Oslo

Building applications with Serverless Framework and AWS Lambda

Fredrik Vraalsen

Berlin Buzzwords 2019

© Thomas Ekström

Workshop agenda

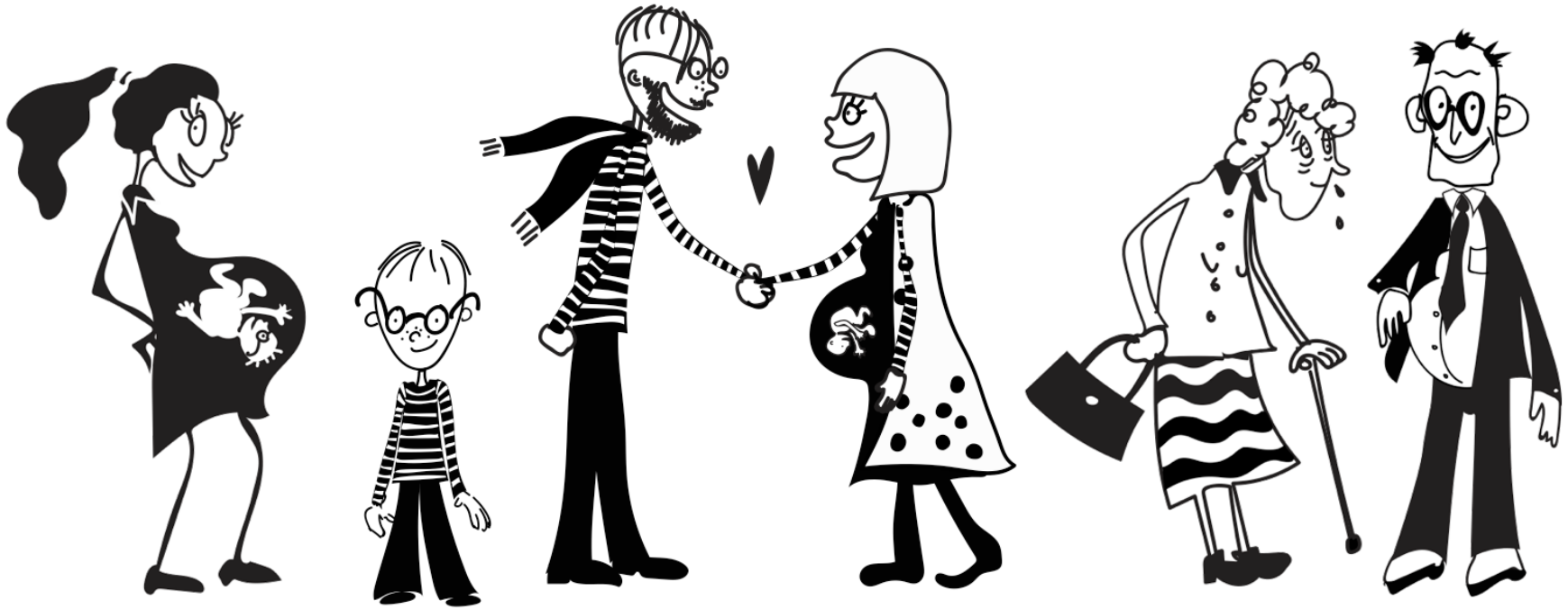
- AWS Lambda
- Serverless Framework
- Configuration and deployment
- Backend API
- Event processing
- Orchestration
- Performance
- Packaging
- Testing

http://bit.ly/bbuzz19-serverless-lambda

Who is Fredrik?

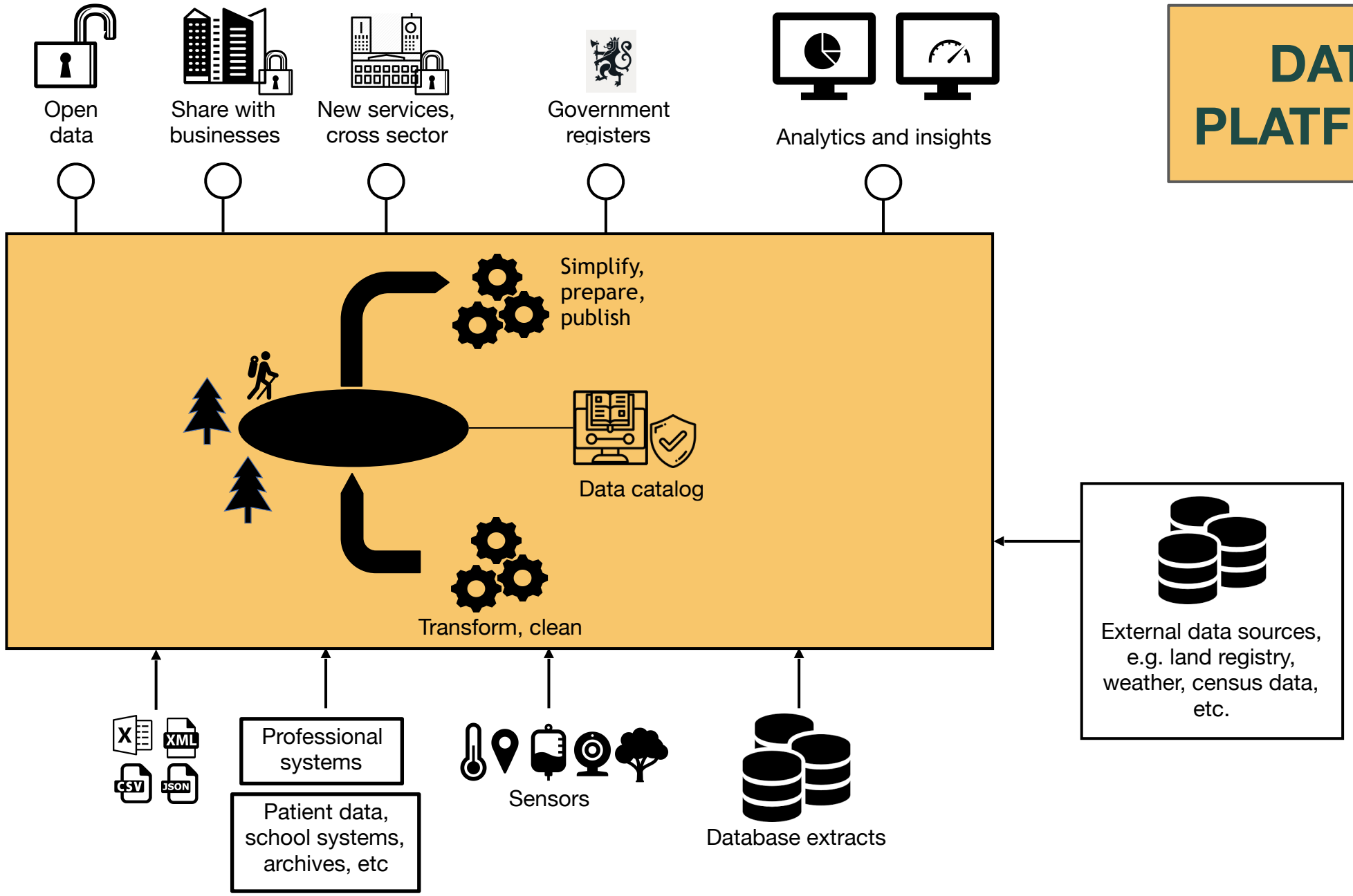
- Oslo, Norway
- Developer for 22+ years
- Data Platform Architect at Origo (City of Oslo)
- Dad, cyclist, gamer, sci-fi fan, photo geek

The Story of Tim

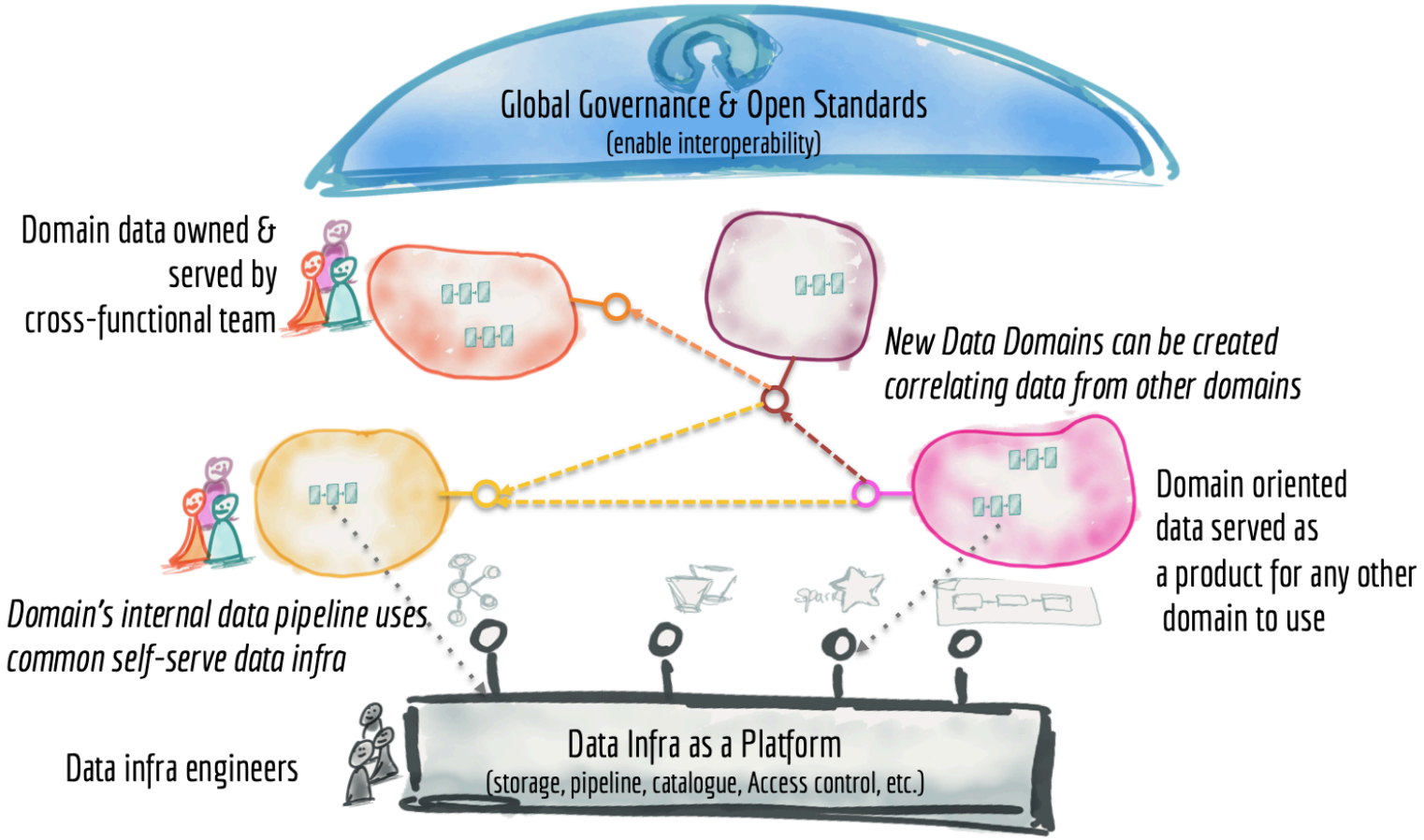


https://www.youtube.com/watch?v=xPciIWM3ztQ

DATA PLATFORM



Distributed data platform?



https://martinfowler.com/articles/data-monolith-to-mesh.html

Serverless / Lambda in our Data Platform

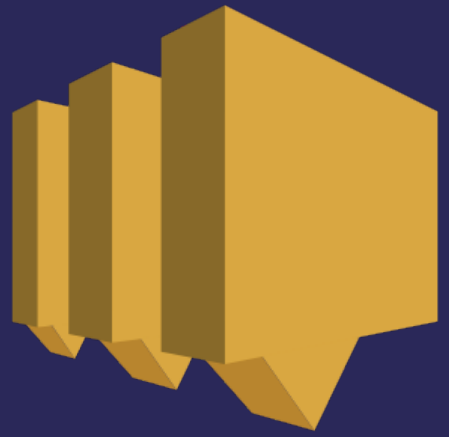
- Microservices - REST APIs
- Processing pipeline components
  - Transformations
  - Validation

Kinesis

Step Functions

Simple Storage Service (S3)

API Gateway



Simple Notification Service (SNS)

Simple Queue Service (SQS)

AWS Lambda

https://aws.amazon.com/lambda/

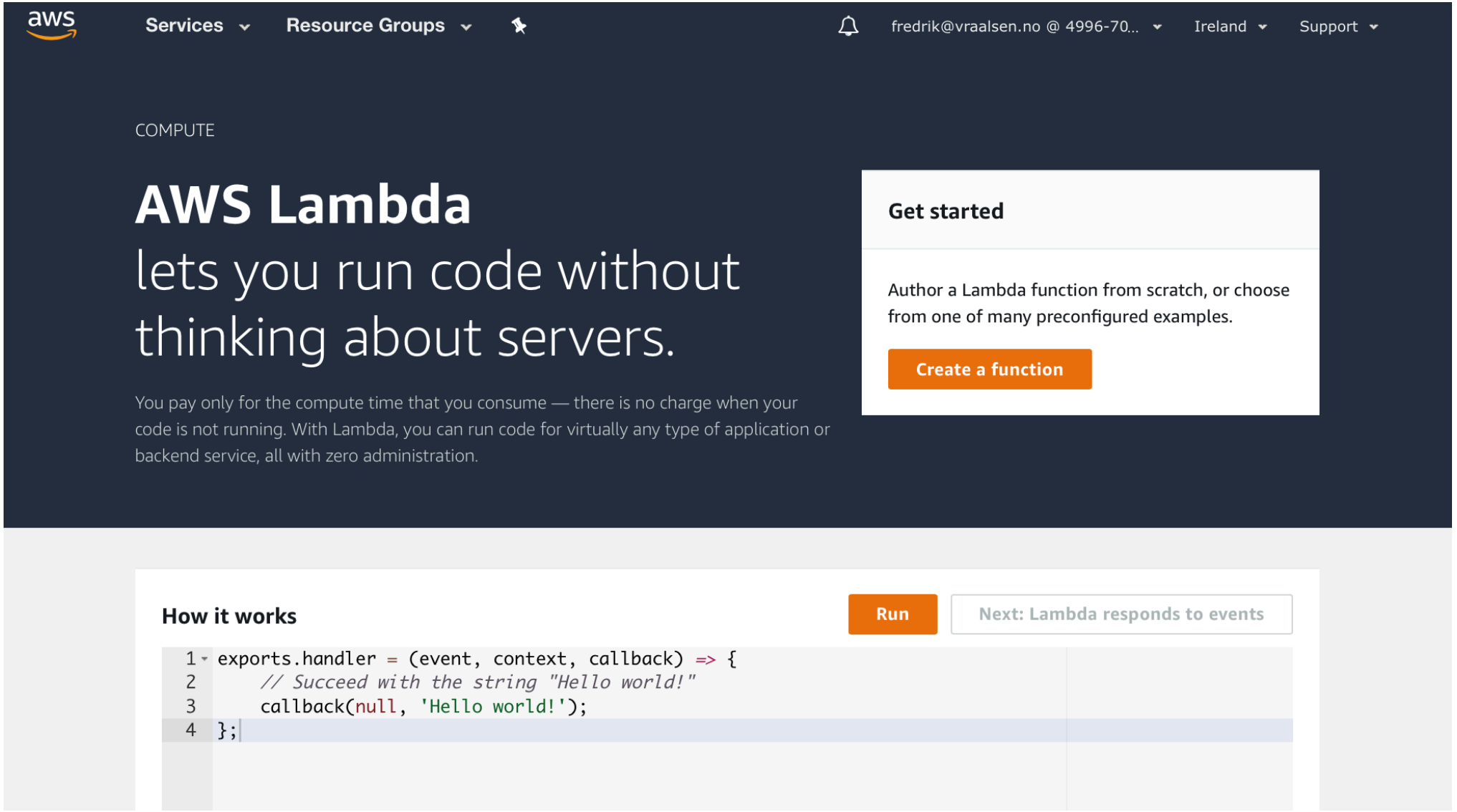
AWS Lambda

- Function-as-a-Service (FaaS)
- Serverless
- Pay for compute time (+ memory)

Hello World

```
def hello(event, context):
    return "Hello, world!"
```

Lambda demo


aws
Services
Resource Groups
fredrik@vraalsen.no @ 4996-70...
Ireland
Support
COMPUTE
AWS Lambda
lets you run code without thinking about servers.
You pay only for the compute time that you consume — there is no charge when your code is not running. With Lambda, you can run code for virtually any type of application or backend service, all with zero administration.
Get started
Author a Lambda function from scratch, or choose from one of many preconfigured examples.
Create a function
How it works
Run
Next: Lambda responds to events
1 exports.handler = (event, context, callback) => {
2 // Succeed with the string "Hello world!"
3 callback(null, 'Hello world!');
4 };

Manual configuration and deployment

Infrastructure as Code

Infrastructure as ~~Code~~ Configuration

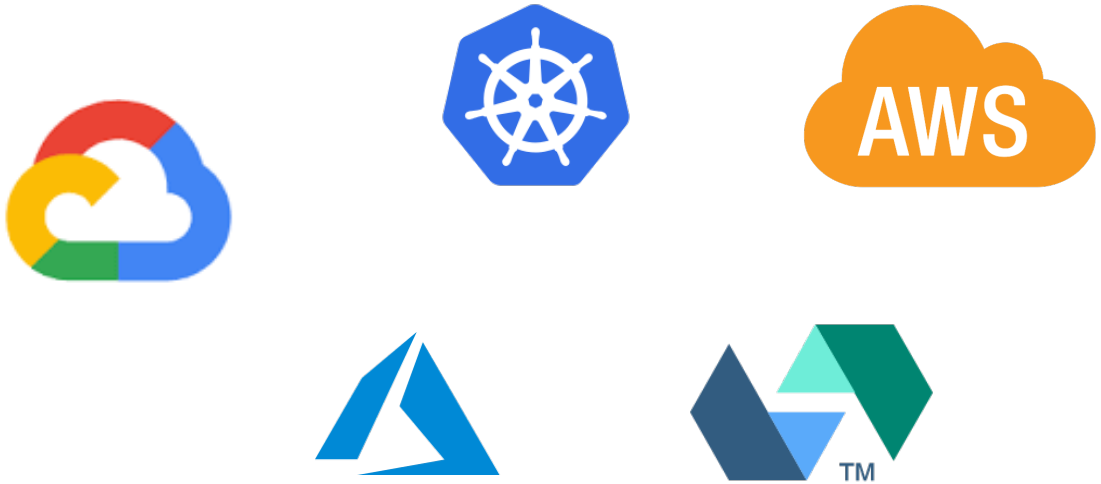
pulumi

serverless

https://serverless.com/framework/

Serverless Framework

- Configure and deploy serverless applications
- Multi-cloud



Configuration and deployment

In git repo: `1_hello`

Templates

```
sls create --template aws-python3 --name hello
```

https://serverless.com/framework/docs/providers/aws/cli-reference/create/

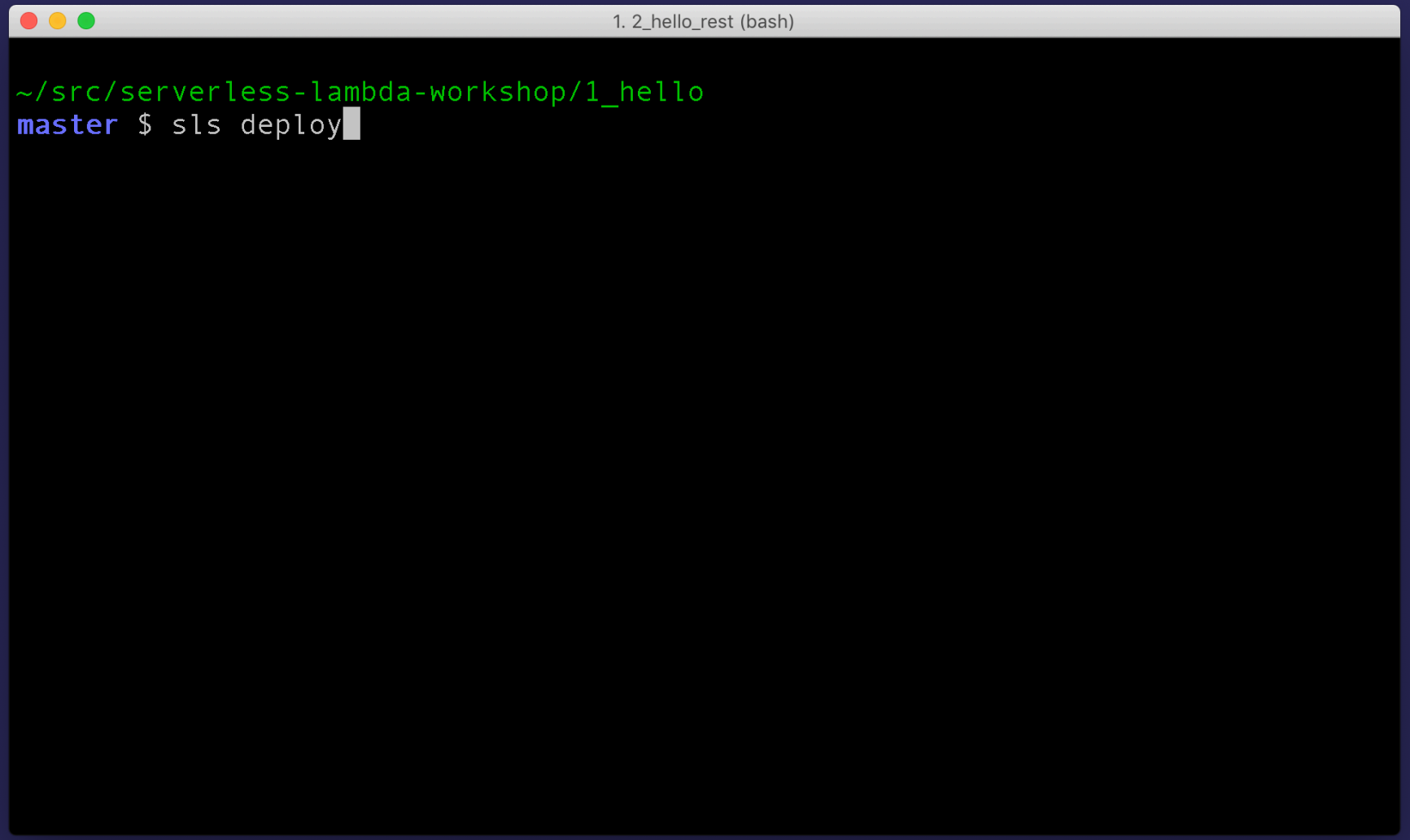
serverless.yml

```
service: hello

provider:
  name: aws
  region: eu-west-1
  runtime: python3.7

functions:
  hello:
    handler: handler.hello
```

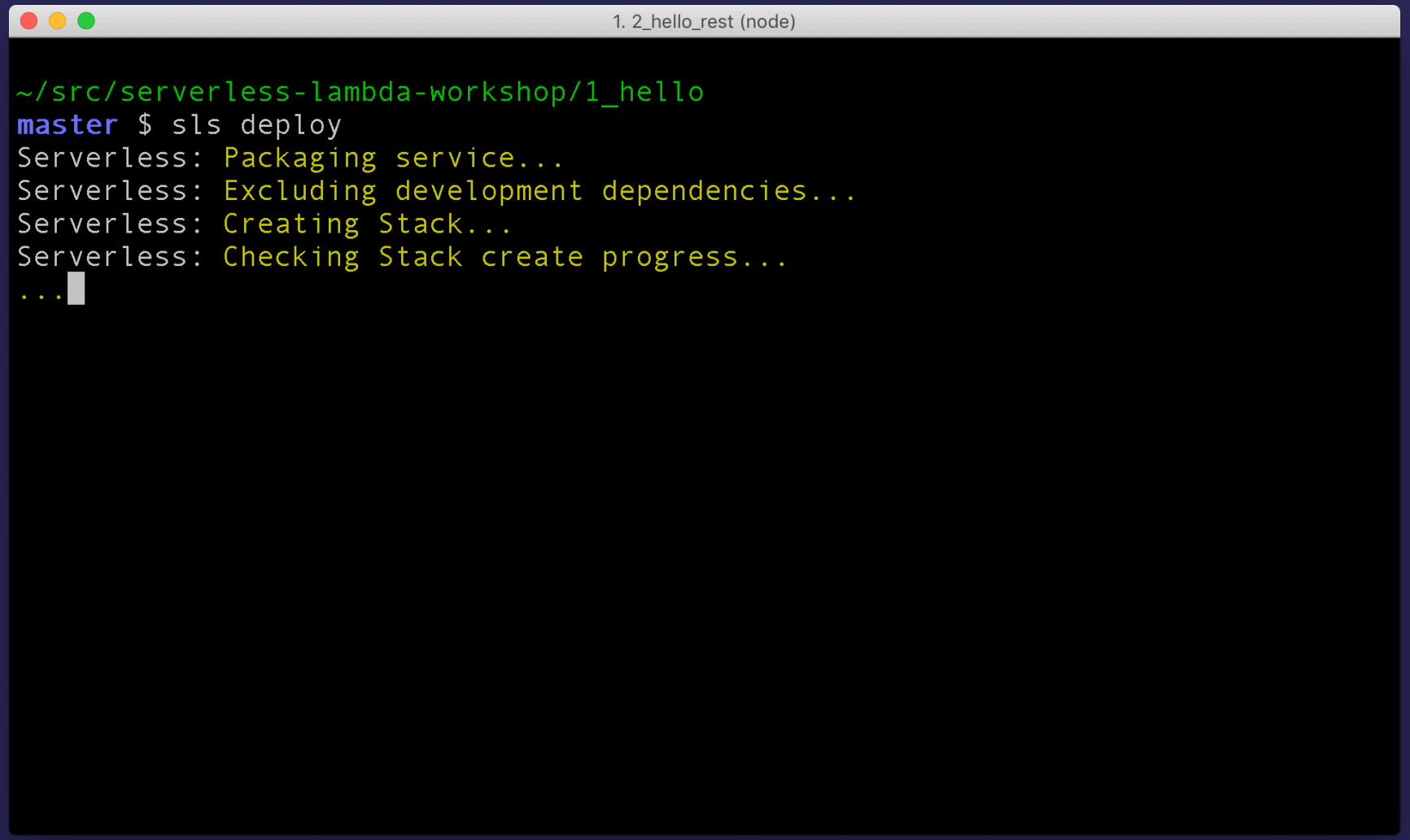
handler.py

```
def hello(event, context):
    return "Hello, world!"
```

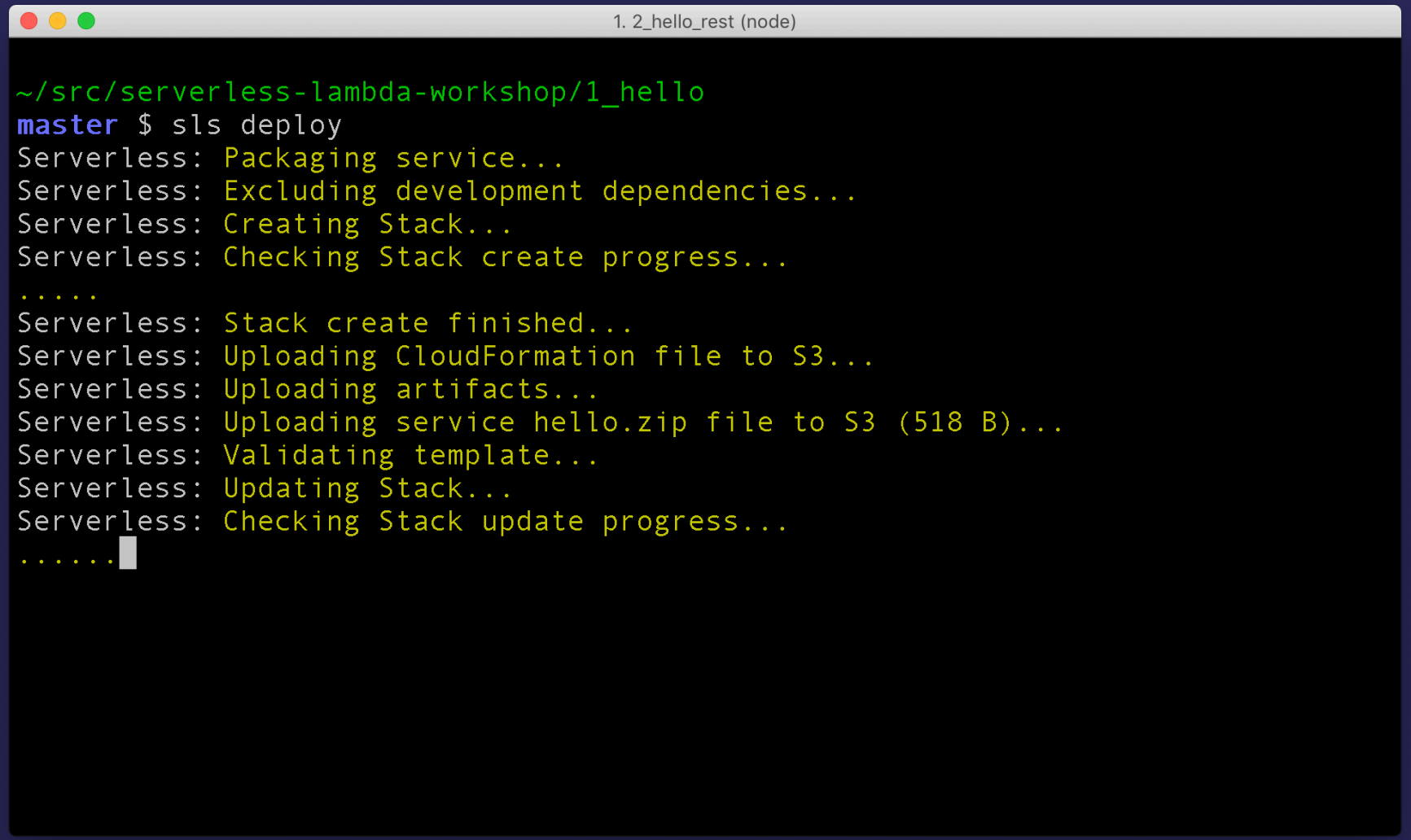
Deployment


1. 2_hello_rest (bash)
~/src/serverless-lambda-workshop/1_hello
master $ sls deploy

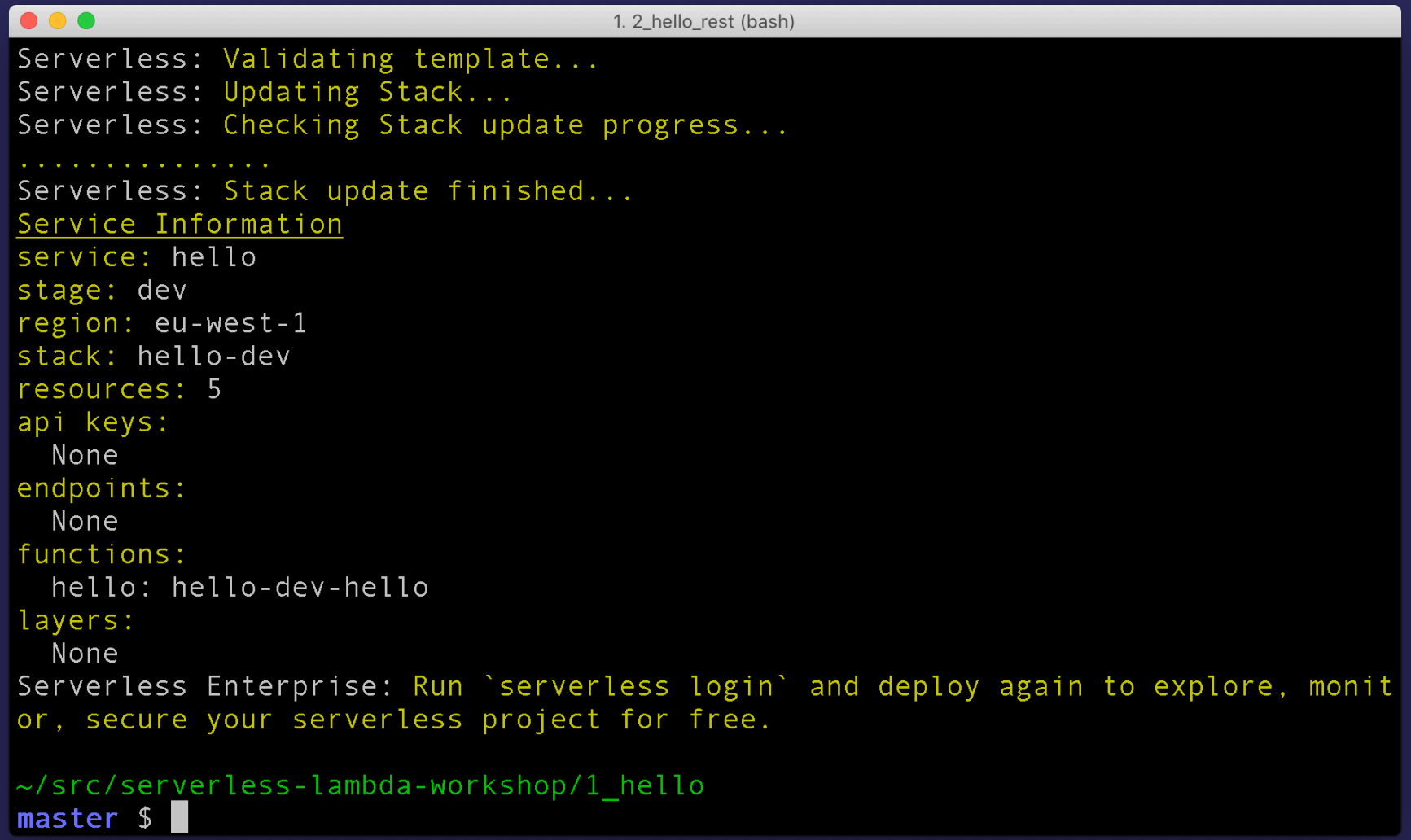
Deployment


1. 2_hello_rest (node)
~/src/serverless-lambda-workshop/1_hello
master $ sls deploy
Serverless: Packaging service...
Serverless: Excluding development dependencies...
Serverless: Creating Stack...
Serverless: Checking Stack create progress...
...

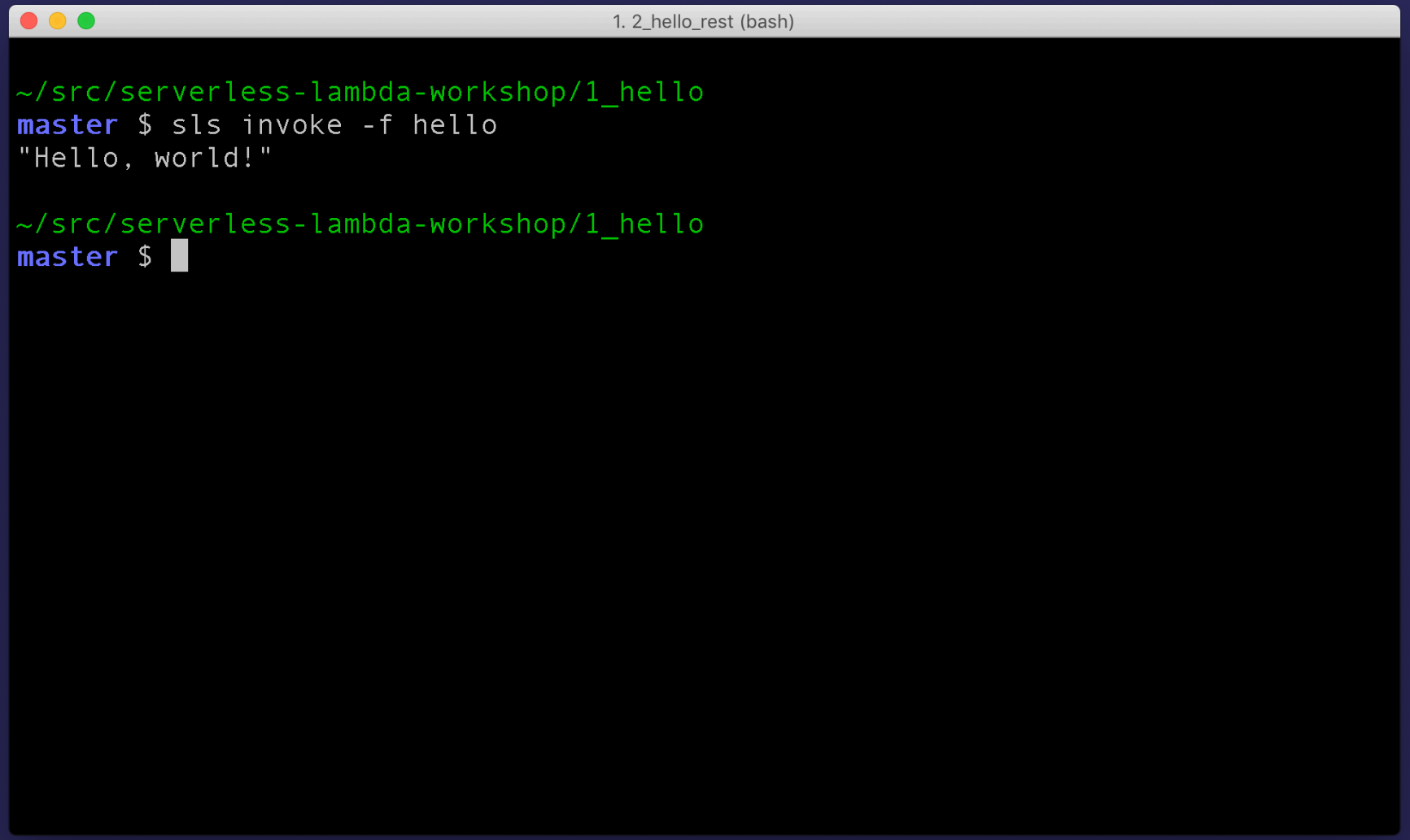
Deployment


1. 2_hello_rest (node)
~/src/serverless-lambda-workshop/1_hello
master $ sls deploy
Serverless: Packaging service...
Serverless: Excluding development dependencies...
Serverless: Creating Stack...
Serverless: Checking Stack create progress...
.....
Serverless: Stack create finished...
Serverless: Uploading CloudFormation file to S3...
Serverless: Uploading artifacts...
Serverless: Uploading service hello.zip file to S3 (518 B)...
Serverless: Validating template...
Serverless: Updating Stack...
Serverless: Checking Stack update progress...
......

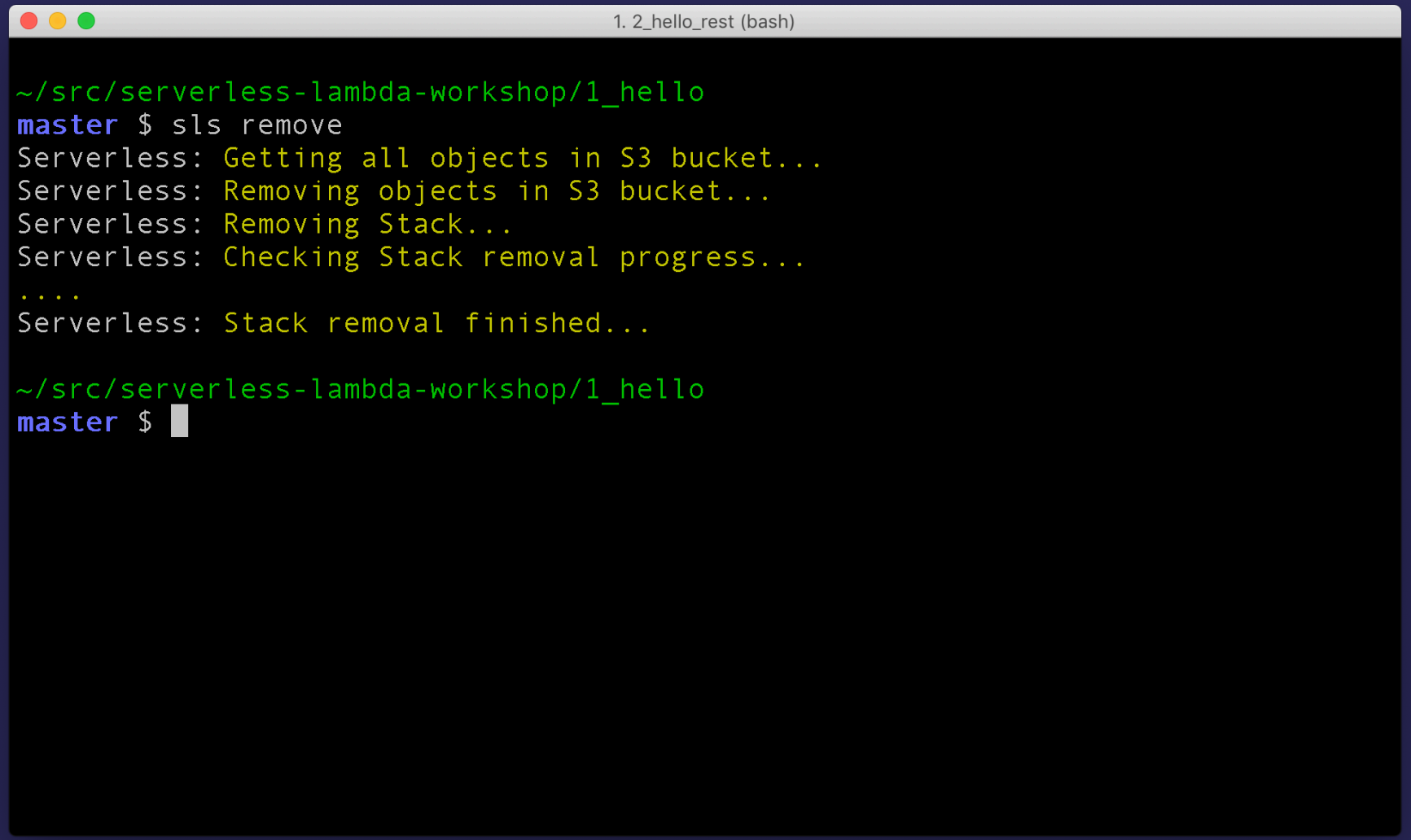
Deployment


1. 2_hello_rest (bash)
Serverless: Validating template...
Serverless: Updating Stack...
Serverless: Checking Stack update progress...
....
Serverless: Stack update finished...
Service Information
service: hello
stage: dev
region: eu-west-1
stack: hello-dev
resources: 5
api keys:
None
endpoints:
None
functions:
hello: hello-dev-hello
layers:
None
Serverless Enterprise: Run `serverless login` and deploy again to explore, monitor, secure your serverless project for free.
~/src/serverless-lambda-workshop/1_hello
master $

Invoking function



Undeploy


1. 2_hello_rest (bash)
~/src/serverless-lambda-workshop/1_hello
master $ sls remove
Serverless: Getting all objects in S3 bucket...
Serverless: Removing objects in S3 bucket...
Serverless: Removing Stack...
Serverless: Checking Stack removal progress...
....
Serverless: Stack removal finished...
~/src/serverless-lambda-workshop/1_hello
master $

Lambda use cases

- Backend APIs
- Web applications
- Event processing
- File processing
- ETL

Lambda use cases

- **Backend APIs**
- Web applications
- **Event processing**
- **File processing**
- ETL

Triggering Lambda functions

- API Gateway
- Application load balancer
- SNS topic
- SQS queue
- Kinesis
- S3
- DynamoDB
- Websocket
- IoT
- Alexa
- CloudWatch
- Schedule

https://serverless.com/framework/docs/providers/aws/events/

Triggering Lambda functions

- **API Gateway**
- Application load balancer
- **SNS topic**
- SQS queue
- Kinesis
- S3
- DynamoDB
- Websocket
- IoT
- Alexa
- CloudWatch
- Schedule

https://serverless.com/framework/docs/providers/aws/events/

Backend API

In git repo: `2_hello_http`

https://serverless.com/framework/docs/providers/aws/events/apigateway/

Hello HTTP!

```
functions:
  hello:
    handler: handler.hello
    events:
      - http:
          path: hello/{name}
          method: get
```

Hello HTTP!

```
def hello(event, context):
    name = event["pathParameters"]["name"]
```

Hello HTTP!

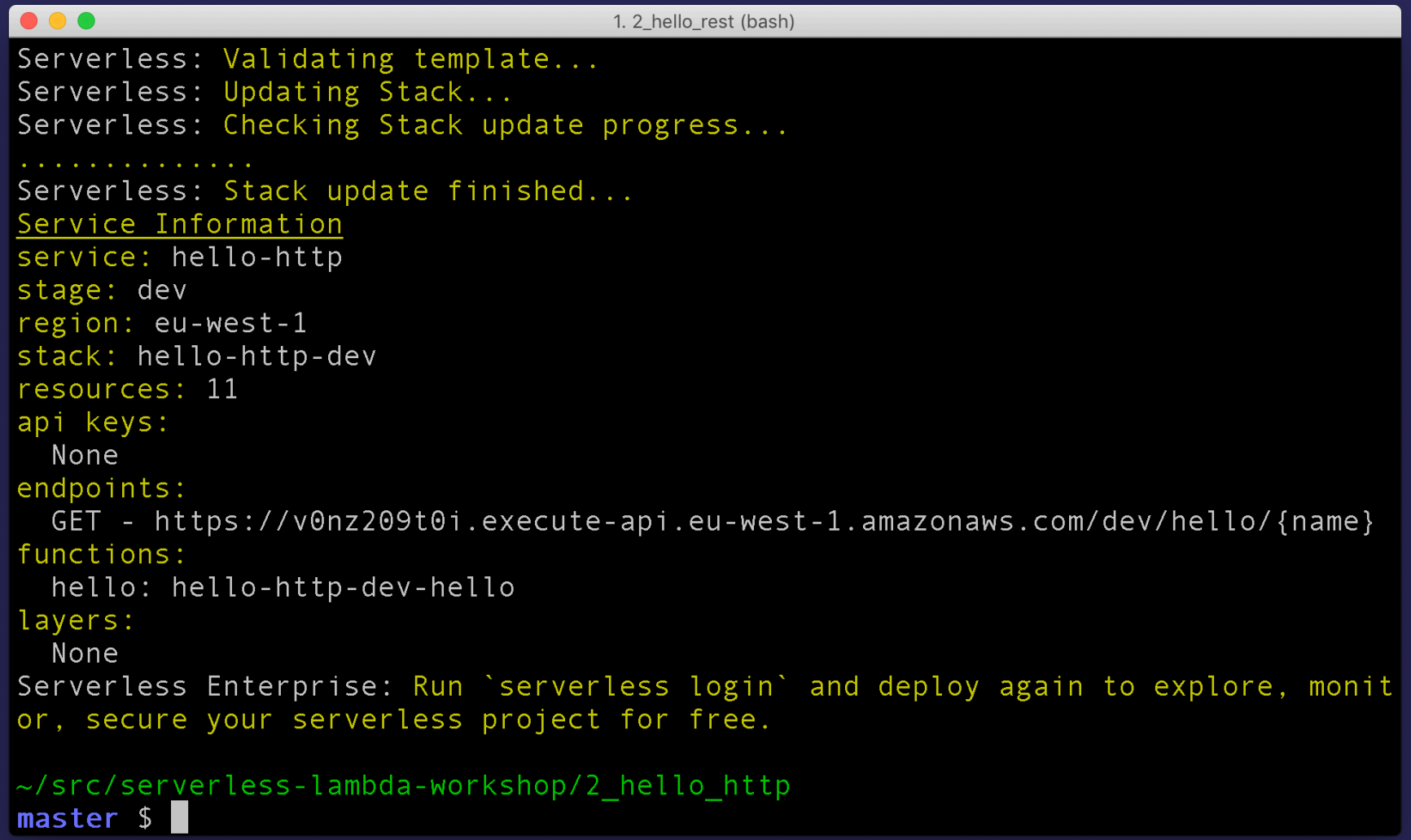
```
def hello(event, context):
    name = event["pathParameters"]["name"]

    response = { "message": f"Hello, {name}!" }
```

Hello HTTP!

```
import json

def hello(event, context):
    name = event["pathParameters"]["name"]

    response = { "message": f"Hello, {name}!" }

    return {
        "statusCode": 200,
        "body": json.dumps(response)
    }
```

https://docs.aws.amazon.com/apigateway/latest/developerguide/set-up-lambda-proxy-integrations.html#api-gateway-simple-proxy-for-lambda-input-format

Deploy

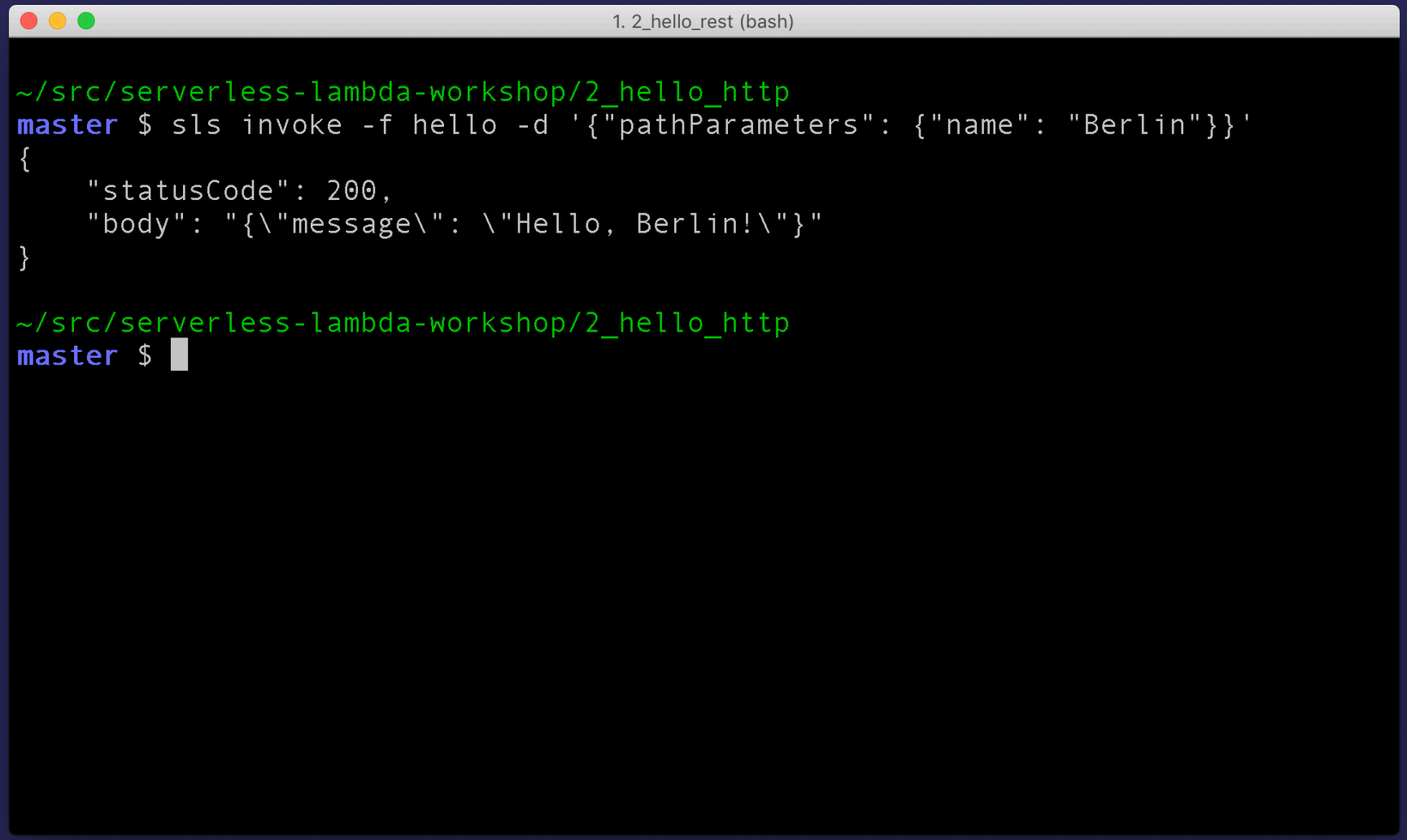

1. 2_hello_rest (bash)
Serverless: Validating template...
Serverless: Updating Stack...
Serverless: Checking Stack update progress...
.....
Serverless: Stack update finished...
Service Information
service: hello-http
stage: dev
region: eu-west-1
stack: hello-http-dev
resources: 11
api keys:
None
endpoints:
GET - https://v0nz209t0i.execute-api.eu-west-1.amazonaws.com/dev/hello/{name}
functions:
hello: hello-http-dev-hello
layers:
None
Serverless Enterprise: Run `serverless login` and deploy again to explore, monitor, secure your serverless project for free.
~/src/serverless-lambda-workshop/2_hello_http
master $

Invoking function

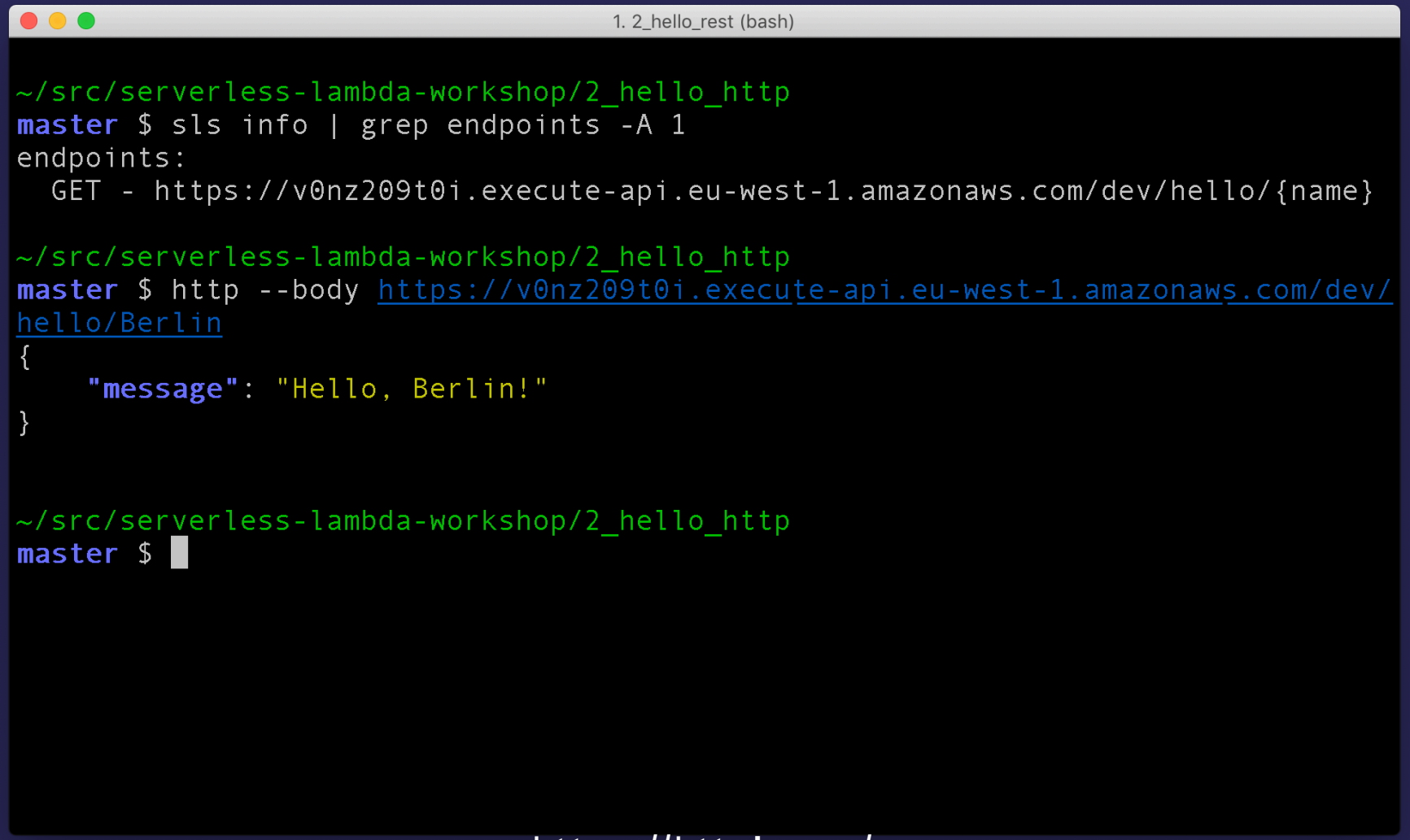
1. 2_hello_rest (bash)

```
~/src/serverless-lambda-workshop/2_hello_http
master $ sls invoke -f hello
{
    "errorMessage": "'pathParameters'",
    "errorType": "KeyError",
    "stackTrace": [
        "  File \"/var/task/handler.py\", line 10, in hello\n    name = unquote(
event[\"pathParameters\"][\"name\"])\n"
    ]
}

  Error --------------------------------------------------

  Invoked function failed

    For debugging logs, run again after setting the "SLS_DEBUG=*" environment v
ariable.

  Get Support --------------------------------------------
    Docs:          docs.serverless.com
    Bugs:          github.com/serverless/serverless/issues
    Issues:        forum.serverless.com
```

Invoking function



Invoking function



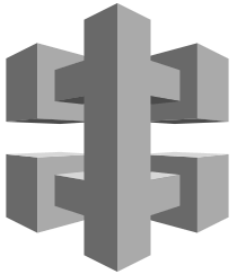
https://httpie.org/

Documentation

Export as... Swagger OpenAPI 3

Export as Swagger

Export as Swagger + API Gateway



Export as Swagger + Postman Extensions

JSON YAML


```
{
  "swagger": "2.0",
  "info": {
    "version": "2019-06-18T07:37:10Z",
    "title": "dev-hello-http"
  },
  "host": "v0nz209t0i.execute-api.eu-west-1.amazonaws.com",
  "basePath": "/dev",
  "schemes": [
    "https"
  ],
  "paths": {
    "/hello/{name}": {
      "get": {
        "responses": {}
      }
    }
  }
```

Documentation

- Plugin: `serverless-aws-documentation`
- Request content, parameters, responses, headers
- Swagger / OpenAPI

- In git repo: 3_hello_documentation

Documentation

```yaml
functions:
  hello:
    handler: handler.hello
    events:
      - http:
          path: hello/{name}
          method: get
          documentation:
            description: "Generate a personalized greeting"
            pathParams:
              -
                name: "name"
                description: "Name of person to be greeted"
                required: true
            methodResponses:
              -
                statusCode: "200"
                responseModels:
                  application/json: "HelloResponse"
```

Documentation

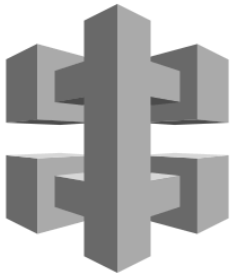
```yaml
custom:
  documentation:
    models:
      -
        name: HelloResponse
        description: "Greeting response"
        contentType: "application/json"
        schema: ${file(models/hello-response.yml)}

plugins:
  - serverless-aws-documentation
```

Documentation

Export as... Swagger OpenAPI 3

Export as Swagger

Export as Swagger + API Gateway

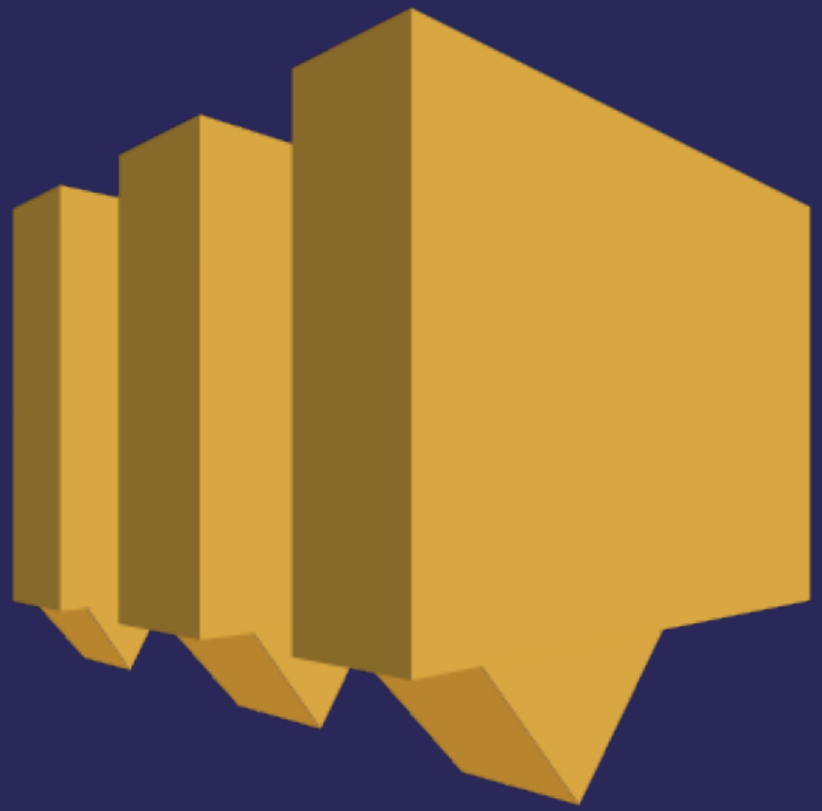


Export as Swagger + Postman Extensions

JSON YAML

```
12  "paths": {
13    "/hello/{name}": {
14      "get": {
15        "description": "Generate a personalized greeting",
16        "produces": [
17          "application/json"
18        ],
19        "parameters": [
20          {
21            "name": "name",
22            "in": "path",
23            "description": "Name of person to be greeted",
24            "required": true,
25            "type": "string"
26          },
27          {
28            "name": "title",
29            "in": "query",
```

Event processing



In git repo: 4_event_processing

https://serverless.com/framework/docs/providers/aws/events/sns/

serverless.yml

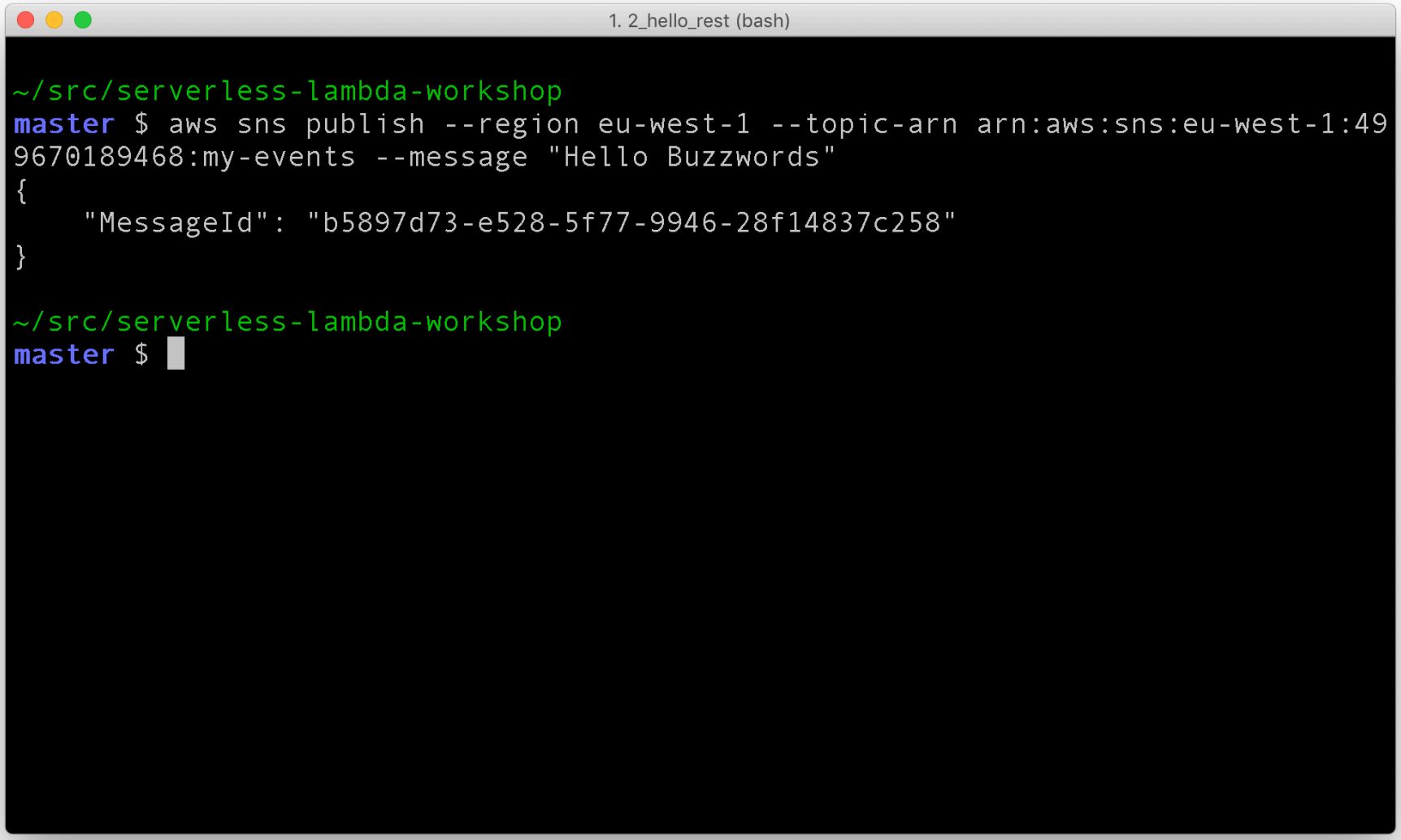
```
functions:
  handle_event:
    handler: handler.handle_event
    events:
      - sns: my-events
```

handler.py

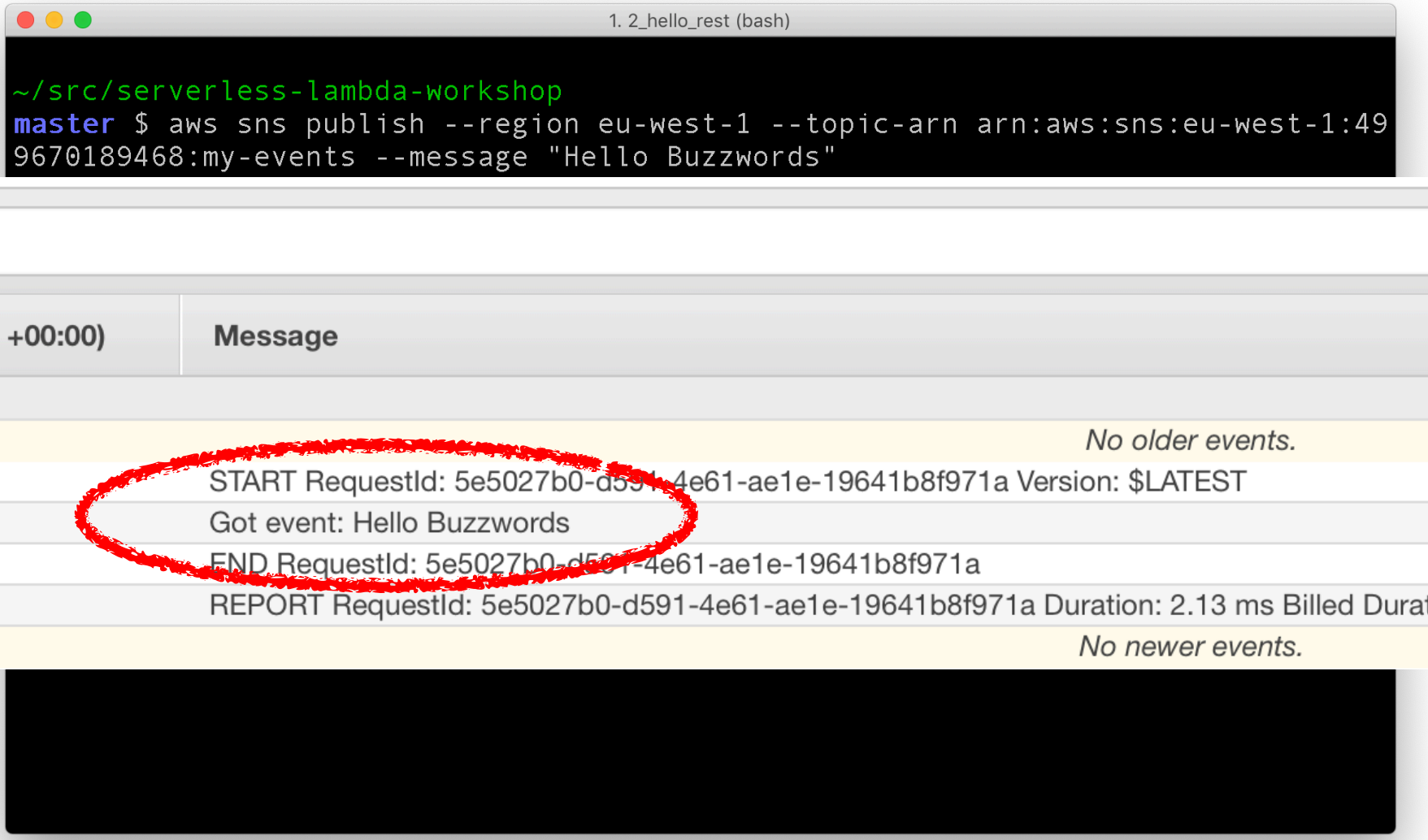
```
def handle_event(event, context):

    for record in event["Records"]:
        msg = record["Sns"]["Message"]

        print(f"Got event: {msg}")
```

https://docs.aws.amazon.com/lambda/latest/dg/with-sns.html

Publish event



Publish event



Orchestration

Orchestration

In git repo: 5_orchestration

https://serverless.com/plugins/serverless-step-functions/

serverless.yml

```
functions:
  validateTemperature:
    handler: handler.validate_temperature
  enrichLocation:
    handler: handler.enrich_location

stepFunctions:
  stateMachines:
    ProcessTemperatures:
      name: ProcessTemperatures
      events:
        - http:
            path: temperatures
            method: post
```

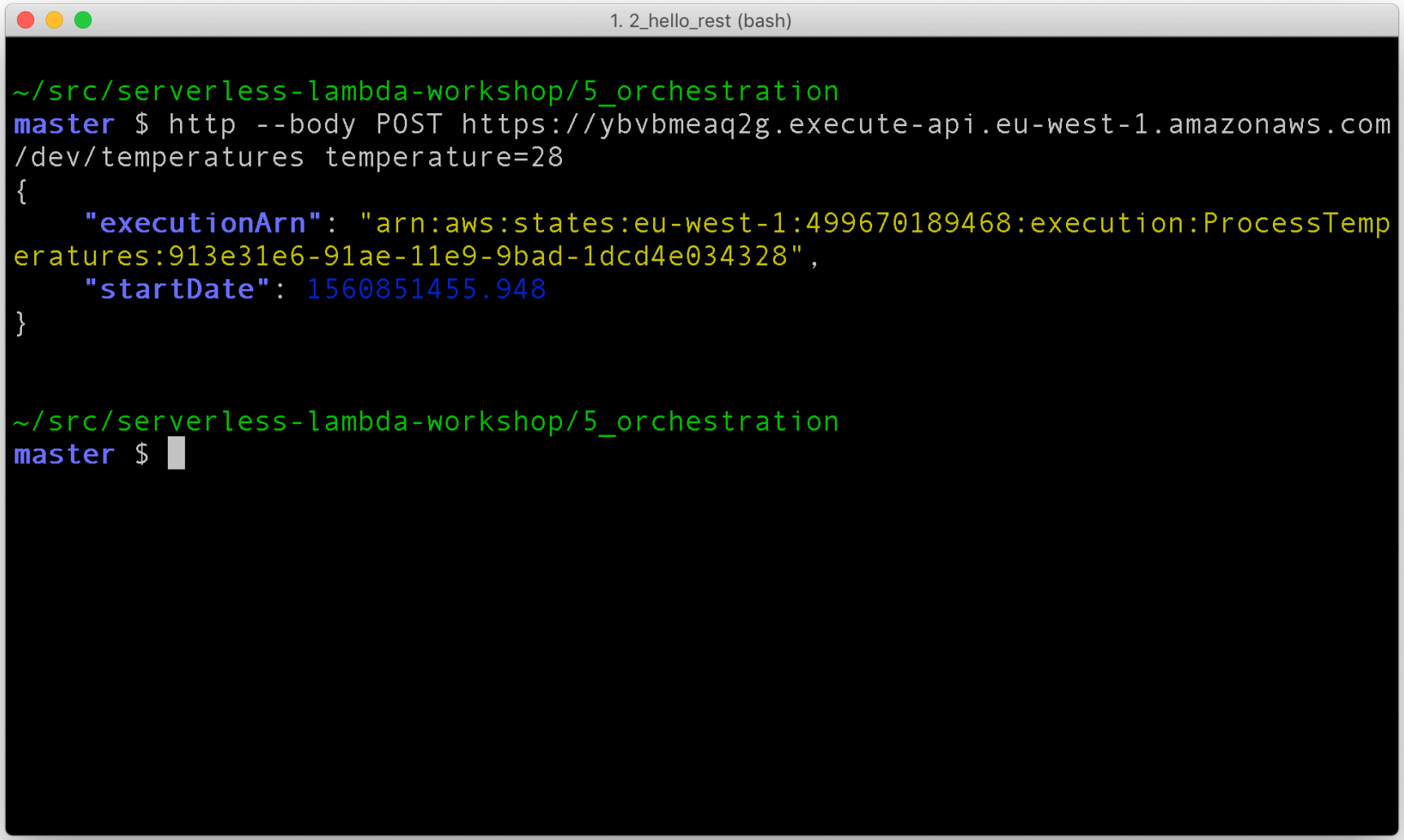
serverless.yml

```
definition:
  StartAt: ValidateTemperature
  States:
    ValidateTemperature:
      Type: Task
      Resource:
        Fn::GetAtt: [ValidateTemperatureLambdaFunction, Arn]
      Next: EnrichLocation
    EnrichLocation:
      Type: Task
      Resource:
        Fn::GetAtt: [EnrichLocationLambdaFunction, Arn]
      End: True
```

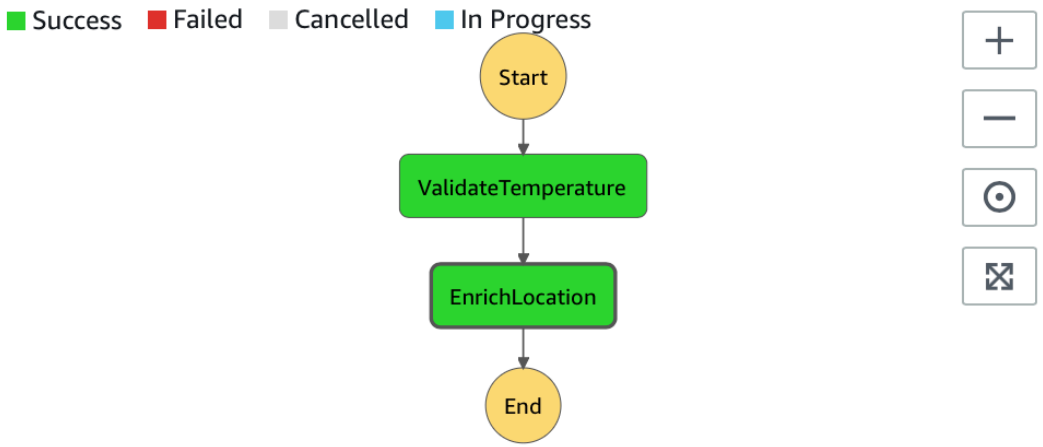
handler.py

```python
def validate_temperature(event, context):
    temperature = int(event["temperature"])
    if temperature < -20:
        raise ValueError("Too cold!")
    if temperature > 30:
        raise ValueError("Too warm!")

    return event

def enrich_location(event, context):
    event["location"] = "Berlin"
    return event
```

Running


1. 2_hello_rest (bash)
~/src/serverless-lambda-workshop/5_orchestration
master $ http --body POST https://ybvbmeaq2g.execute-api.eu-west-1.amazonaws.com/dev/temperatures temperature=28
{
"executionArn": "arn:aws:states:eu-west-1:499670189468:execution:ProcessTemperatures:913e31e6-91ae-11e9-9bad-1dcd4e034328",
"startDate": 1560851455.948
}
~/src/serverless-lambda-workshop/5_orchestration
master $

Visual workflow | Code



Step details (EnrichLocation)

Status

Succeeded

Resource

arn:aws:lambda:eu-west-1:499670189468:function:orchestration-demo-dev-enrichLocation | CloudWatch logs

▼ Input

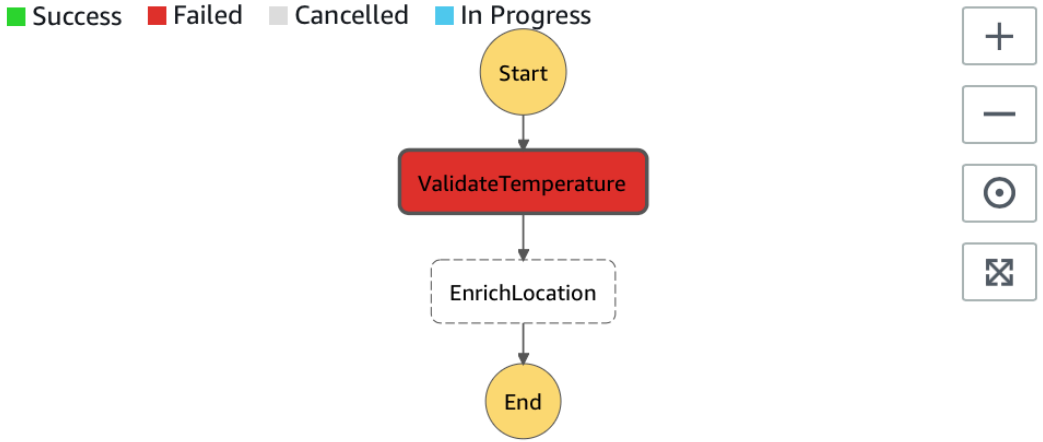
```
{
  "temperature": "28"
}
```

▼ Output

```
{
  "temperature": "28",
  "location": "Berlin"
}
```

▶ Exception

Visual workflow | Code



Step details (ValidateTemperature)

Status

Failed

Resource

arn:aws:lambda:eu-west-1:499670189468:function:orchestration-demo-dev-validateTemperature | CloudWatch logs

▼ Input

```
{
  "temperature": "-29"
}
```

▶ Output

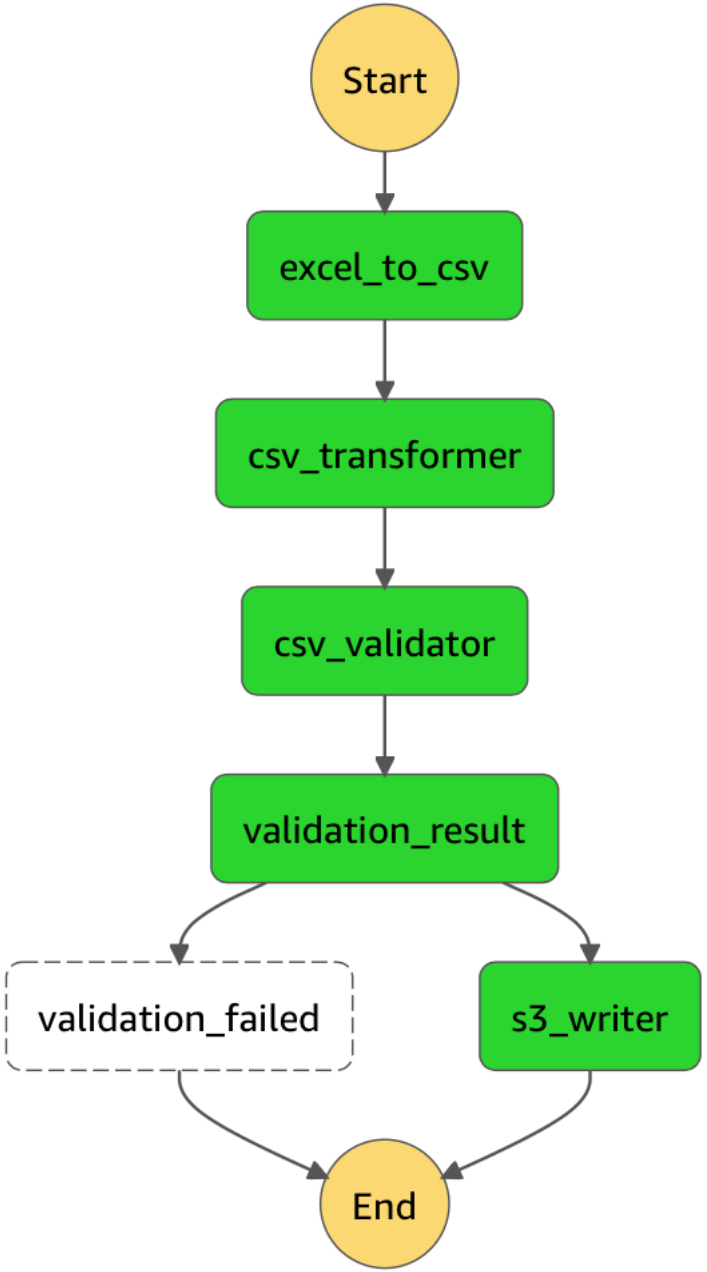
▼ Exception

Error

```
ValueError
```

Cause

```
{
  "errorMessage": "Too cold!",
  "errorType": "ValueError",
```


Start
excel_to_csv
csv_transformer
csv_validator
validation_result
validation_failed
s3_writer
End

Performance

Cold start

Lambda lifecycle

https://blog.travelex.io/from-containers-to-aws-lambda-23f712f9e925

Lambda lifecycle

- 1 container = 1 execution
- Spin up more on demand
- Reuse: Freeze & Thaw
- VPCs require ENIs ==> Slow cold start

Packaging

- Deploy zip file
- Bundle dependencies
- Layers
- Linux!

https://serverless.com/plugins/serverless-python-requirements/

Testing

- separate business logic ==> unit tests, mocking
- Python: moto library
- `sls invoke local -f` *function-name*
- kinesalite, local dynamodb (docker)

https://serverless.com/framework/docs/providers/aws/guide/testing/

More things

- Infrastructure
- Access control (IAM)
- API Gateway integrations
  - Lambda Proxy vs Lambda integration
- Validation
- Versioning

Wrap up

- AWS Lambda
  - FaaS, simple, pay-as-you-go, scalable
- Serverless Framework
  - Configuration, deployment, testing, plugins
  - Easy = simpler, Hard = ?
- We're still evaluating

Questions? Feedback?

fredrik@vraalsen.no

@fredriv

Thanks for listening!

fredrik@vraalsen.no

@fredriv