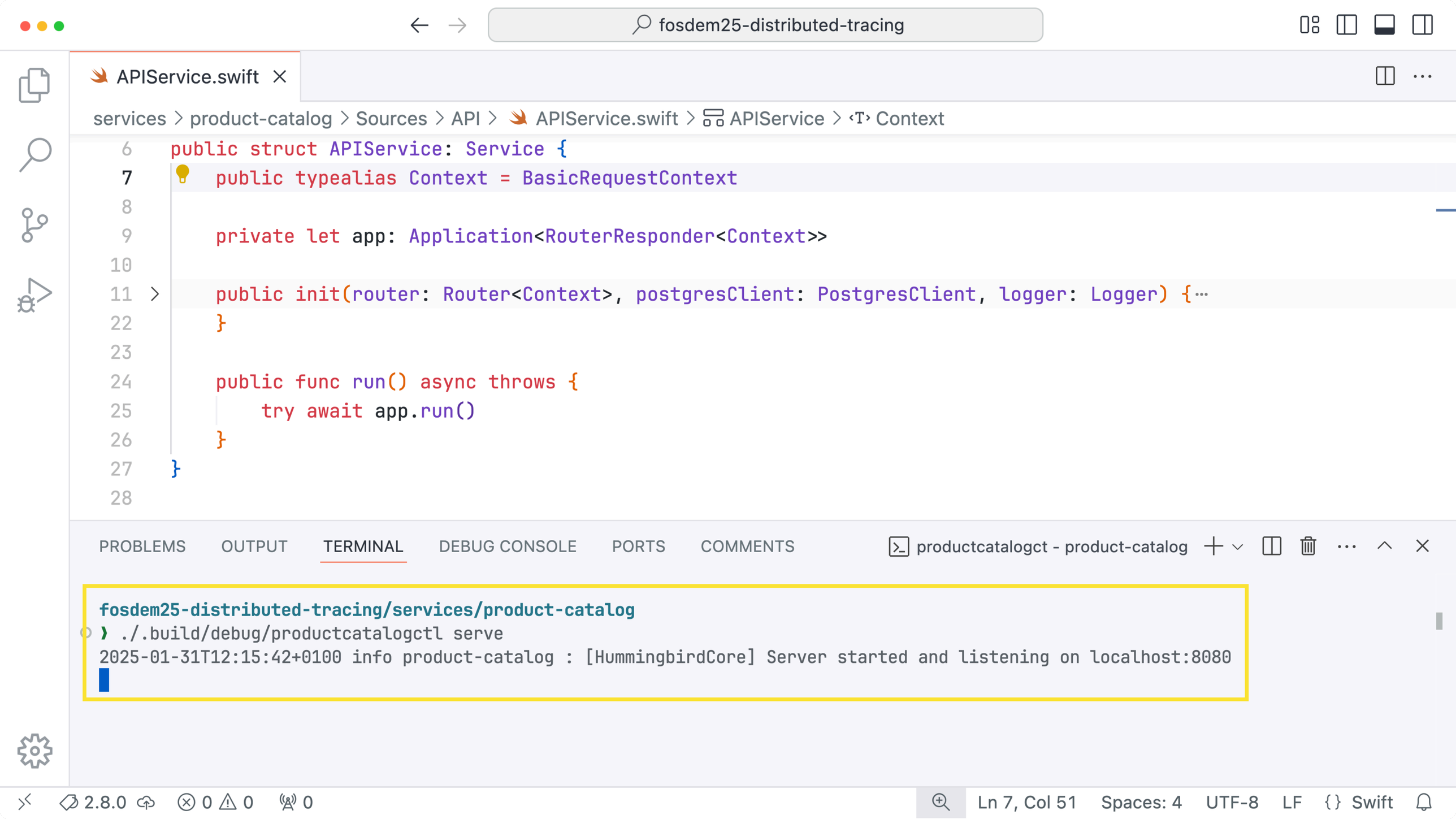
- Swift Observability
- Swift Distributed Tracing 📦
- Tracing ❤️ Logging
- OpenTelemetry & Swift OTel 📦
- Next steps

Swift Observability



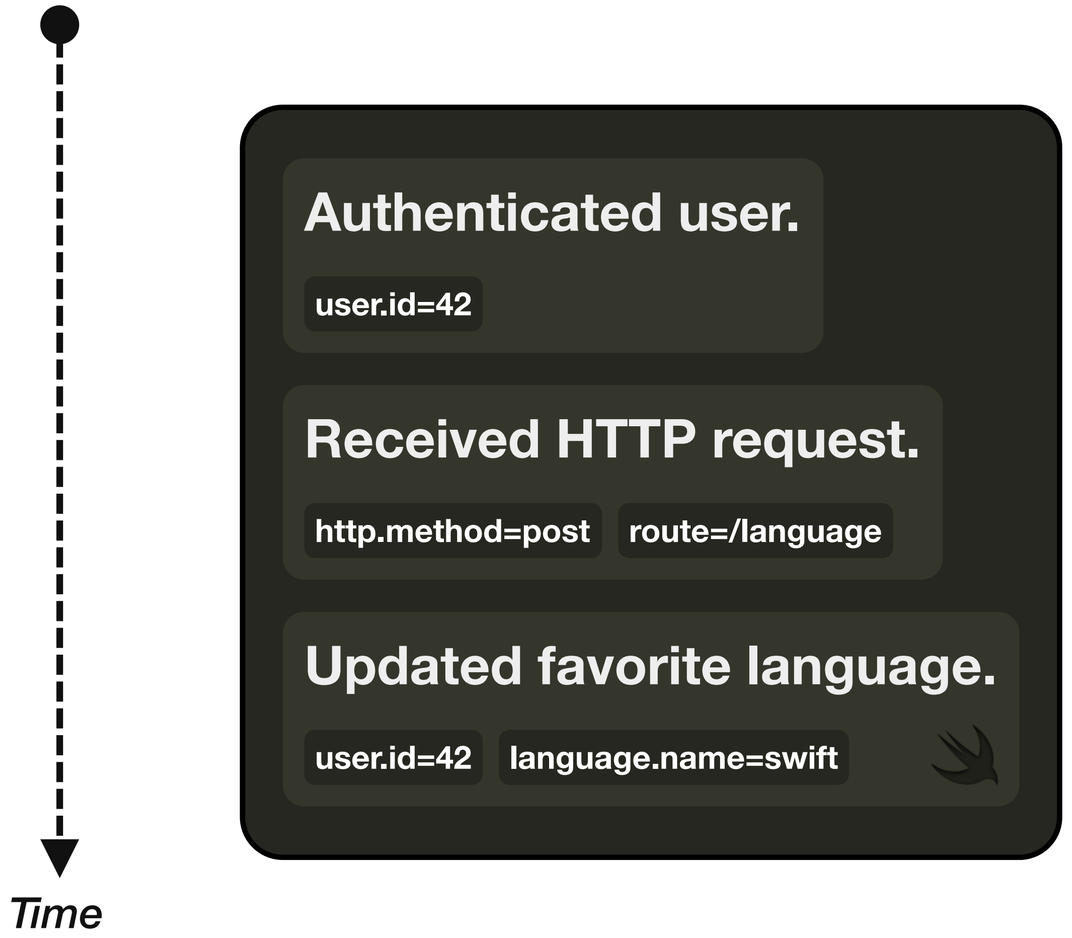
Product Catalog

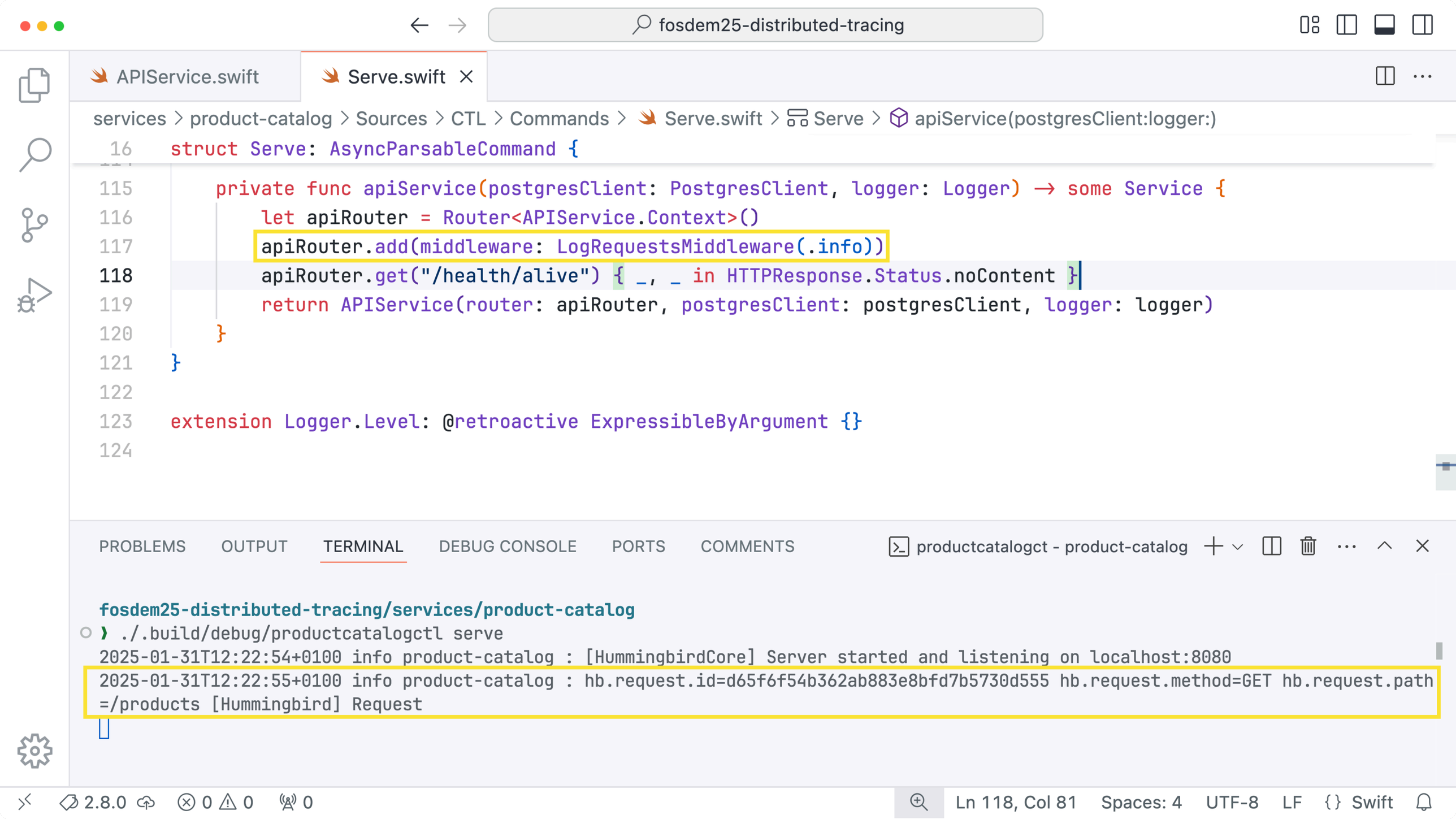
```
> curl "http://localhost:8080/products" | jq

[
  {
    "sku": "FOSDEM-2025-TSH-001",
    "price": {
      "currency_code": "EUR",
      "cents": 1500
    },
    "title": "FOSDEM T-Shirt"
  }
]

> curl "http://localhost:8080/products/FOSDEM-2025-TSH-001" | jq

{
  "sku": "FOSDEM-2025-TSH-001",
  "price": {
    "currency_code": "EUR",
    "cents": 1500
  },
  "title": "FOSDEM T-Shirt"
}
```

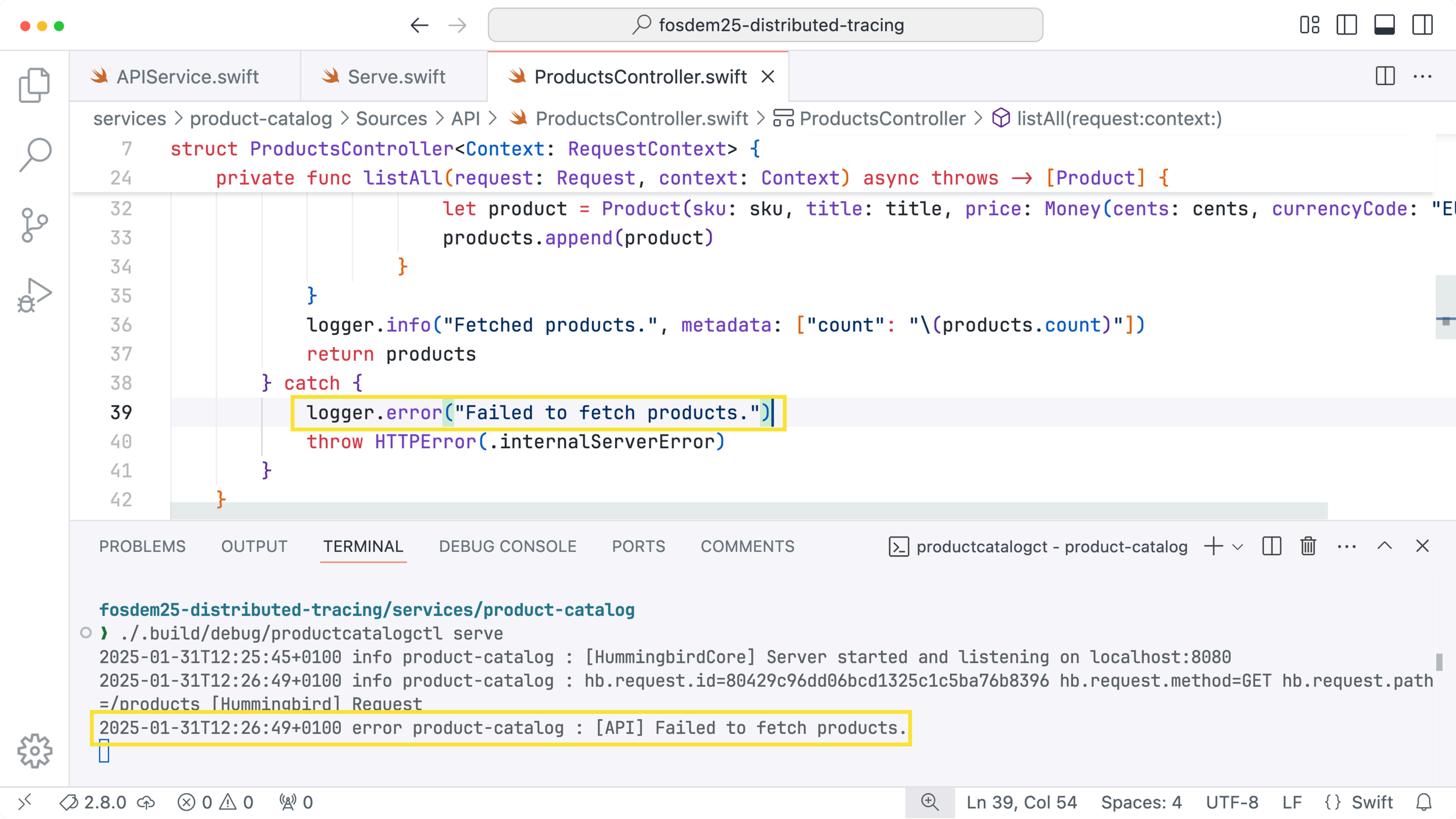

fosdem25-distributed-tracing
APIService.swift
services > product-catalog > Sources > API > APIService.swift > APIService > Context
6 public struct APIService: Service {
7 public typealias Context = BasicRequestContext
8
9 private let app: Application<RouterResponder<Context>>
10
11 public init(router: Router<Context>, postgresClient: PostgresClient, logger: Logger) {…
22 }
23
24 public func run() async throws {
25 try await app.run()
26 }
27 }
28
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE PORTS COMMENTS
productcatalogct - product-catalog
fosdem25-distributed-tracing/services/product-catalog
❯ ./.build/debug/productcatalogctl serve
2025-01-31T12:15:42+0100 info product-catalog : [HummingbirdCore] Server started and listening on localhost:8080
2.8.0
0 0
0
Ln 7, Col 51
Spaces: 4
UTF-8
LF
Swift

Logging

- Capture what happened at a specific point in time
- Detailed via metadata
- Hard to understand a specific request as a whole

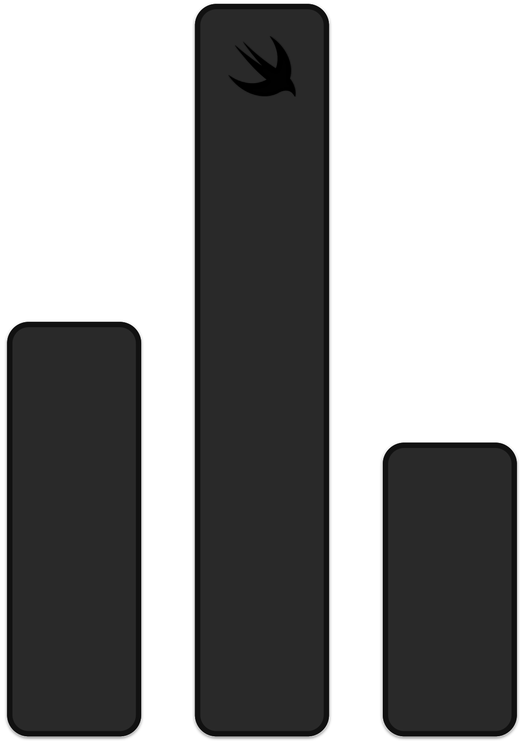


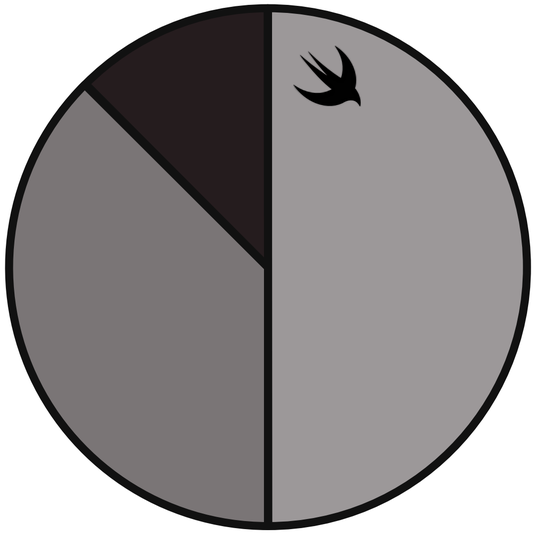

fosdem25-distributed-tracing
APIService.swift
Serve.swift
services > product-catalog > Sources > CTL > Commands > Serve.swift > Serve > apiService(postgresClient:logger:)
16 struct Serve: AsyncParsableCommand {
115 private func apiService(postgresClient: PostgresClient, logger: Logger) → some Service {
116 let apiRouter = Router<APIService.Context>()
117 apiRouter.add(middleware: LogRequestsMiddleware(.info))
118 apiRouter.get("/health/alive") { _, _ in HTTPResponse.Status.noContent }
119 return APIService(router: apiRouter, postgresClient: postgresClient, logger: logger)
120 }
121 }
122
123 extension Logger.Level: @retroactive ExpressibleByArgument {}
124
PROBLEMS
OUTPUT
TERMINAL
DEBUG CONSOLE
PORTS
COMMENTS
productcatalogct - product-catalog
fosdem25-distributed-tracing/services/product-catalog
❯ ./.build/debug/productcatalogctl serve
2025-01-31T12:22:54+0100 info product-catalog : [HummingbirdCore] Server started and listening on localhost:8080
2025-01-31T12:22:55+0100 info product-catalog : hb.request.id=d65f6f54b362ab883e8bfd7b5730d555 hb.request.method=GET hb.request.path
=/products [Hummingbird] Request
2.8.0
0 0
0
Ln 118, Col 81
Spaces: 4
UTF-8
LF
Swift

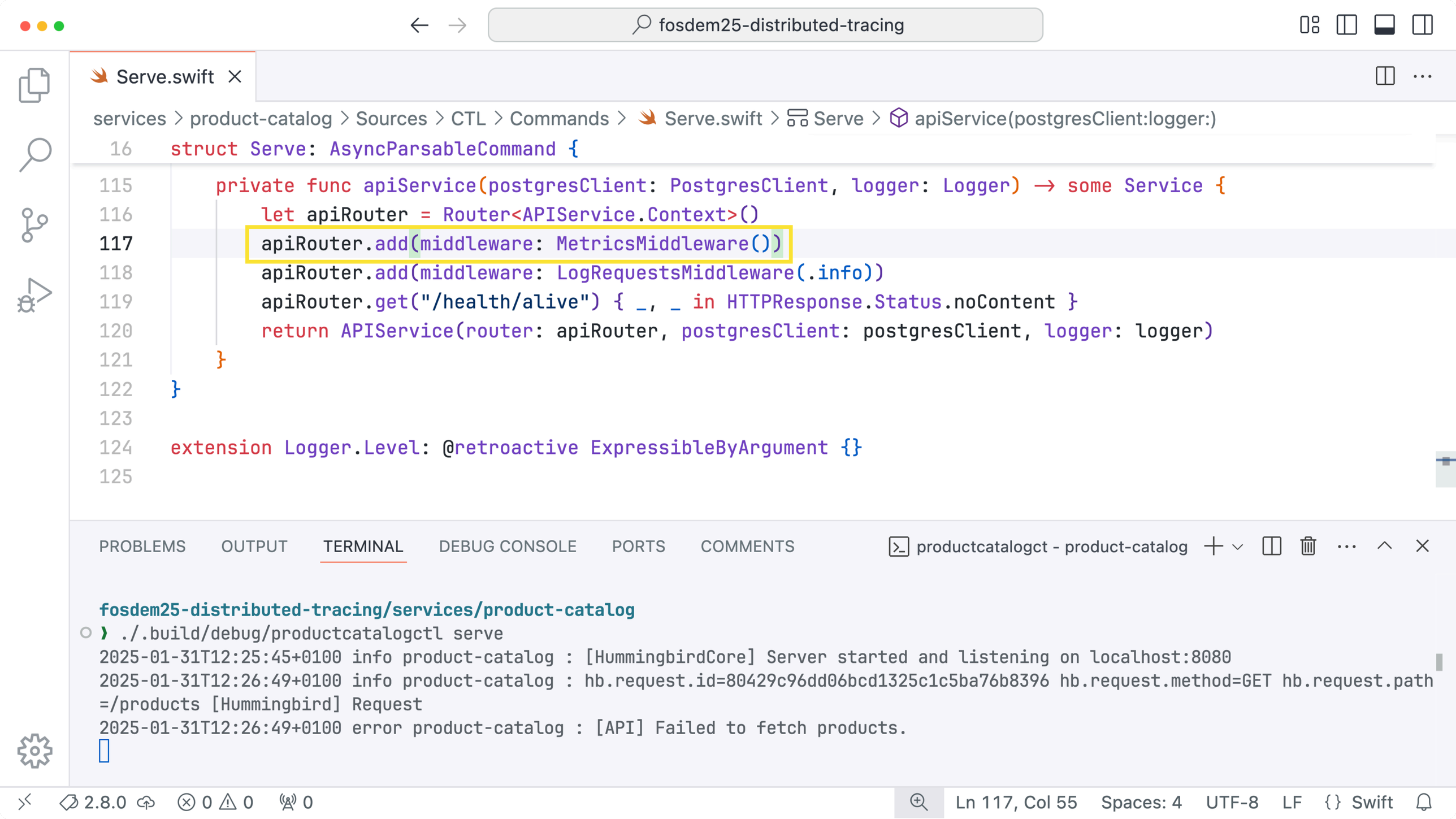

fosdem25-distributed-tracing
APIService.swift
Serve.swift
ProductsController.swift
services > product-catalog > Sources > API > ProductsController.swift > ProductsController > listAll(request:context:)
7 struct ProductsController<Context: RequestContext> {
24 private func listAll(request: Request, context: Context) async throws -> [Product] {
32 let product = Product(sku: sku, title: title, price: Money(cents: cents, currencyCode: "E
33 products.append(product)
34 }
35 }
36 logger.info("Fetched products.", metadata: ["count": "\(products.count)"])
37 return products
38 } catch {
39 logger.error("Failed to fetch products.")
40 throw HTTPError(.internalServerError)
41 }
42 }
PROBLEMS
OUTPUT
TERMINAL
DEBUG CONSOLE
PORTS
COMMENTS
productcatalogct - product-catalog
fosdem25-distributed-tracing/services/product-catalog
./.build/debug/productcatalogctl serve
2025-01-31T12:25:45+0100 info product-catalog : [HummingbirdCore] Server started and listening on localhost:8080
2025-01-31T12:26:49+0100 info product-catalog : hb.request.id=80429c96dd06bcd1325c1c5ba76b8396 hb.request.method=GET hb.request.path
=/products [Hummingbird] Request
2025-01-31T12:26:49+0100 error product-catalog : [API] Failed to fetch products.
2.8.0
0 0
0
Ln 39, Col 54
Spaces: 4
UTF-8
LF
Swift

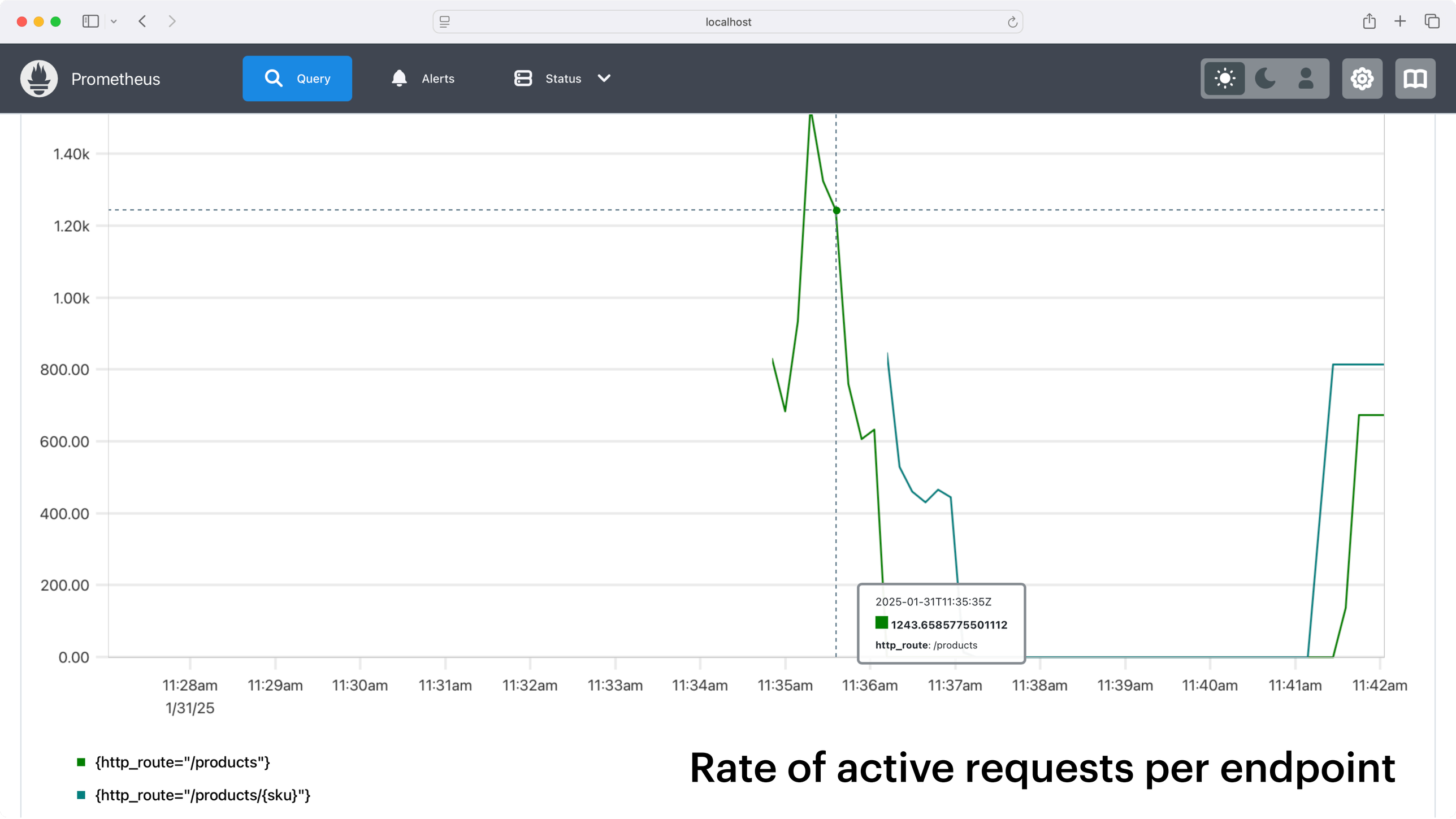
Metrics

- Aggregated
- High-level overview
- Indicate traffic spikes
- Alert when errors ramp up



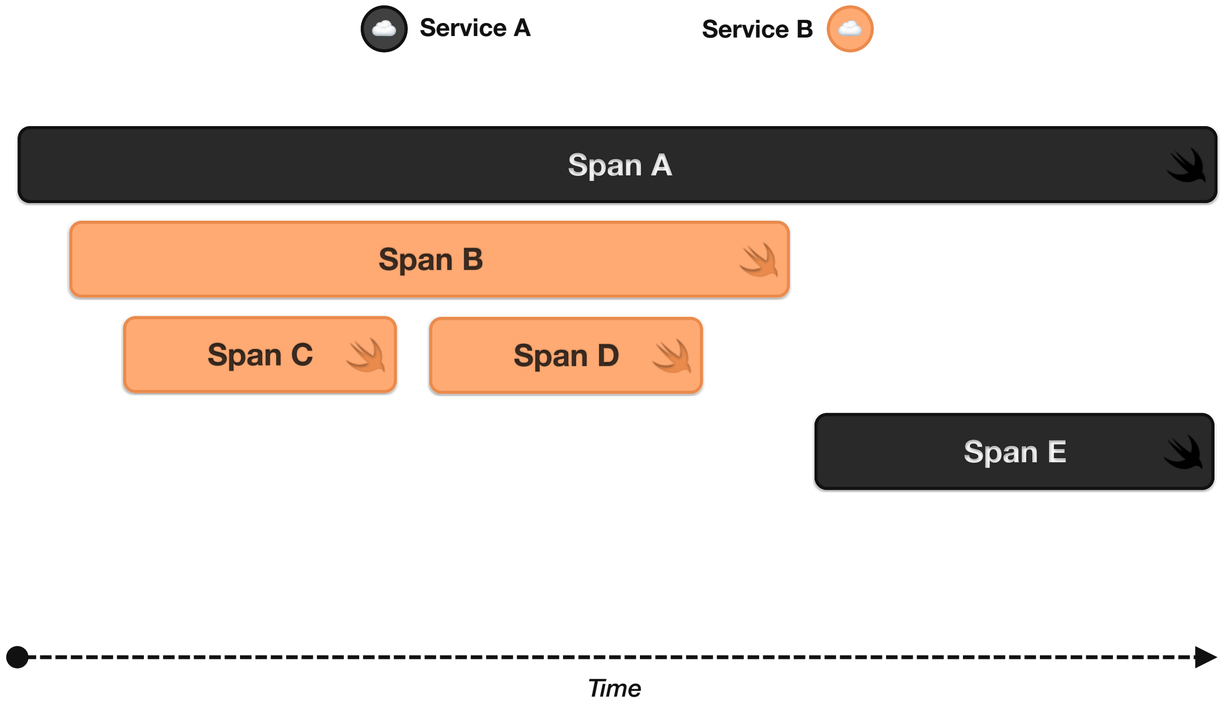


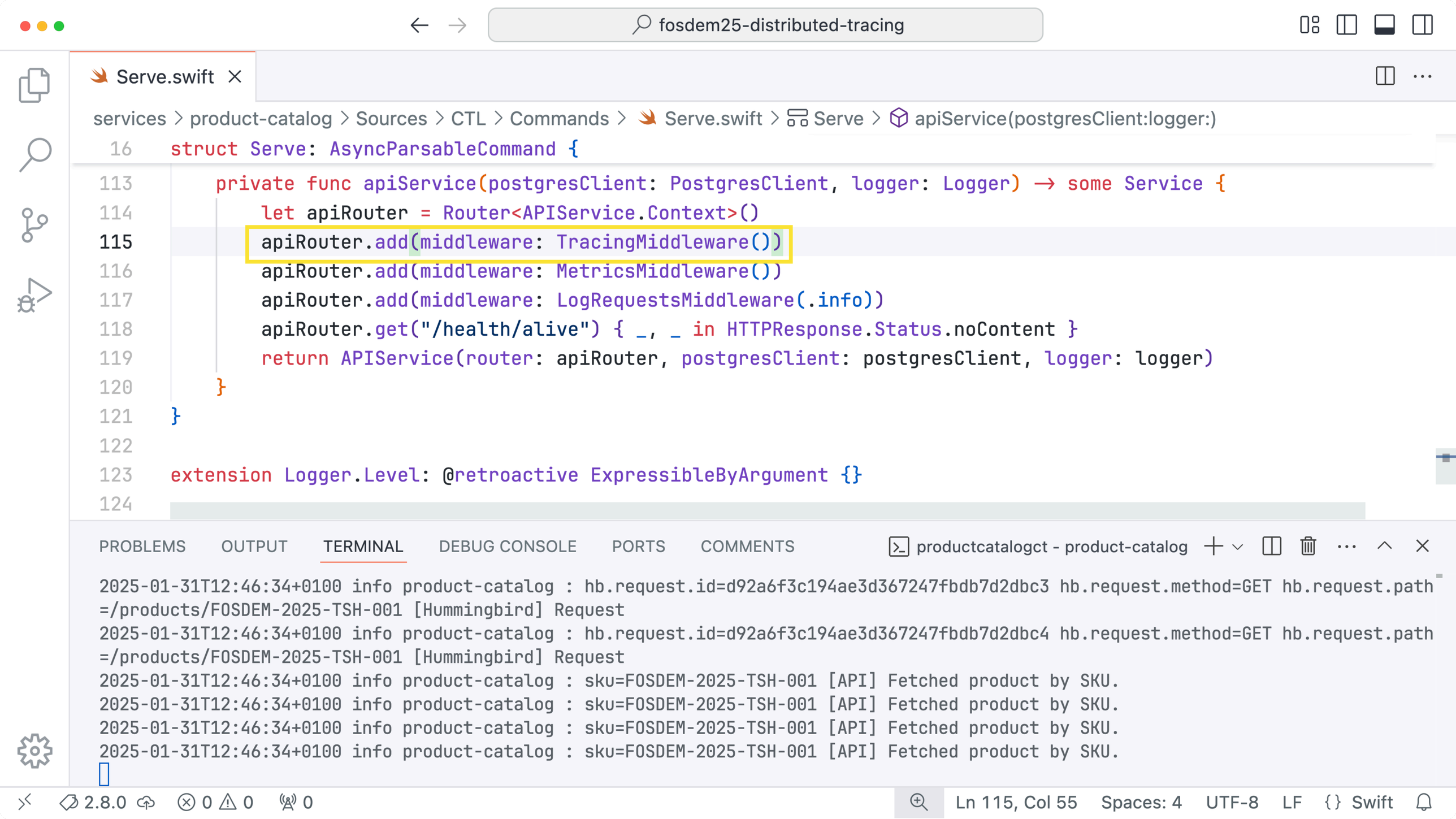

fosdem25-distributed-tracing
Serve.swift
services > product-catalog > Sources > CTL > Commands > Serve.swift > Serve > apiService(postgresClient:logger:)
16 struct Serve: AsyncParsableCommand {
115 private func apiService(postgresClient: PostgresClient, logger: Logger) → some Service {
116 let apiRouter = Router<APIService.Context>()
117 apiRouter.add(middleware: MetricsMiddleware())
118 apiRouter.add(middleware: LogRequestsMiddleware(.info))
119 apiRouter.get("/health/alive") { _, _ in HTTPResponse.Status.noContent }
120 return APIService(router: apiRouter, postgresClient: postgresClient, logger: logger)
121 }
122 }
123
124 extension Logger.Level: @retroactive ExpressibleByArgument {}
125
PROBLEMS
OUTPUT
TERMINAL
DEBUG CONSOLE
PORTS
COMMENTS
productcatalogct - product-catalog
fosdem25-distributed-tracing/services/product-catalog
❯ ./.build/debug/productcatalogctl serve
2025-01-31T12:25:45+0100 info product-catalog : [HummingbirdCore] Server started and listening on localhost:8080
2025-01-31T12:26:49+0100 info product-catalog : hb.request.id=80429c96dd06bcd1325c1c5ba76b8396 hb.request.method=GET hb.request.path
=/products [Hummingbird] Request
2025-01-31T12:26:49+0100 error product-catalog : [API] Failed to fetch products.
2.8.0
0 0
0
Ln 117, Col 55
Spaces: 4
UTF-8
LF
Swift


localhost
Prometheus
Query
Alerts
Status
1.40k
1.20k
1.00k
800.00
600.00
400.00
200.00
0.00
11:28am
1/31/25
11:29am
11:30am
11:31am
11:32am
11:33am
11:34am
11:35am
11:36am
11:37am
11:38am
11:39am
11:40am
11:41am
11:42am
2025-01-31T11:35:35Z
1243.6585775501112
http_route: /products
{http_route="/products"}
{http_route="/products/{sku}"}
Rate of active requests per endpoint

Distributed Tracing

- Specific to one request
- Trace comprised of multiple spans (operations)
- Both high-level and detailed
- Highlights where time is spent
- Sequential *and* concurrent
- Spot which operation caused a request failure




fosdem25-distributed-tracing
Serve.swift
services > product-catalog > Sources > CTL > Commands > Serve.swift > Serve > apiService(postgresClient:logger:)
16 struct Serve: AsyncParsableCommand {
113 private func apiService(postgresClient: PostgresClient, logger: Logger) -> some Service {
114 let apiRouter = Router<APIService.Context>()
115 apiRouter.add(middleware: TracingMiddleware())
116 apiRouter.add(middleware: MetricsMiddleware())
117 apiRouter.add(middleware: LogRequestsMiddleware(.info))
118 apiRouter.get("/health/alive") { _, _ in HTTPResponse.Status.noContent }
119 return APIService(router: apiRouter, postgresClient: postgresClient, logger: logger)
120 }
121 }
122
123 extension Logger.Level: @retroactive ExpressibleByArgument {}
124
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE PORTS COMMENTS
productcatalogct - product-catalog
2025-01-31T12:46:34+0100 info product-catalog : hb.request.id=d92a6f3c194ae3d367247fbdb7d2dbc3 hb.request.method=GET hb.request.path =/products/FOSDEM-2025-TSH-001 [Hummingbird] Request
2025-01-31T12:46:34+0100 info product-catalog : hb.request.id=d92a6f3c194ae3d367247fbdb7d2dbc4 hb.request.method=GET hb.request.path =/products/FOSDEM-2025-TSH-001 [Hummingbird] Request
2025-01-31T12:46:34+0100 info product-catalog : sku=FOSDEM-2025-TSH-001 [API] Fetched product by SKU.
2025-01-31T12:46:34+0100 info product-catalog : sku=FOSDEM-2025-TSH-001 [API] Fetched product by SKU.
2025-01-31T12:46:34+0100 info product-catalog : sku=FOSDEM-2025-TSH-001 [API] Fetched product by SKU.
2025-01-31T12:46:34+0100 info product-catalog : sku=FOSDEM-2025-TSH-001 [API] Fetched product by SKU.
2.8.0 0 0 0
Ln 115, Col 55 Spaces: 4 UTF-8 LF Swift

Search | Upload

Service (6)

product-catalog

Operation (4)

all

Tags

http.status_code=200 error=true

Lookback

Last Hour

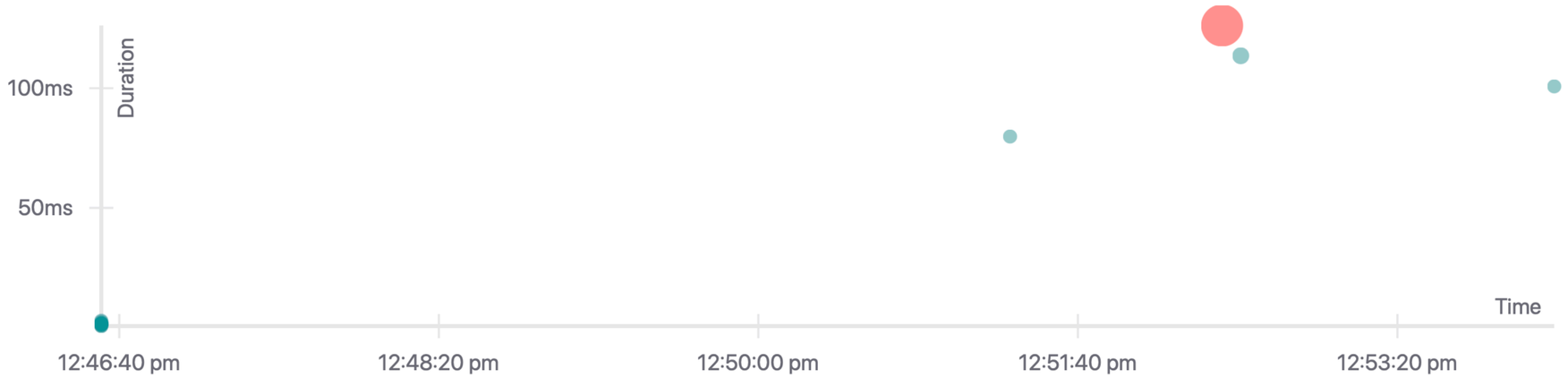
Max Duration

e.g. 1.2s, 10...

Min Duration

e.g. 1.2s, 10...

Limit Results



20 Traces

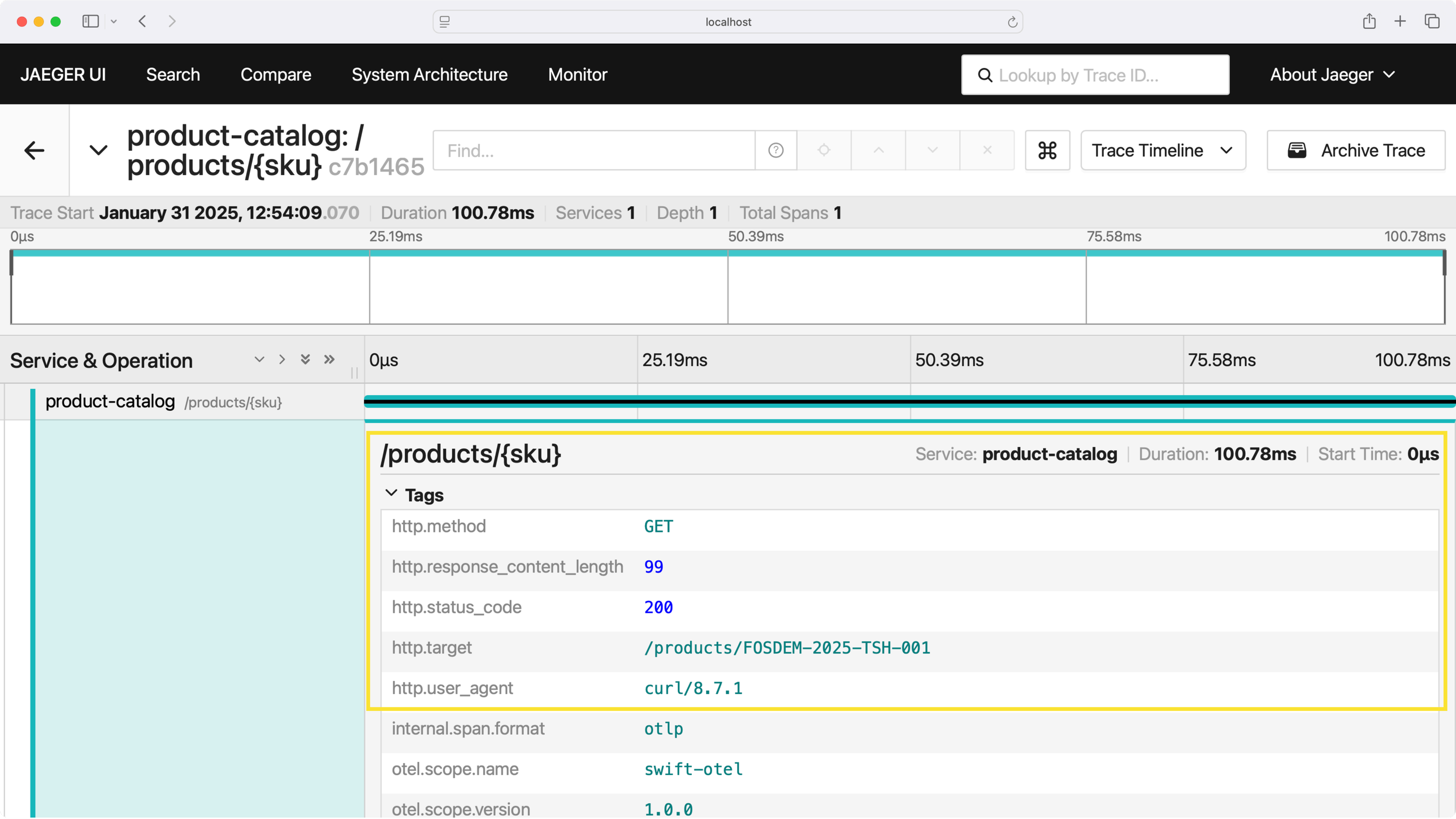
Sort: Most Recent | Download Results | Deep Dependency Graph

Compare traces by selecting result items

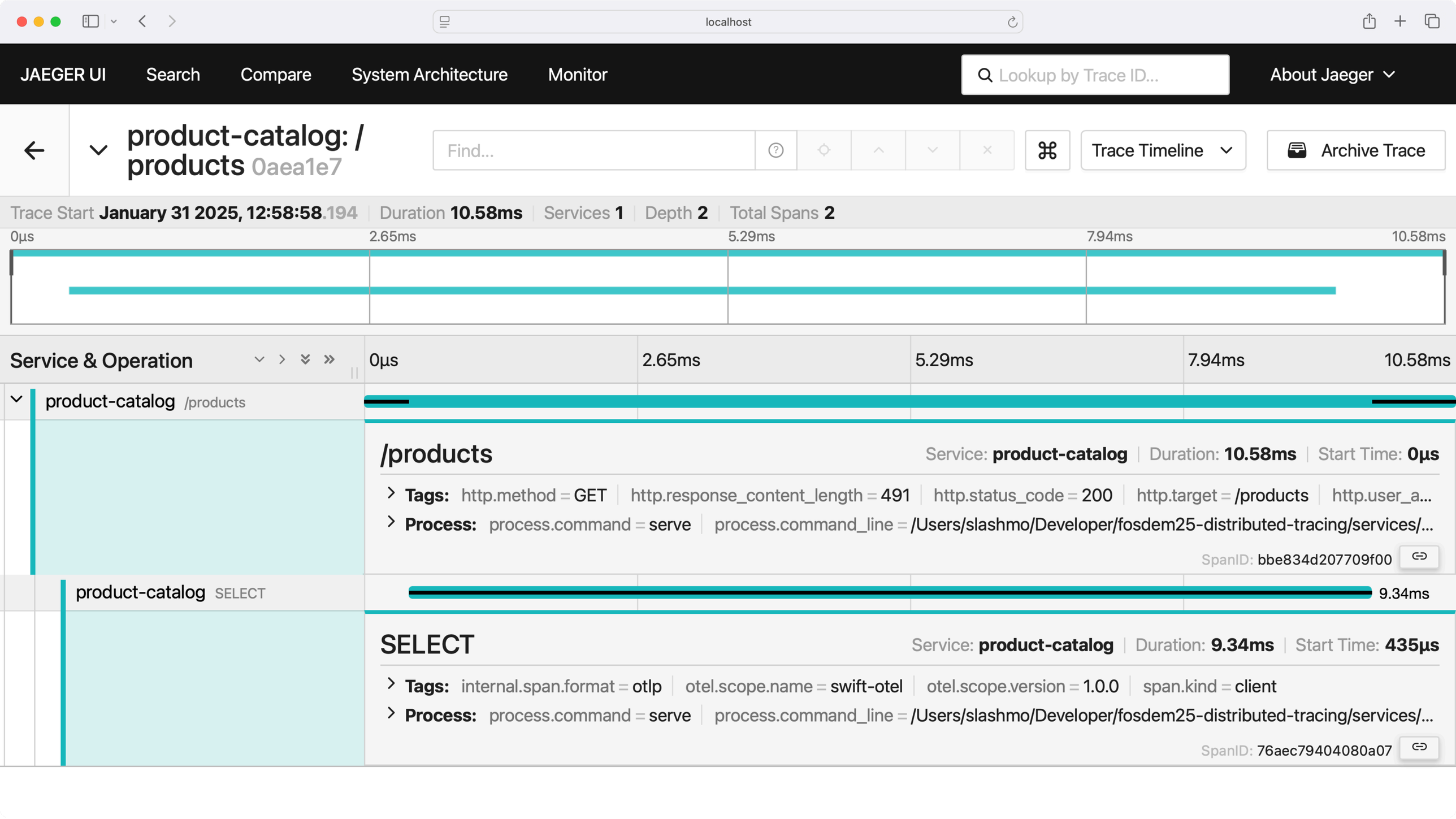
product-catalog: /products/{sku} c7b1465 — 100.78ms

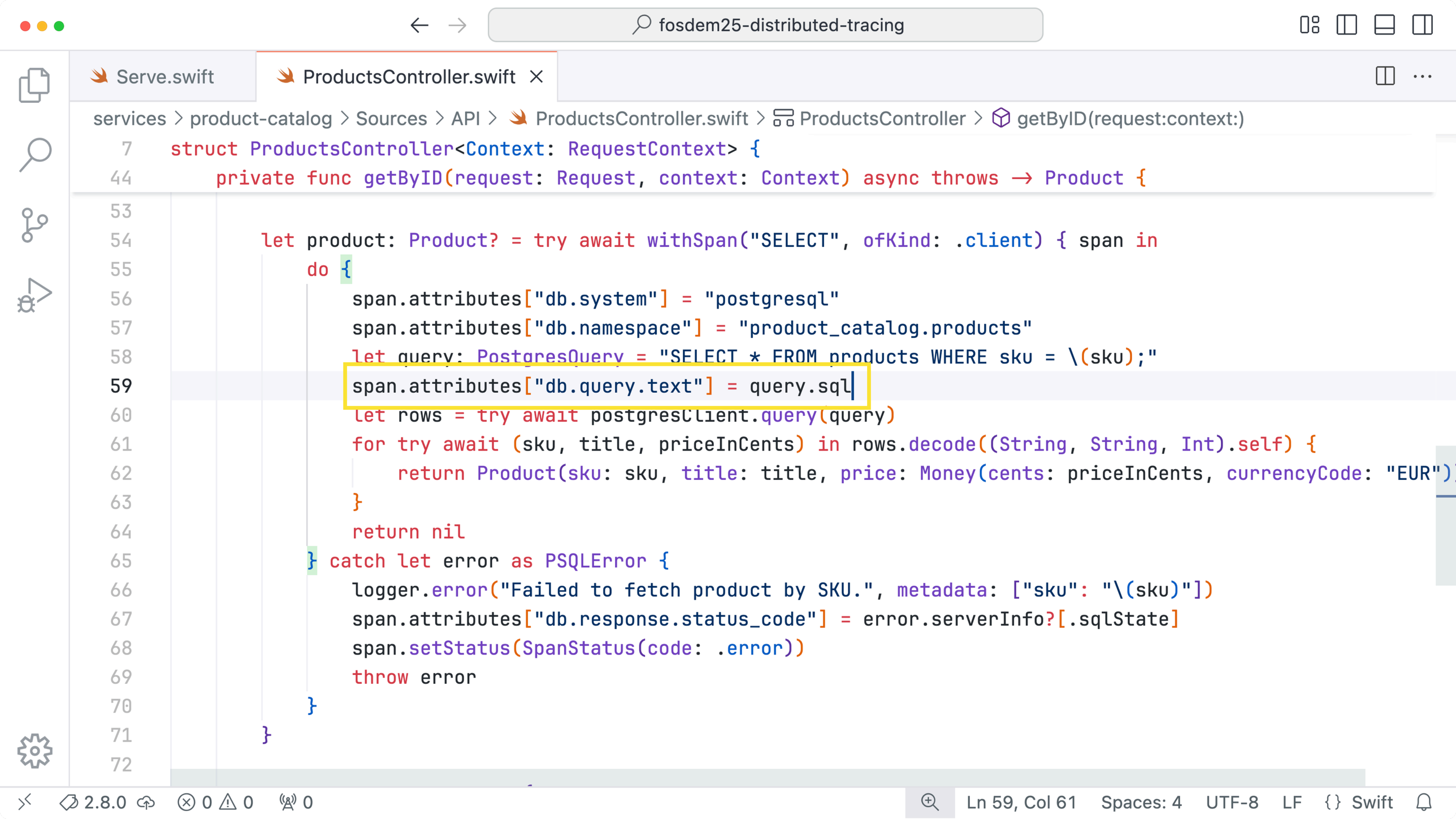
1 Span | product-catalog (1)

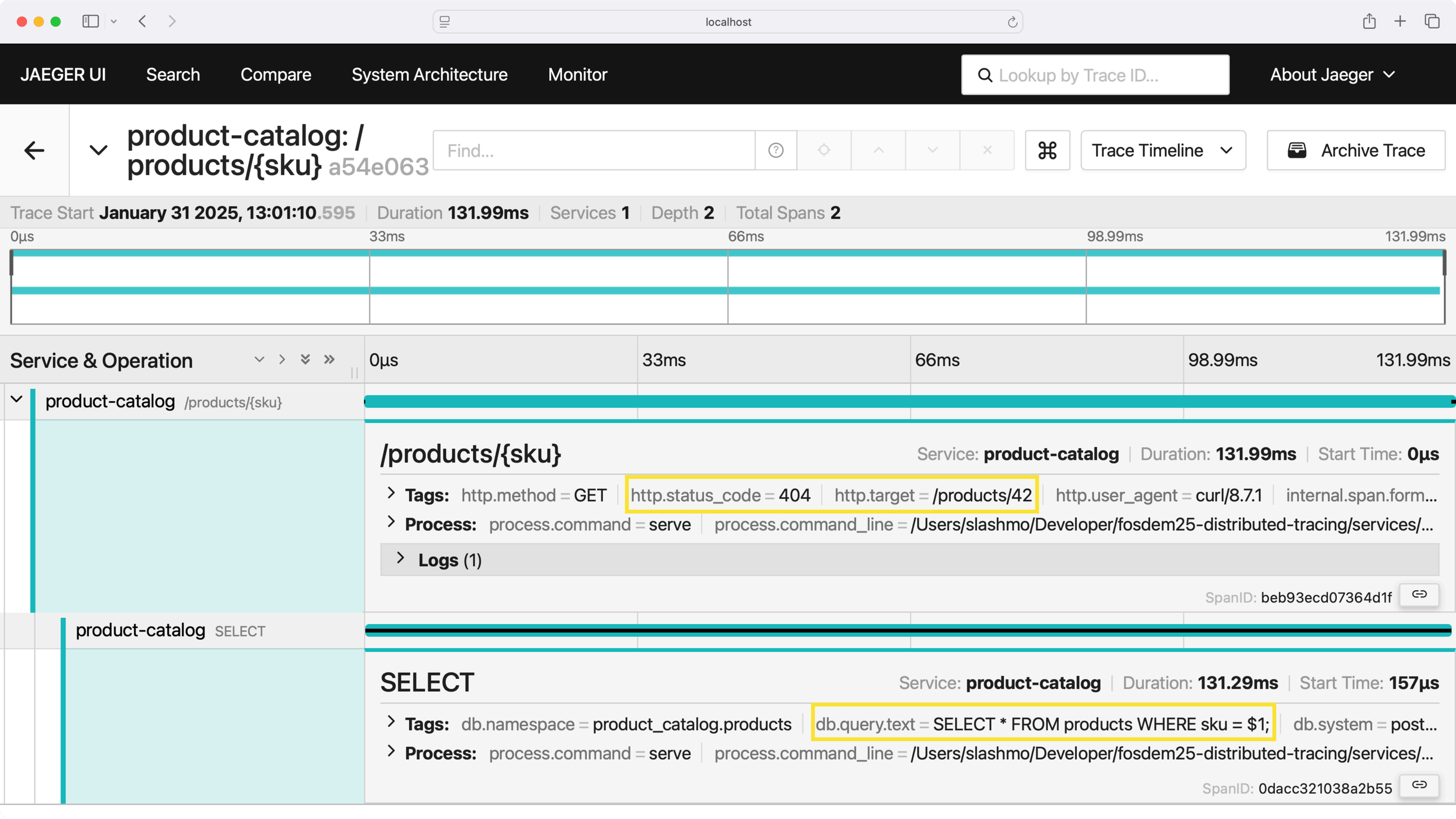
Today
12:54:09 pm
a minute ago


localhost
JAEGER UI
Search
Compare
System Architecture
Monitor
Lookup by Trace ID...
About Jaeger
product-catalog: /products/{sku} c7b1465
Find...
Trace Timeline
Archive Trace
Trace Start January 31 2025, 12:54:09.070 | Duration 100.78ms | Services 1 | Depth 1 | Total Spans 1
0µs
25.19ms
50.39ms
75.58ms
100.78ms
Service & Operation
product-catalog /products/{sku}
/products/{sku}
Service: product-catalog | Duration: 100.78ms | Start Time: 0µs
Tags
http.method GET
http.response_content_length 99
http.status_code 200
http.target /products/FOSDEM-2025-TSH-001
http.user_agent curl/8.7.1
internal.span.format otlp
otel.scope.name swift-otel
otel.scope.version 1.0.0

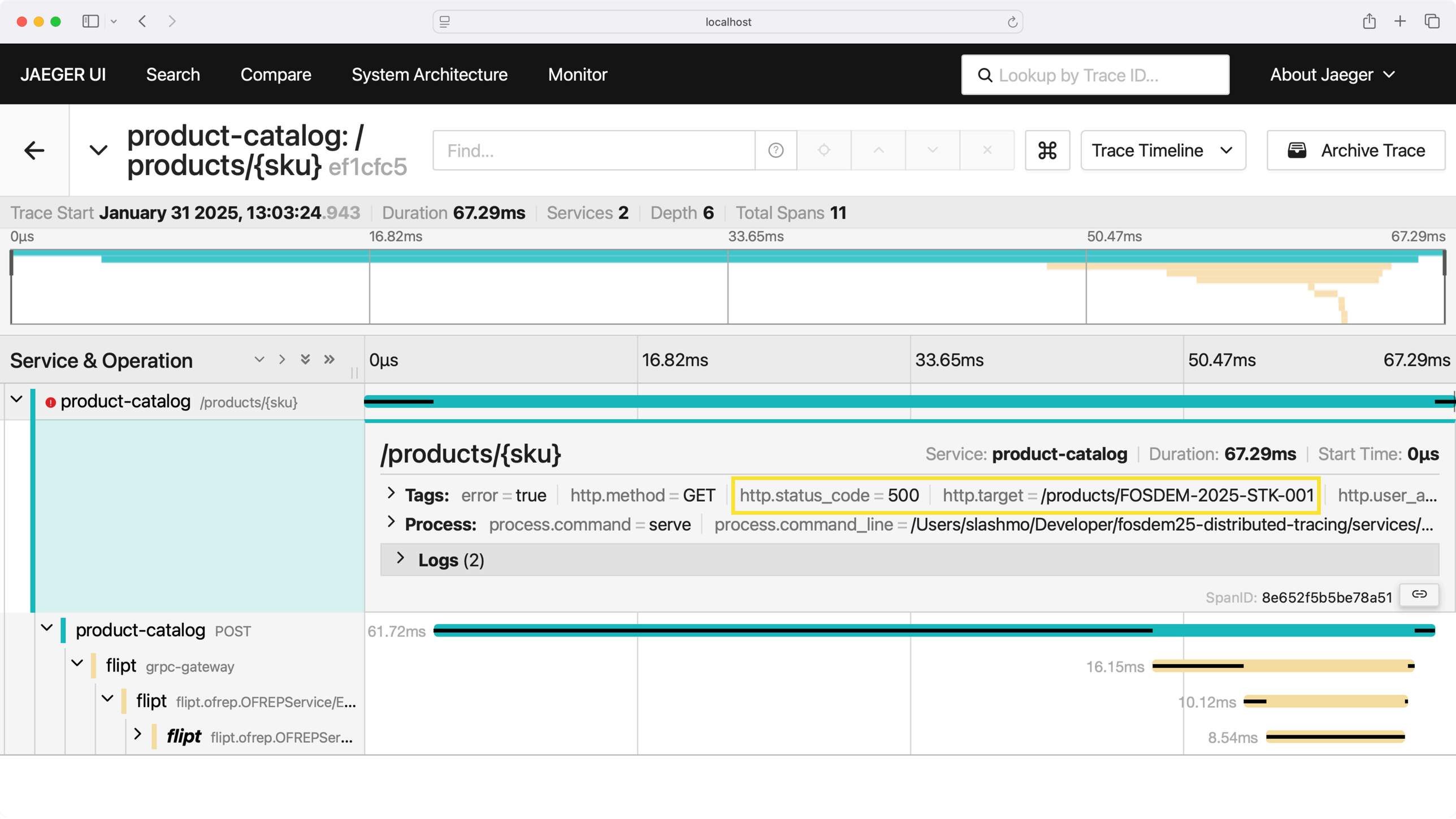

fosdem25-distributed-tracing
Serve.swift
ProductsController.swift
services > product-catalog > Sources > API > ProductsController.swift > ProductsController > listAll(request:context:)
uct ProductsController<Context: RequestContext> {
private func listAll(request: Request, context: Context) async throws -> [Product] {
do {
let products = try await withSpan("SELECT", ofKind: .client) { span in
try await postgresClient
.query("SELECT * FROM products")
.decode((String, String, Int).self)
.reduce(into: [Product]()) { (products, row) in
let (sku, title, cents) = row
let product = Product(sku: sku, title: title, price: Money(cents: cents, currencyCode: "EUR"
products.append(product)
}
}
logger.info("Fetched products.", metadata: ["count": "\(products.count)"])
return products
} catch {
logger.error("Failed to fetch products.")
throw HTTPError(.internalServerError)
}
}
2.8.0
0 0
0
Ln 26, Col 73
Spaces: 4
UTF-8
LF
Swift


localhost
JAEGER UI
Search
Compare
System Architecture
Monitor
Lookup by Trace ID...
About Jaeger
product-catalog: /products 0aea1e7
Find...
Trace Timeline
Archive Trace
Trace Start January 31 2025, 12:58:58.194 | Duration 10.58ms | Services 1 | Depth 2 | Total Spans 2
0µs
2.65ms
5.29ms
7.94ms
10.58ms
Service & Operation
product-catalog /products
/products
Service: product-catalog | Duration: 10.58ms | Start Time: 0µs
Tags: http.method = GET | http.response_content_length = 491 | http.status_code = 200 | http.target = /products | http.user_a...
Process: process.command = serve | process.command_line = /Users/slashmo/Developer/fosdem25-distributed-tracing/services/...
SpanID: bbe834d207709f00
product-catalog SELECT
9.34ms
SELECT
Service: product-catalog | Duration: 9.34ms | Start Time: 435µs
Tags: internal.span.format = otlp | otel.scope.name = swift-otel | otel.scope.version = 1.0.0 | span.kind = client
Process: process.command = serve | process.command_line = /Users/slashmo/Developer/fosdem25-distributed-tracing/services/...
SpanID: 76aec79404080a07


fosdem25-distributed-tracing
Serve.swift
ProductsController.swift
services > product-catalog > Sources > API > ProductsController.swift > ProductsController > getByID(request:context:)
7 struct ProductsController<Context: RequestContext> {
44 private func getByID(request: Request, context: Context) async throws -> Product {
53
54 let product: Product? = try await withSpan("SELECT", ofKind: .client) { span in
55 do {
56 span.attributes["db.system"] = "postgresql"
57 span.attributes["db.namespace"] = "product_catalog.products"
58 let query: PostgresQuery = "SELECT * FROM products WHERE sku = \(sku);"
59 span.attributes["db.query.text"] = query.sql
60 let rows = try await postgresClient.query(query)
61 for try await (sku, title, priceInCents) in rows.decode((String, String, Int).self) {
62 return Product(sku: sku, title: title, price: Money(cents: priceInCents, currencyCode: "EUR"))
63 }
64 return nil
65 } catch let error as PSQLError {
66 logger.error("Failed to fetch product by SKU.", metadata: ["sku": "\(sku)"])
67 span.attributes["db.response.status_code"] = error.serverInfo?[.sqlState]
68 span.setStatus(SpanStatus(code: .error))
69 throw error
70 }
71 }
72
2.8.0
0 0
0
Ln 59, Col 61
Spaces: 4
UTF-8
LF
Swift


localhost
JAEGER UI
Search
Compare
System Architecture
Monitor
Lookup by Trace ID...
About Jaeger
product-catalog: /products/{sku} a54e063
Find...
Trace Timeline
Archive Trace
Trace Start January 31 2025, 13:01:10.595 | Duration 131.99ms | Services 1 | Depth 2 | Total Spans 2
0µs
33ms
66ms
98.99ms
131.99ms
Service & Operation
product-catalog /products/{sku}
/products/{sku}
Service: product-catalog | Duration: 131.99ms | Start Time: 0µs
Tags: http.method = GET | http.status_code = 404 | http.target = /products/42 | http.user_agent = curl/8.7.1 | internal.span.form...
Process: process.command = serve | process.command_line = /Users/slashmo/Developer/fosdem25-distributed-tracing/services/...
Logs (1)
SpanID: beb93ecd07364d1f
product-catalog SELECT
SELECT
Service: product-catalog | Duration: 131.29ms | Start Time: 157µs
Tags: db.namespace = product_catalog.products | db.query.text = SELECT * FROM products WHERE sku = $1; | db.system = post...
Process: process.command = serve | process.command_line = /Users/slashmo/Developer/fosdem25-distributed-tracing/services/...
SpanID: 0dacc321038a2b55


fosdem25-distributed-tracing
Serve.swift
ProductsController.swift
services > product-catalog > Sources > API > ProductsController.swift > ProductsController > getByID(request:context:)
7 struct ProductsController<Context: RequestContext> {
44 private func getByID(request: Request, context: Context) async throws -> Product {
45 let sku = try context.parameters.require("sku")
46
47 if sku == "FOSDEM-2025-STK-001", await OpenFeatureSystem.client().value(
48 for: "productCatalogFailure",
49 defaultingTo: false
50) {
51 throw HTTPError(.internalServerError)
52 }
53
54 let product: Product? = try await withSpan("SELECT", ofKind: .client) { span in
55 do {
56 span.attributes["db.system"] = "postgresql"
57 span.attributes["db.namespace"] = "product_catalog.products"
58 let query: PostgresQuery = "SELECT * FROM products WHERE sku = \(sku);"
59 span.attributes["db.query.text"] = query.sql
60 let rows = try await postgresClient.query(query)
61 for try await (sku, title, priceInCents) in rows.decode((String, String, Int).self) {
62 return Product(sku: sku, title: title, price: Money(cents: priceInCents, currencyCode: "EUR"))
63 }
64 return nil
2.8.0
0 0
0
Ln 46, Col 1
Spaces: 4
UTF-8
LF
Swift


localhost
JAEGER UI
Search
Compare
System Architecture
Monitor
Lookup by Trace ID...
About Jaeger
product-catalog: /products/{sku} ef1cfc5
Find...
Trace Timeline
Archive Trace
Trace Start January 31 2025, 13:03:24.943 | Duration 67.29ms | Services 2 | Depth 6 | Total Spans 11
0µs
16.82ms
33.65ms
50.47ms
67.29ms
Service & Operation
product-catalog /products/{sku}
/products/{sku}
Service: product-catalog | Duration: 67.29ms | Start Time: 0µs
Tags: error = true | http.method = GET | http.status_code = 500 | http.target = /products/FOSDEM-2025-STK-001 | http.user_a...
Process: process.command = serve | process.command_line = /Users/slashmo/Developer/fosdem25-distributed-tracing/services/...
Logs (2)
SpanID: 8e652f5b5be78a51
product-catalog POST
61.72ms
flipt grpc-gateway
16.15ms
flipt flipt.ofrep.OFREPService/E...
10.12ms
flipt flipt.ofrep.OFREPSer...
8.54ms

Search | Upload

Service (6)

product-catalog

Operation (4)

all

Tags

http.status_code=200 error=true

Lookback

Last Hour

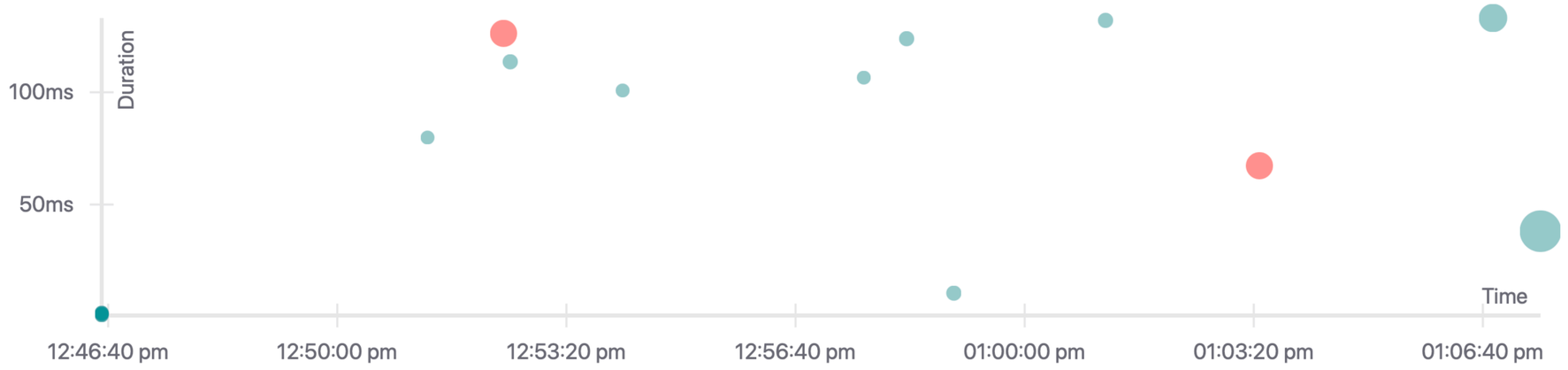
Max Duration

e.g. 1.2s, 10...

Min Duration

e.g. 1.2s, 10...

Limit Results



20 Traces

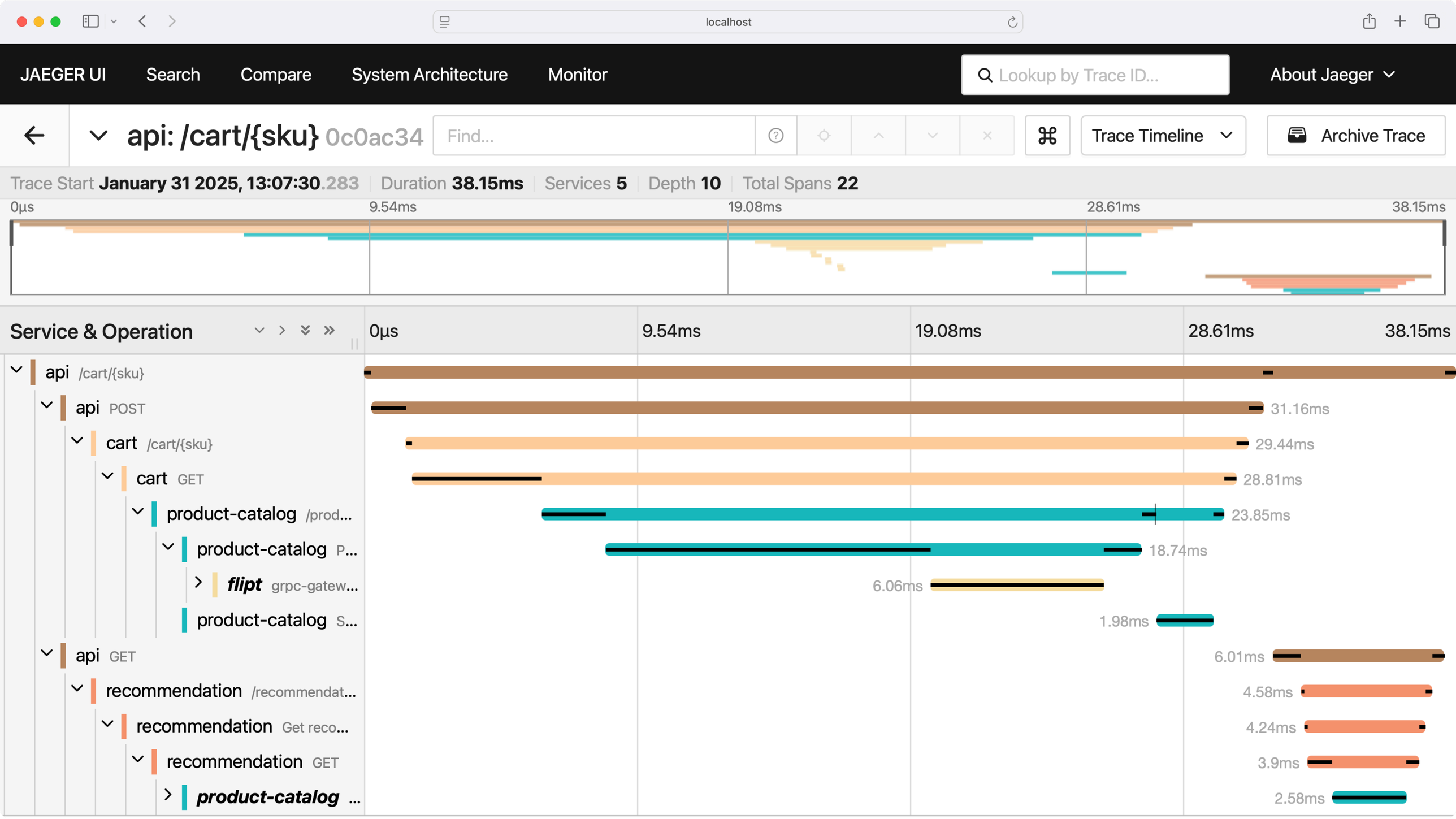
Sort: Most Recent | Download Results | Deep Dependency Graph

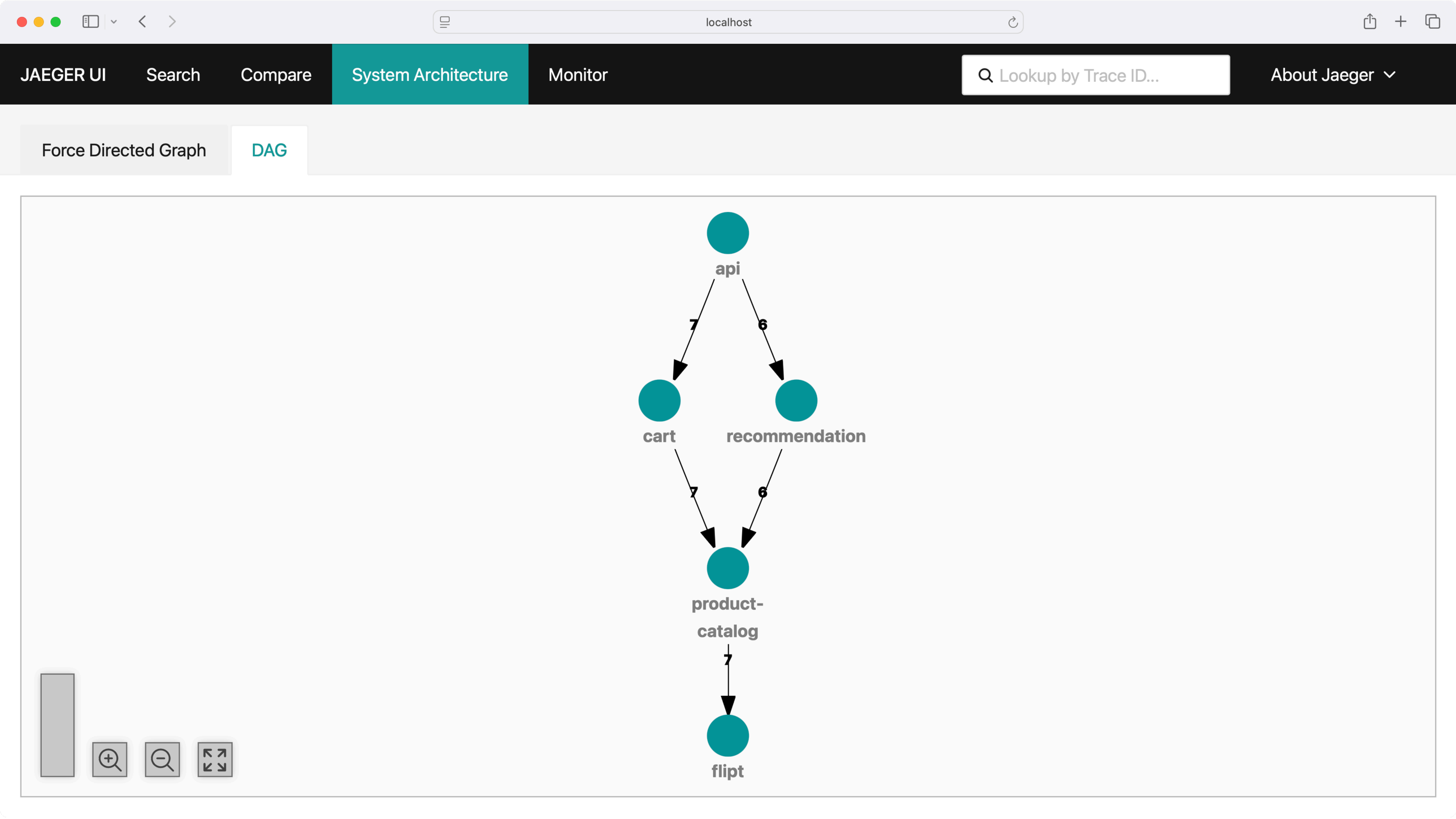
Compare traces by selecting result items

api: /cart/{sku} 0c0ac34 — 38.15ms

22 Spans | api (3) | cart (2) | flipt (9) | product-catalog (5) | recommendation (3)

Today 1:07:30 pm
a few seconds ago


localhost
JAEGER UI
Search
Compare
System Architecture
Monitor
Lookup by Trace ID...
About Jaeger
api: /cart/{sku} 0c0ac34
Find...
Trace Timeline
Archive Trace
Trace Start January 31 2025, 13:07:30.283
Duration 38.15ms
Services 5
Depth 10
Total Spans 22
0µs
9.54ms
19.08ms
28.61ms
38.15ms
Service & Operation
api /cart/{sku}
api POST
cart /cart/{sku}
cart GET
product-catalog /prod...
product-catalog P...
flipt grpc-gatew...
product-catalog S...
api GET
recommendation /recommendat...
recommendation Get reco...
recommendation GET
product-catalog ...
31.16ms
29.44ms
28.81ms
23.85ms
18.74ms
6.06ms
1.98ms
6.01ms
4.58ms
4.24ms
3.9ms
2.58ms


localhost
JAEGER UI
Search
Compare
System Architecture
Monitor
Lookup by Trace ID...
About Jaeger
Force Directed Graph
DAG
api
7
6
cart
recommendation
7
6
product-
catalog
7
flipt

Swift Distributed Tracing 📦

github.com/apple/swift-distributed-tracing

- Similar to swift-log and swift-metrics
- Provides *only* the interface
- Three target audiences
  - Library Authors
  - Instrumentation Authors
  - Application Developers

Library Authors

- Agnostic of the specific tracer
- Use withSpan and similar APIs

Instrumentation Authors

- Conform to the Tracer protocol
- Export the recorded spans to a specific Distributed Tracing system

Application Developers

- Select one Tracer implementation
- Use libraries that support Swift Distributed Tracing
- *Optionally* create additional spans

Tracer

- Associated Span type
- Ability to create spans

```
25  public protocol Tracer: LegacyTracer {
26      /// The concrete type of span this tracer will be producing/
27      associatedtype Span: Tracing.Span
28
29  >   /// Start a new ``Span`` with the given `ServiceContext`.…
53      func startSpan<Instant: TracerInstant>(
54          _ operationName: String,
55          context: @autoclosure () → ServiceContext,
56          ofKind kind: SpanKind,
57          at instant: @autoclosure () → Instant,
58          function: String,
59          file fileID: String,
60          line: UInt
61      ) → Self.Span
62
63  >   /// Retrieve the recording span for the given `ServiceContext`.…
69      func activeSpan(identifiedBy context: ServiceContext) → Span?
70  }
```

Span

- Mutable until finished
- Must be finished by calling *end(instant:)*
- Uniquely identified via ServiceContext

```
34  public protocol Span: Sendable {
35      var context: ServiceContext { get }
36
37      var operationName: String {…
40      }
41
42      func setStatus(_ status: SpanStatus)
43
44      func addEvent(_ event: SpanEvent)
45
46      func recordError<Instant: TracerInstant>(…
50      )
51
52      var attributes: SpanAttributes {…
55      }
56
57      var isRecording: Bool { get }
58
59      func addLink(_ link: SpanLink)
60
61      func end<Instant: TracerInstant>(at instant: @autoclosure () → Instant)
62  }
```

ServiceContext

- Contains span/trace ID
- Stored in task-local
- Automatically create child spans

```
try await withSpan("parent") { parentSpan in
    let value = try await withSpan("child") { childSpan in
        try await nestedOperation()
    }
    try await process(value)
}
```

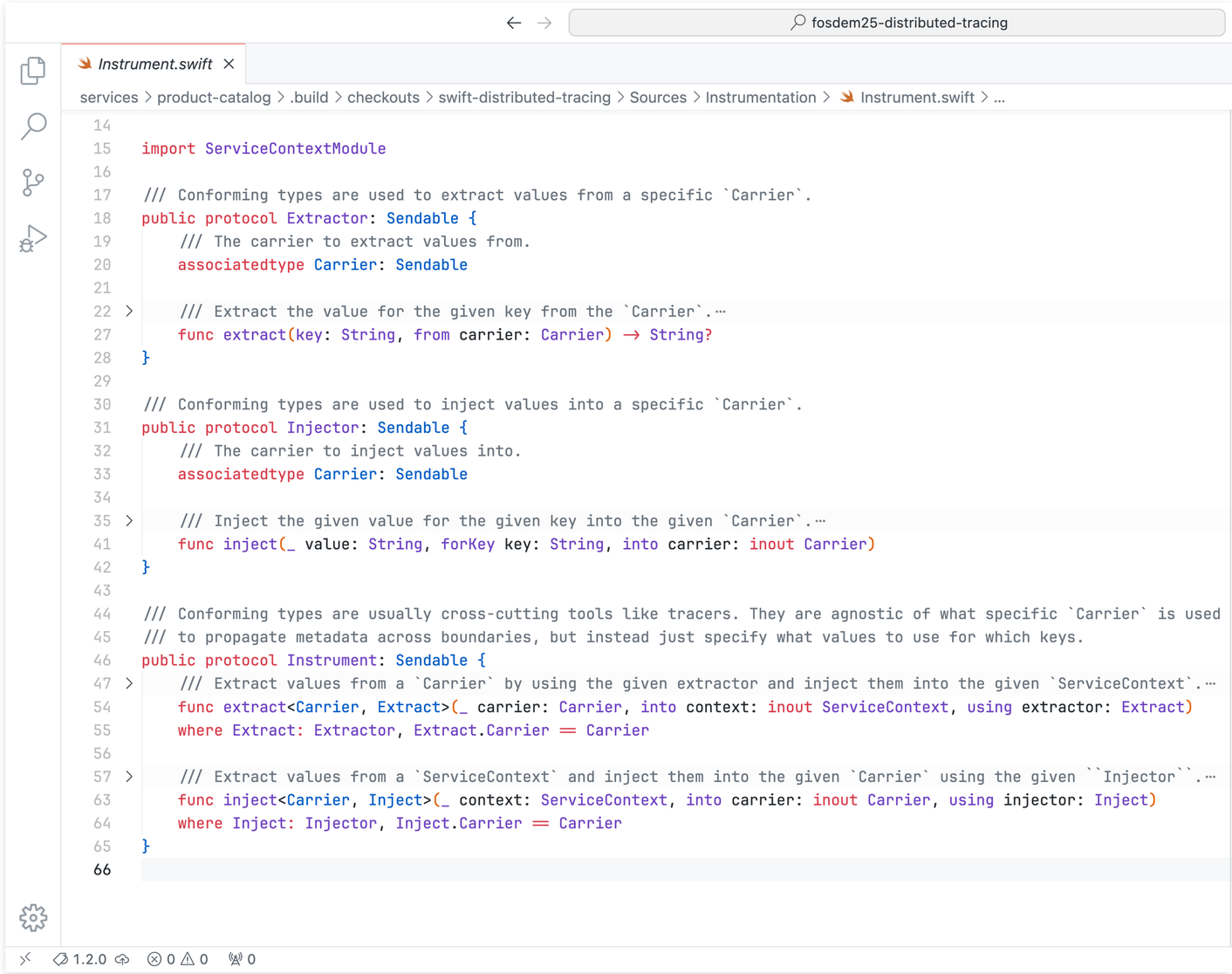
Context Propagation

Distributed Tracing

- Carry tracing identifiers across async/process boundaries
- Example: Client/Server
  - Client: Injects the context into HTTP headers
  - Server: Extracts the context from HTTP headers
  - Server: Creates child span by using the propagated context

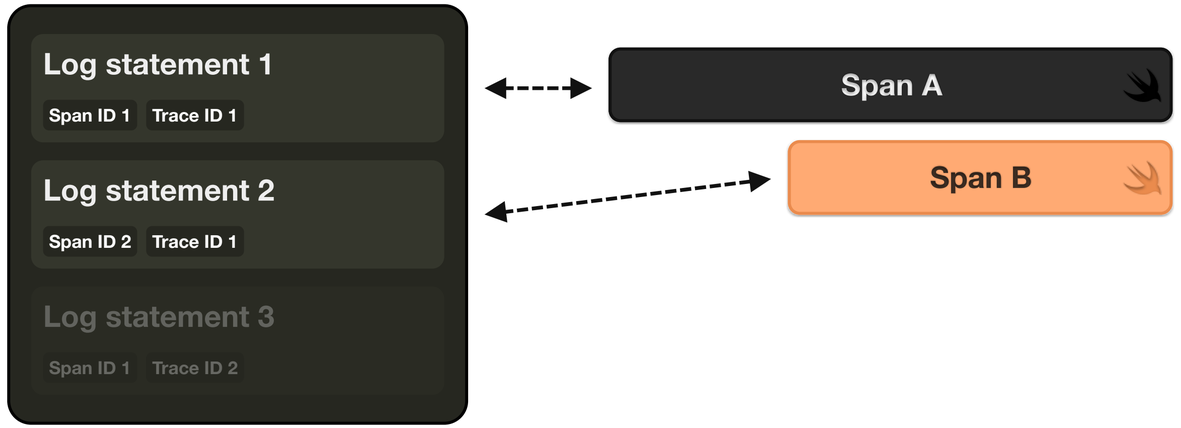
Instrument

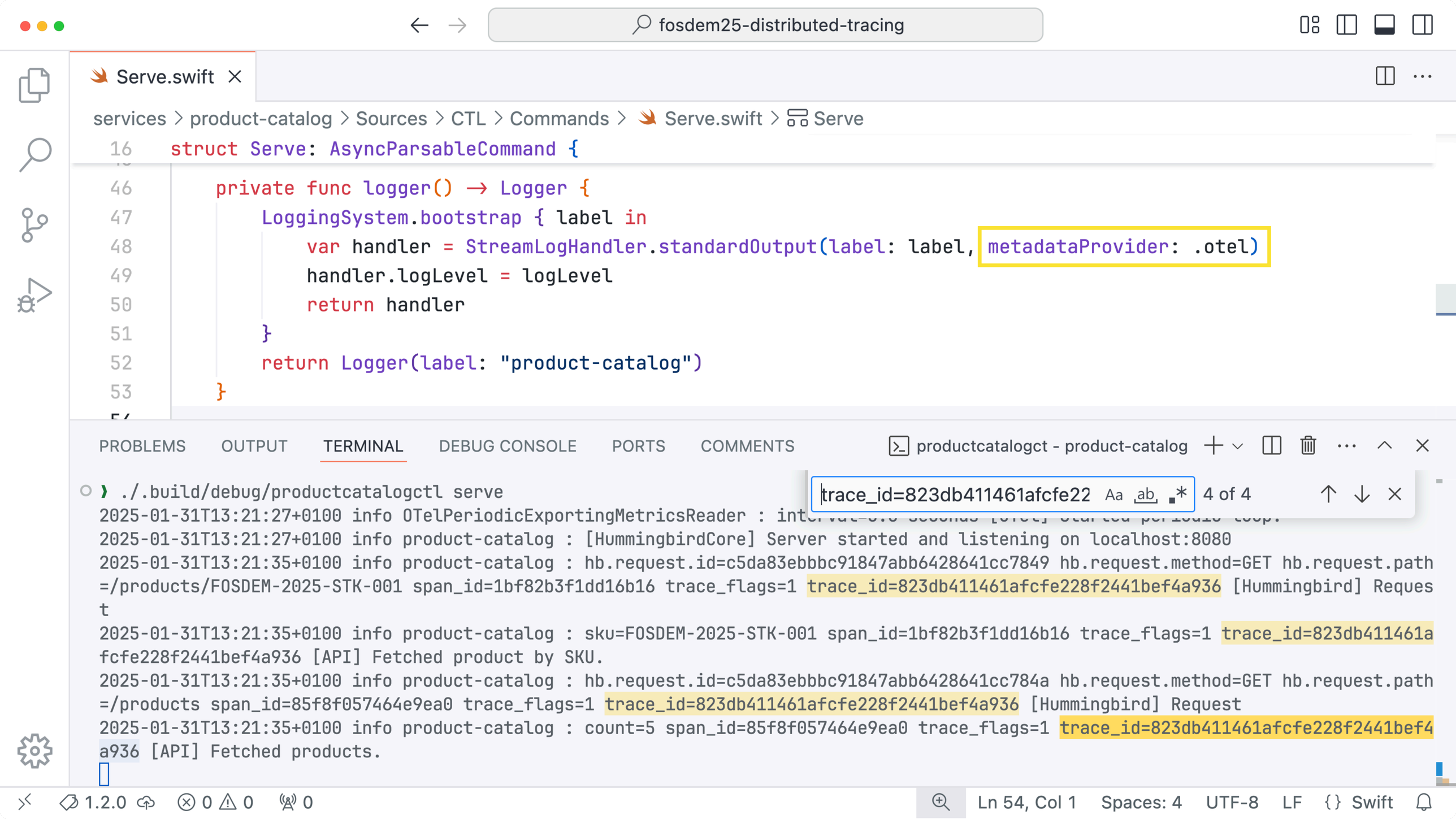
- Agnostic about carrier (e.g. HTTP headers)
- Implementors know about keys/values
- Extended by Tracer protocol

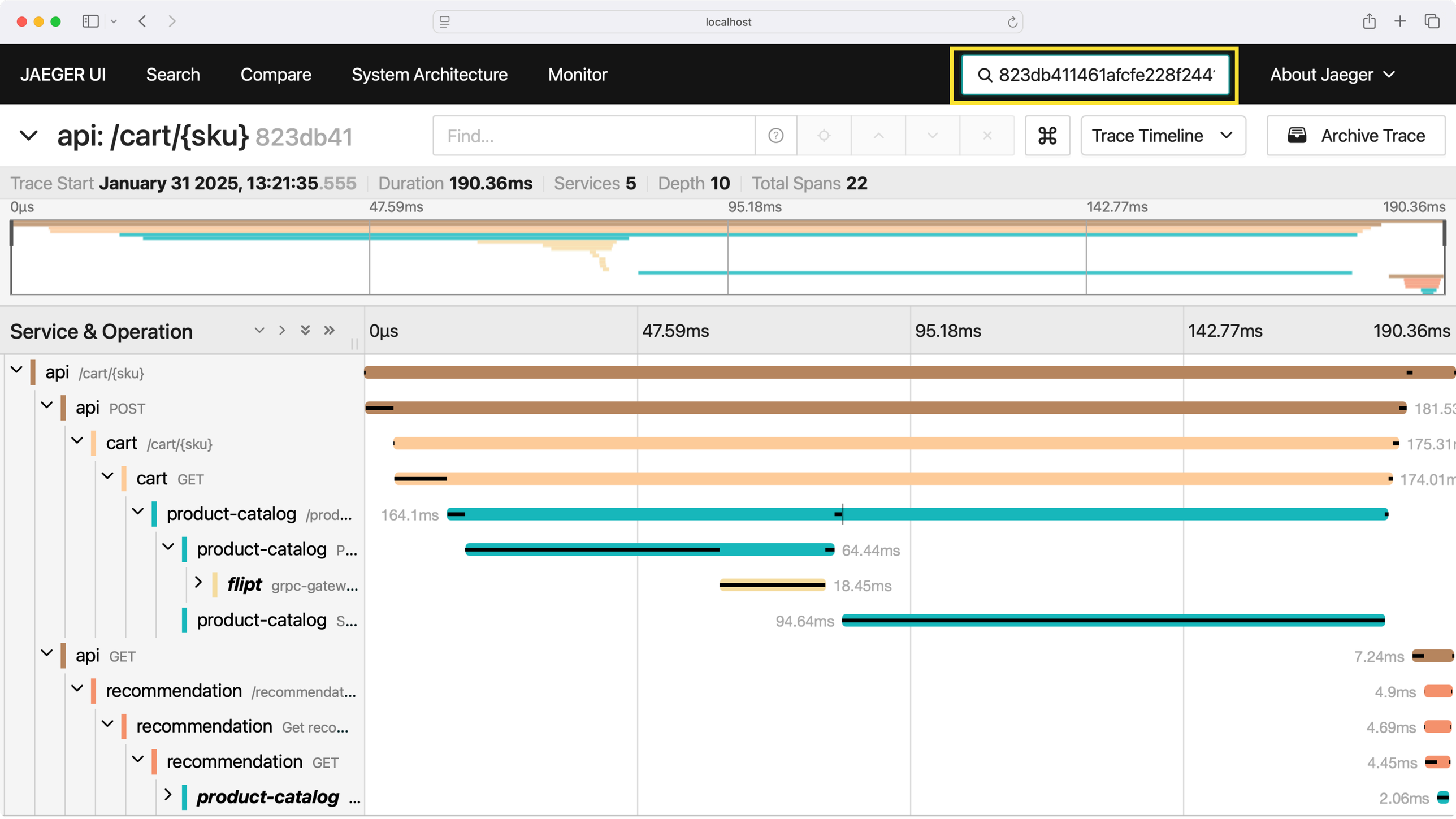


Tracing ❤️ Logging

- Uses swift-log metadata providers
- Transforms task-local ServiceContext into log metadata




fosdem25-distributed-tracing
Serve.swift
services > product-catalog > Sources > CTL > Commands > Serve.swift > Serve
16 struct Serve: AsyncParsableCommand {
46 private func logger() -> Logger {
47 LoggingSystem.bootstrap { label in
48 var handler = StreamLogHandler.standardOutput(label: label, metadataProvider: .otel)
49 handler.logLevel = logLevel
50 return handler
51 }
52 return Logger(label: "product-catalog")
53 }
PROBLEMS
OUTPUT
TERMINAL
DEBUG CONSOLE
PORTS
COMMENTS
productcatalogct - product-catalog
./.build/debug/productcatalogctl serve
trace_id=823db411461afcfe22
4 of 4
2025-01-31T13:21:27+0100 info OTelPeriodicExportingMetricsReader : interval=5.0 seconds [otel] started periodic loop.
2025-01-31T13:21:27+0100 info product-catalog : [HummingbirdCore] Server started and listening on localhost:8080
2025-01-31T13:21:35+0100 info product-catalog : hb.request.id=c5da83ebbbc91847abb6428641cc7849 hb.request.method=GET hb.request.path=/products/FOSDEM-2025-STK-001 span_id=1bf82b3f1dd16b16 trace_flags=1 trace_id=823db411461afcfe228f2441bef4a936 [Hummingbird] Request
2025-01-31T13:21:35+0100 info product-catalog : sku=FOSDEM-2025-STK-001 span_id=1bf82b3f1dd16b16 trace_flags=1 trace_id=823db411461afcfe228f2441bef4a936 [API] Fetched product by SKU.
2025-01-31T13:21:35+0100 info product-catalog : hb.request.id=c5da83ebbbc91847abb6428641cc784a hb.request.method=GET hb.request.path=/products span_id=85f8f057464e9ea0 trace_flags=1 trace_id=823db411461afcfe228f2441bef4a936 [Hummingbird] Request
2025-01-31T13:21:35+0100 info product-catalog : count=5 span_id=85f8f057464e9ea0 trace_flags=1 trace_id=823db411461afcfe228f2441bef4a936 [API] Fetched products.
1.2.0
0 0
0
Ln 54, Col 1
Spaces: 4
UTF-8
LF
Swift


localhost
JAEGER UI
Search
Compare
System Architecture
Monitor
823db411461afcfe228f244
About Jaeger
api: /cart/{sku} 823db41
Find...
Trace Timeline
Archive Trace
Trace Start January 31 2025, 13:21:35.555
Duration 190.36ms
Services 5
Depth 10
Total Spans 22
0µs
47.59ms
95.18ms
142.77ms
190.36ms
Service & Operation
api /cart/{sku}
api POST
cart /cart/{sku}
cart GET
product-catalog /prod...
product-catalog P...
flipt grpc-gatew...
product-catalog S...
api GET
recommendation /recommendat...
recommendation Get reco...
recommendation GET
product-catalog ...
164.1ms
64.44ms
18.45ms
94.64ms
7.24ms
4.9ms
4.69ms
4.45ms
2.06ms

- Open observability standard
- Supports Logging, Metrics, and Distributed Tracing
- OpenTelemetry Protocol (OTLP)
- Supported by various observability tools

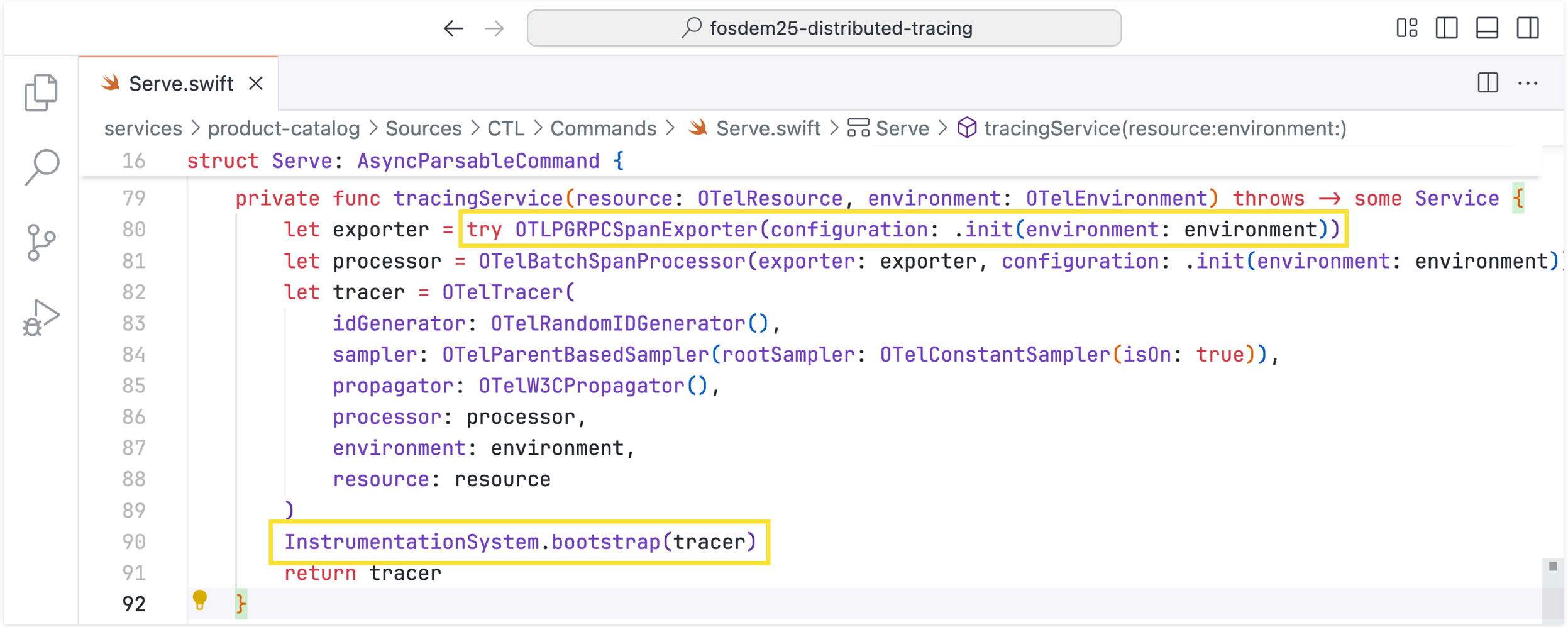
Swift OTel 📦

github.com/swift-otel/swift-otel

- OTLP exporters for Server-Side Swift
- Conforms to Tracer protocol
- Supports Metrics

Swift OTel 📦

Tracer Bootstrap



Next Steps 🚀

- Log exporting in Swift OTel
- Built-in Swift Distributed Tracing in more libraries
  - Database drivers
  - AsyncHTTPClient (swift-server/async-http-client/pull/320)
- Swift OTel 1.0

Links 🔗

Swift Distributed Tracing 📦

Swift OTel 📦