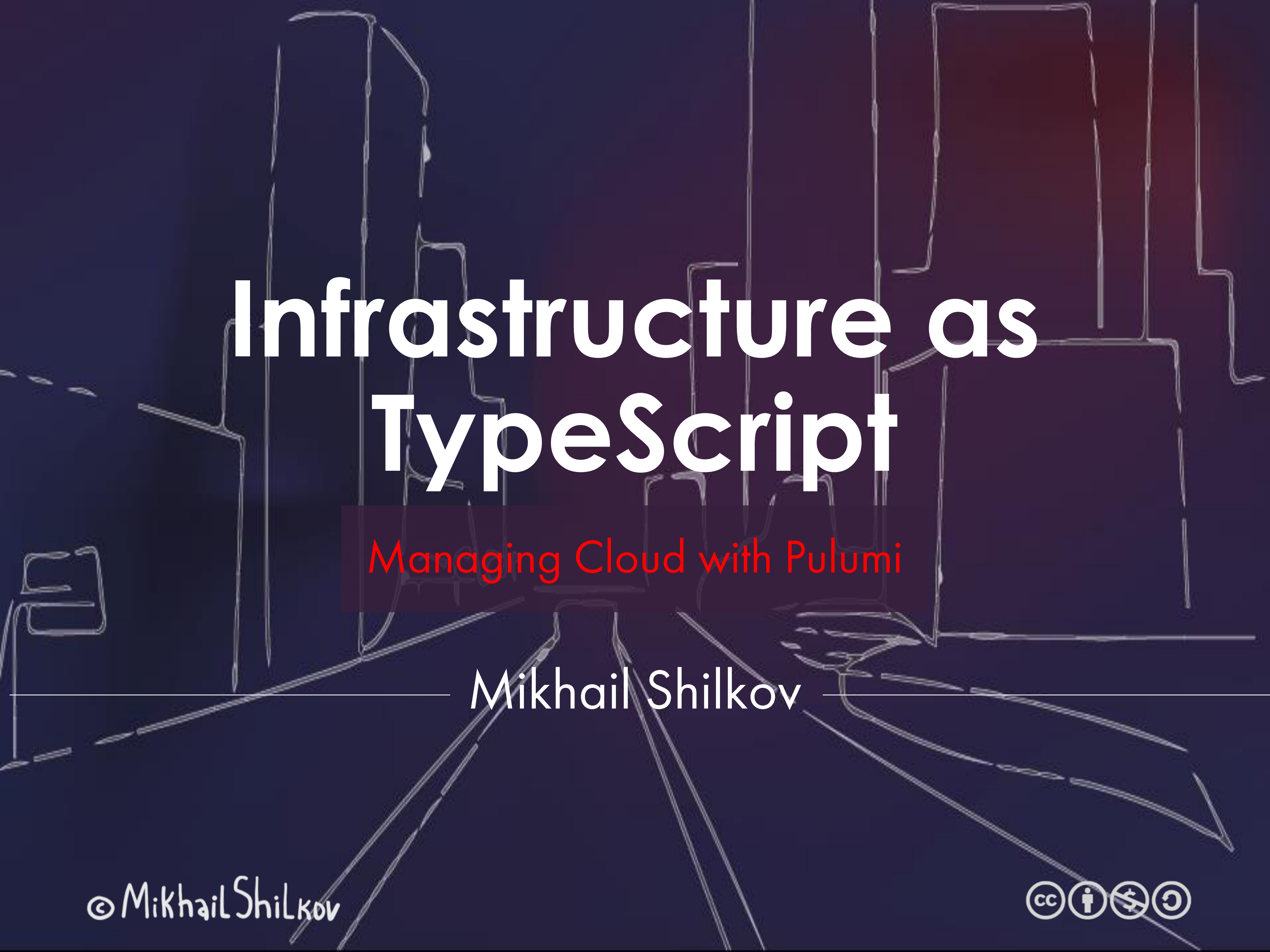


Infrastructure as TypeScript

Managing Cloud with Pulumi

Mikhail Shilkov

About me

- Software developer
- Cloud
- Serverless
- Functional Programming, F#
- Microsoft Azure MVP

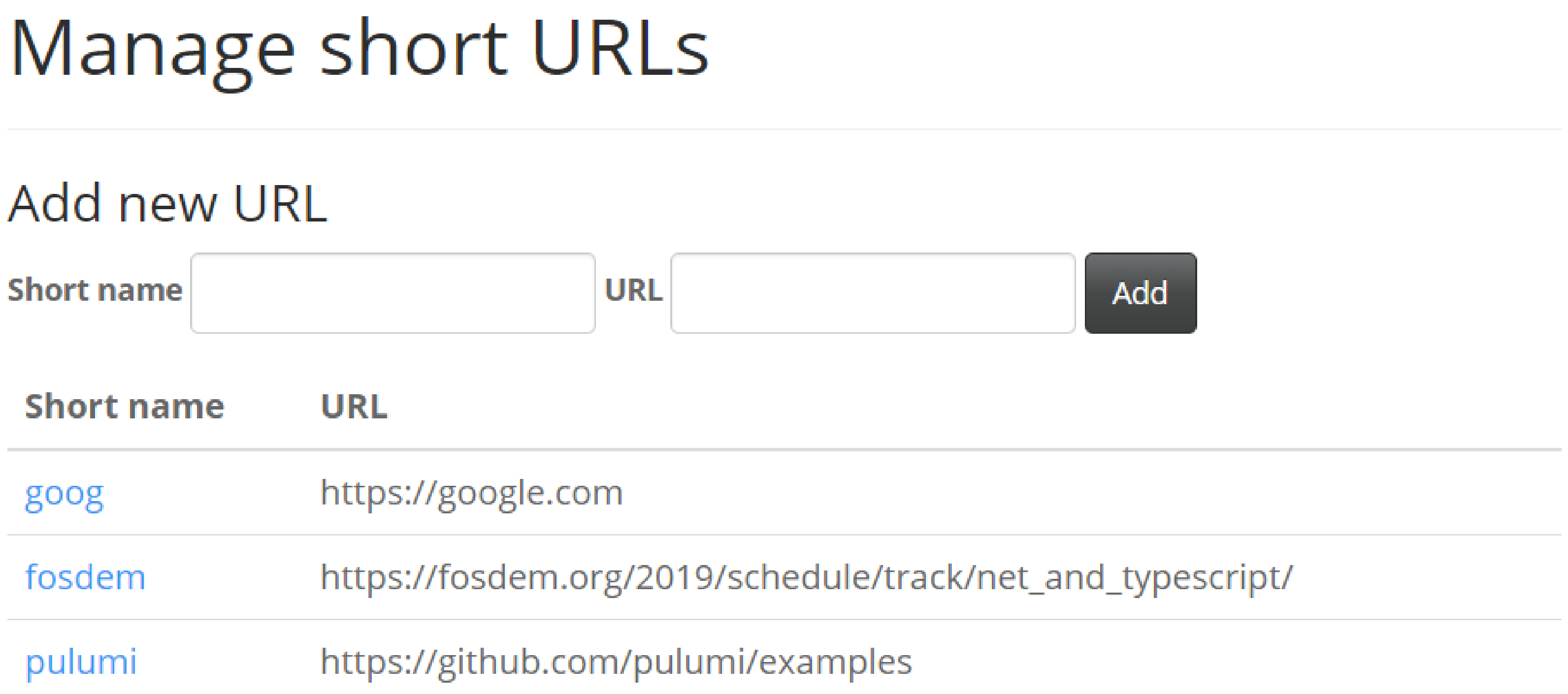
https://mikhail.io

@MikhailShilkov

Sample App

URL Shortener

URL Shortener



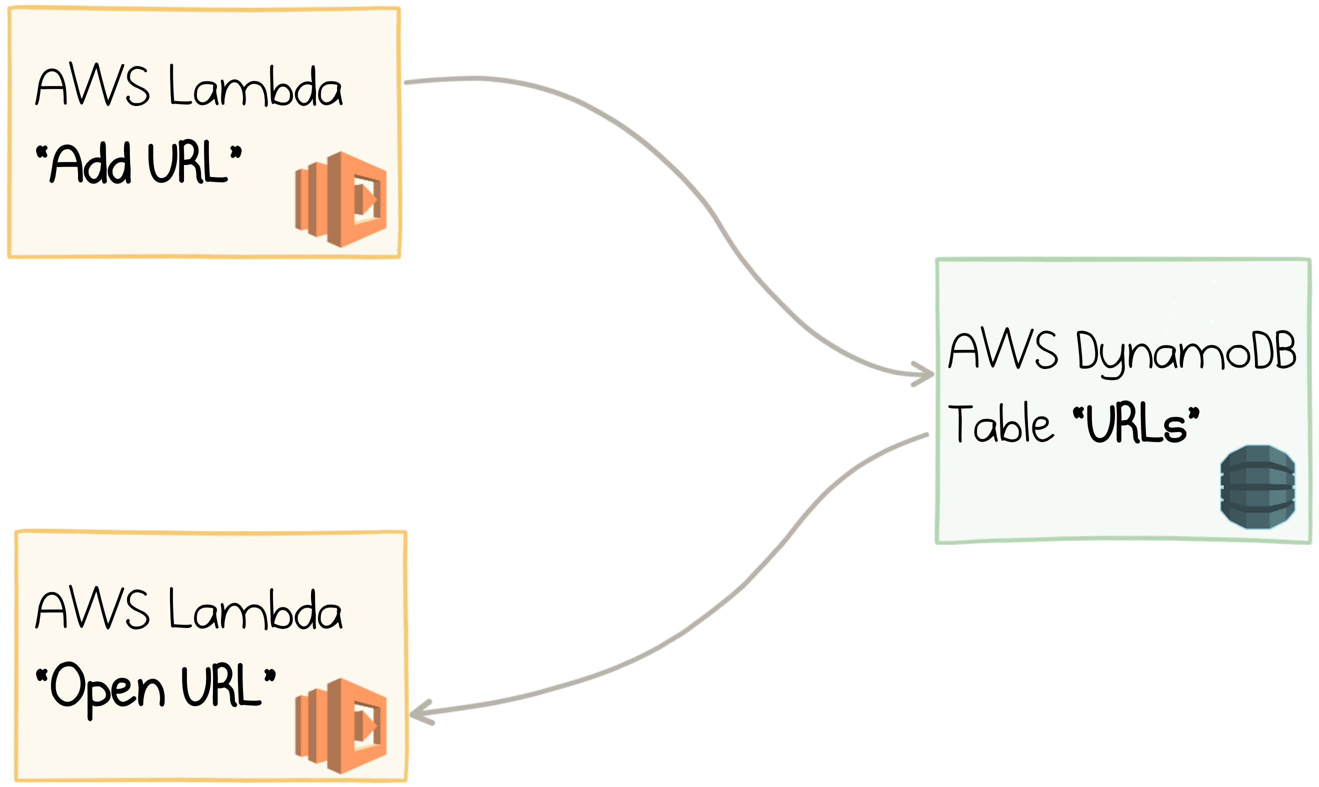
Manage short URLs

Add new URL

Short name		**URL**		Add

Short name	**URL**
goog	https://google.com
fosdem	https://fosdem.org/2019/schedule/track/net_and_typescript/
pulumi	https://github.com/pulumi/examples

Serverless URL Shortener



AWS Lambda code

```javascript
const aws = require('aws-sdk');
const table = new aws.DynamoDB.DocumentClient();

exports.handler = async (event) => {

  const name = event.path.substring(1);
  const params = { TableName: "urls", Key: { "name": name } };

  const value = await table.get(params).promise();
  const url = value && value.Item && value.Item.url;

  return url
    ? { statusCode: 301, body: "", headers: { "Location": url } }
    : { statusCode: 404, body: name + " not found" };
};
```

AWS Lambda code

```
const aws = require('aws-sdk');
const table = new aws.DynamoDB.DocumentClient();

exports.handler = async (event) => {

  const name = event.path.substring(1);
  const params = { TableName: "urls", Key: { "name": name } };

  const value = await table.get(params).promise();
  const url = value && value.Item && value.Item.url;

  return url
    ? { statusCode: 301, body: "", headers: { "Location": url } }
    : { statusCode: 404, body: name + " not found" };
};
```

AWS Lambda code

```
const aws = require('aws-sdk');
const table = new aws.DynamoDB.DocumentClient();

exports.handler = async (event) => {

  const name = event.path.substring(1);
  const params = { TableName: "urls", Key: { "name": name } };

  const value = await table.get(params).promise();
  const url = value && value.Item && value.Item.url;

  return url
    ? { statusCode: 301, body: "", headers: { "Location": url } }
    : { statusCode: 404, body: name + " not found" };
};
```

Diagram of the app with all resources

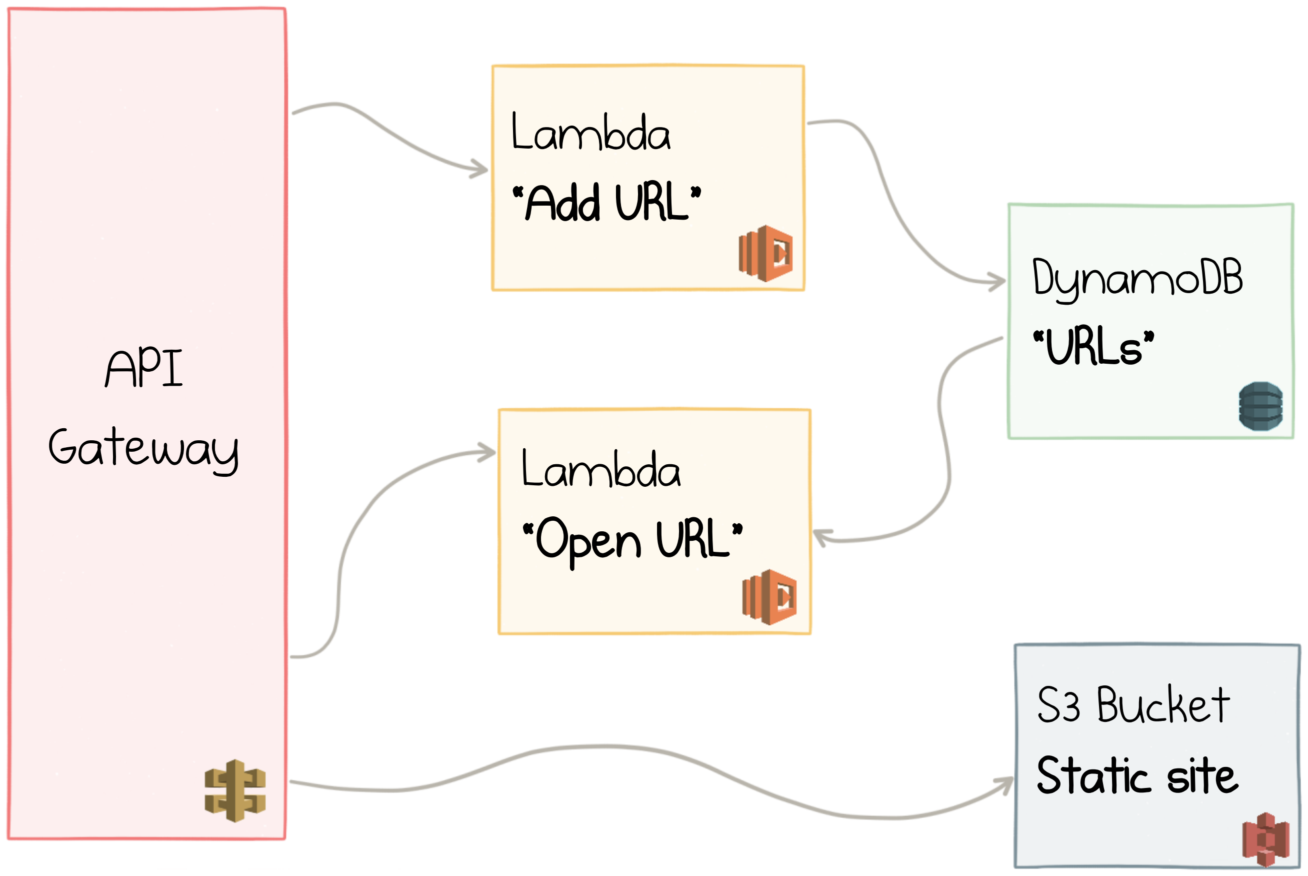
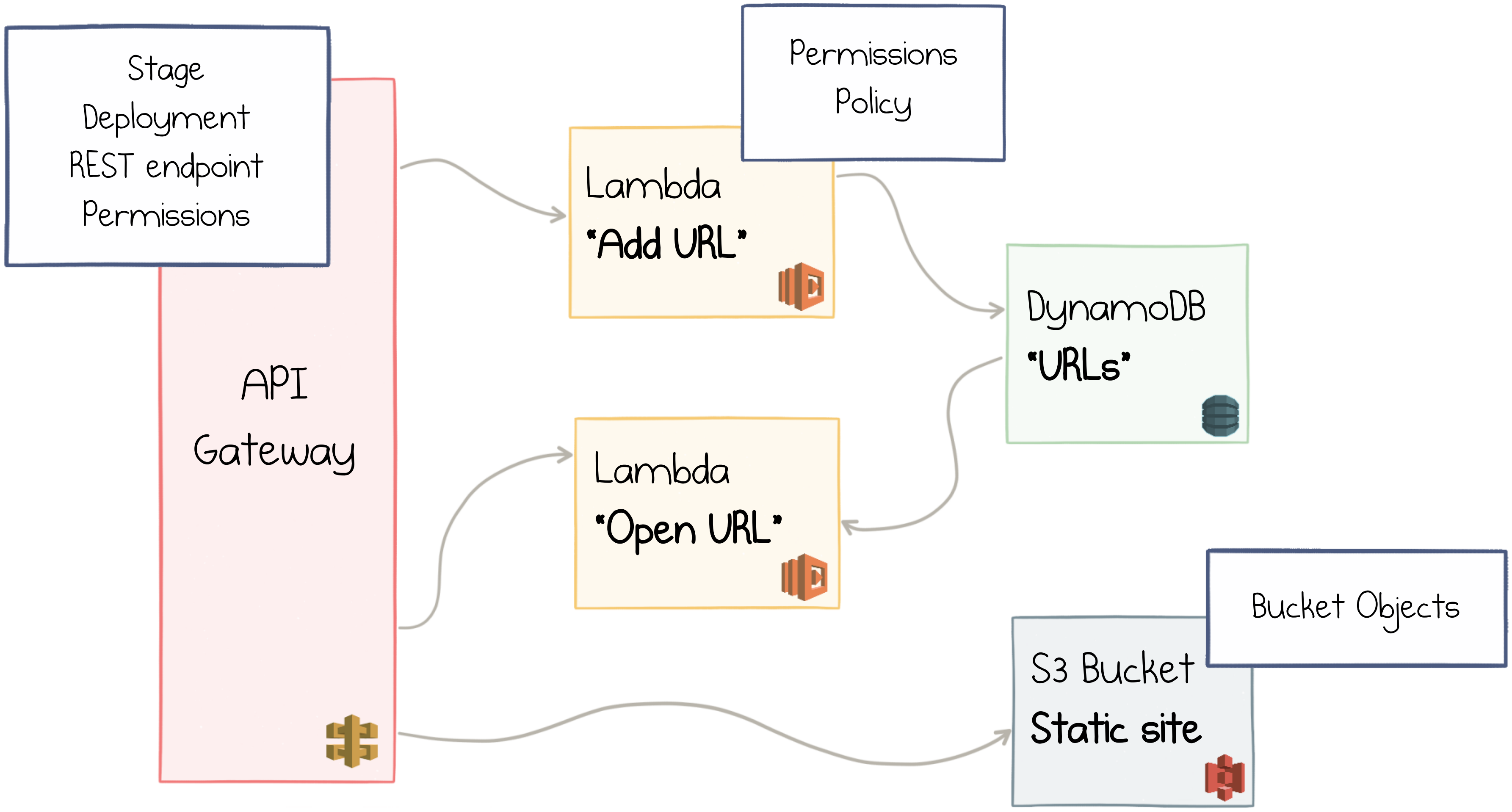
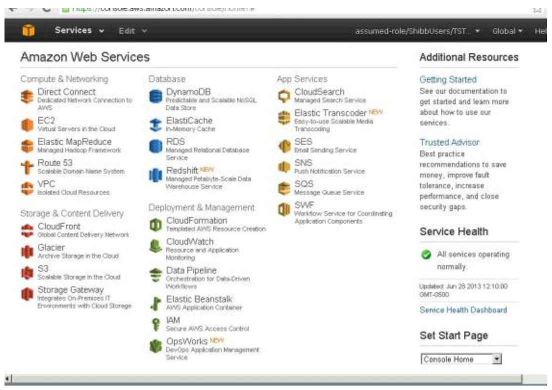


Diagram of the app with all resources



Options to deploy the infra



Amazon Web Services

Compute & Networking
- Direct Connect — Dedicated Network Connection to AWS
- EC2 — Virtual Servers in the Cloud
- Elastic MapReduce — Managed Hadoop Framework
- Route 53 — Scalable Domain Name System
- VPC — Isolated Cloud Resources

Storage & Content Delivery
- CloudFront — Global Content Delivery Network
- Glacier — Archive Storage in the Cloud
- S3 — Scalable Storage in the Cloud
- Storage Gateway — Integrates On-Premises IT Environments with Cloud Storage

Database
- DynamoDB — Predictable and Scalable NoSQL Data Store
- ElastiCache — In-Memory Cache
- RDS — Managed Relational Database Service
- Redshift NEW — Managed Petabyte-Scale Data Warehouse Service

Deployment & Management
- CloudFormation — Templated AWS Resource Creation
- CloudWatch — Resource and Application Monitoring
- Data Pipeline — Orchestration for Data-Driven Workflows
- Elastic Beanstalk — AWS Application Container
- IAM — Secure AWS Access Control
- OpsWorks NEW — DevOps Application Management Service

App Services
- CloudSearch — Managed Search Service
- Elastic Transcoder NEW — Easy-to-use Scalable Media Transcoding
- SES — Email Sending Service
- SNS — Push Notification Service
- SQS — Message Queue Service
- SWF — Workflow Service for Coordinating Application Components

Additional Resources

Getting Started
See our documentation to get started and learn more about how to use our services.

Trusted Advisor
Best practice recommendations to save money, improve fault tolerance, increase performance, and close security gaps.

Service Health

All services operating normally.

Updated: Jun 28 2013 12:10:00 GMT-0500

Service Health Dashboard

Set Start Page

Console Home

CloudFormation

SERVERLESS

Terraform

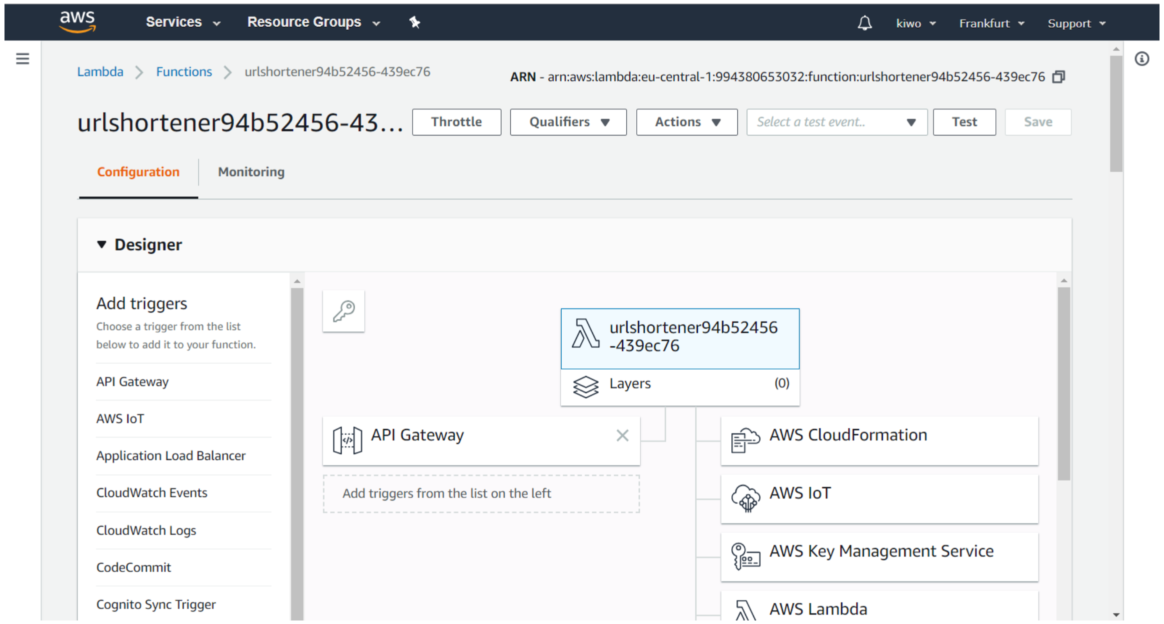
. . .

© Mikhail Shilkov

Options: AWS Web Console

Good for exploration

Not reproducible


aws
Services
Resource Groups
kiwo
Frankfurt
Support
Lambda > Functions > urlshortener94b52456-439ec76
ARN - arn:aws:lambda:eu-central-1:994380653032:function:urlshortener94b52456-439ec76
urlshortener94b52456-43...
Throttle
Qualifiers
Actions
Select a test event..
Test
Save
Configuration
Monitoring
Designer
Add triggers
Choose a trigger from the list below to add it to your function.
API Gateway
AWS IoT
Application Load Balancer
CloudWatch Events
CloudWatch Logs
CodeCommit
Cognito Sync Trigger
urlshortener94b52456 -439ec76
Layers (0)
API Gateway
Add triggers from the list on the left
AWS CloudFormation
AWS IoT
AWS Key Management Service
AWS Lambda

https://console.aws.amazon.com

Options: AWS CLI

Scriptable

Imperative,
“how to do” not
“what to do”

```
aws apigateway create-resource --rest-api-id 1234123412 --parent-id a1b2c3 --path-part 'n
ew-resource'
```

```
aws apigateway create-stage --rest-api-id 1234123412 --stage-name 'dev' --description 'De
velopment stage' --deployment-id a1b2c3 --variables key='value',otherKey='otherValue'
```

```
aws apigateway create-rest-api --name 'My First API' --description 'This is my first API'
```

https://docs.aws.amazon.com/cli/latest/reference/

Options: CloudFormation

Desired state configuration with YAML

Verbose

```
Resources:
  S3BucketForURLs:
    Type: "AWS::S3::Bucket"
    DeletionPolicy: Delete
    Properties:
      BucketName: !If [ "CreateNewBucket", "AWS …
      WebsiteConfiguration:
        IndexDocument: "index.html"
      LifecycleConfiguration:
        Rules:
          -
            Id: DisposeShortUrls
            ExpirationInDays: !Ref URLExpiration
            Prefix: "u"
            Status: Enabled
```

https://aws.amazon.com/blogs/compute/build-a-serverless-private-url-shortener/

Options: Terraform

Proprietary format

Multi-cloud

```
resource "aws_lambda_function" "apply_security_headers" {
  provider = "aws.cloudfront_acm"
  filename = "lambda_functions/security_headers.zip"
  function_name = "apply_security_headers"
  role = "${aws_iam_role.short_url_lambda_iam.arn}"
  handler = "lambda_function.handler"
  source_code_hash = "${data.archive.security.base64}"
  runtime = "nodejs8.10"
  publish = true
  tags = {
    Project = "short_urls"
  }
}
```

https://github.com/jamesridgway/aws-lambda-short-url

Options: Serverless Framework

YAML

Succinct

Narrow scope

Multi-cloud

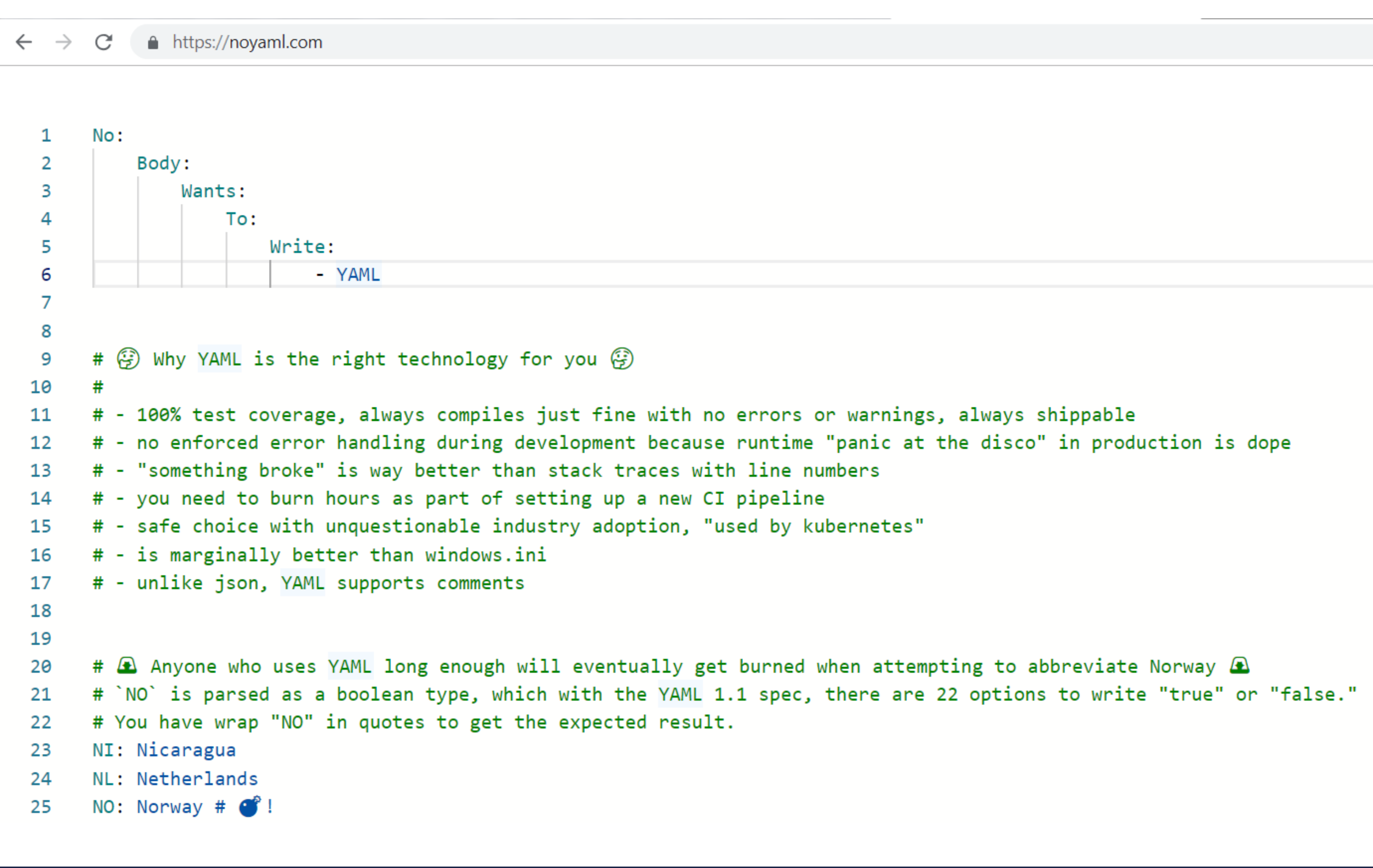
```yaml
functions:
  store:
    handler: api/store.handle
    events:
      - http:
          path: /
          method: post
          cors: true

resources:
  Resources:
    ServerlesslyRedirectS3Bucket:
      Type: AWS::S3::Bucket
      Properties:
        BucketName: ${file(config.json):BUCKET}
        AccessControl: PublicRead
        WebsiteConfiguration:
```

https://github.com/danielireson/serverless-url-shortener

Desired properties of infra

- Scriptable
- Applicable for any cloud service
- Reproducible
- Multi-cloud (+ hybrid)
- Desired state configuration
- Language?



https://noyaml.com

```yaml
No:
    Body:
        Wants:
            To:
                Write:
                    - YAML


# 🤔 Why YAML is the right technology for you 🤔
#
# - 100% test coverage, always compiles just fine with no errors or warnings, always shippable
# - no enforced error handling during development because runtime "panic at the disco" in production is dope
# - "something broke" is way better than stack traces with line numbers
# - you need to burn hours as part of setting up a new CI pipeline
# - safe choice with unquestionable industry adoption, "used by kubernetes"
# - is marginally better than windows.ini
# - unlike json, YAML supports comments


# 🚑 Anyone who uses YAML long enough will eventually get burned when attempting to abbreviate Norway 🚑
# `NO` is parsed as a boolean type, which with the YAML 1.1 spec, there are 22 options to write "true" or "false."
# You have wrap "NO" in quotes to get the expected result.
NI: Nicaragua
NL: Netherlands
NO: Norway # 💣!
```

© Mikhail Shilkov

https://noyaml.com

pulumi

- Scriptable
- Applicable for any cloud service
- Reproducible
- Multi-cloud (+ hybrid)
- Desired state configuration
- ~~YAML~~ TypeScript (or Python, Go, ...)

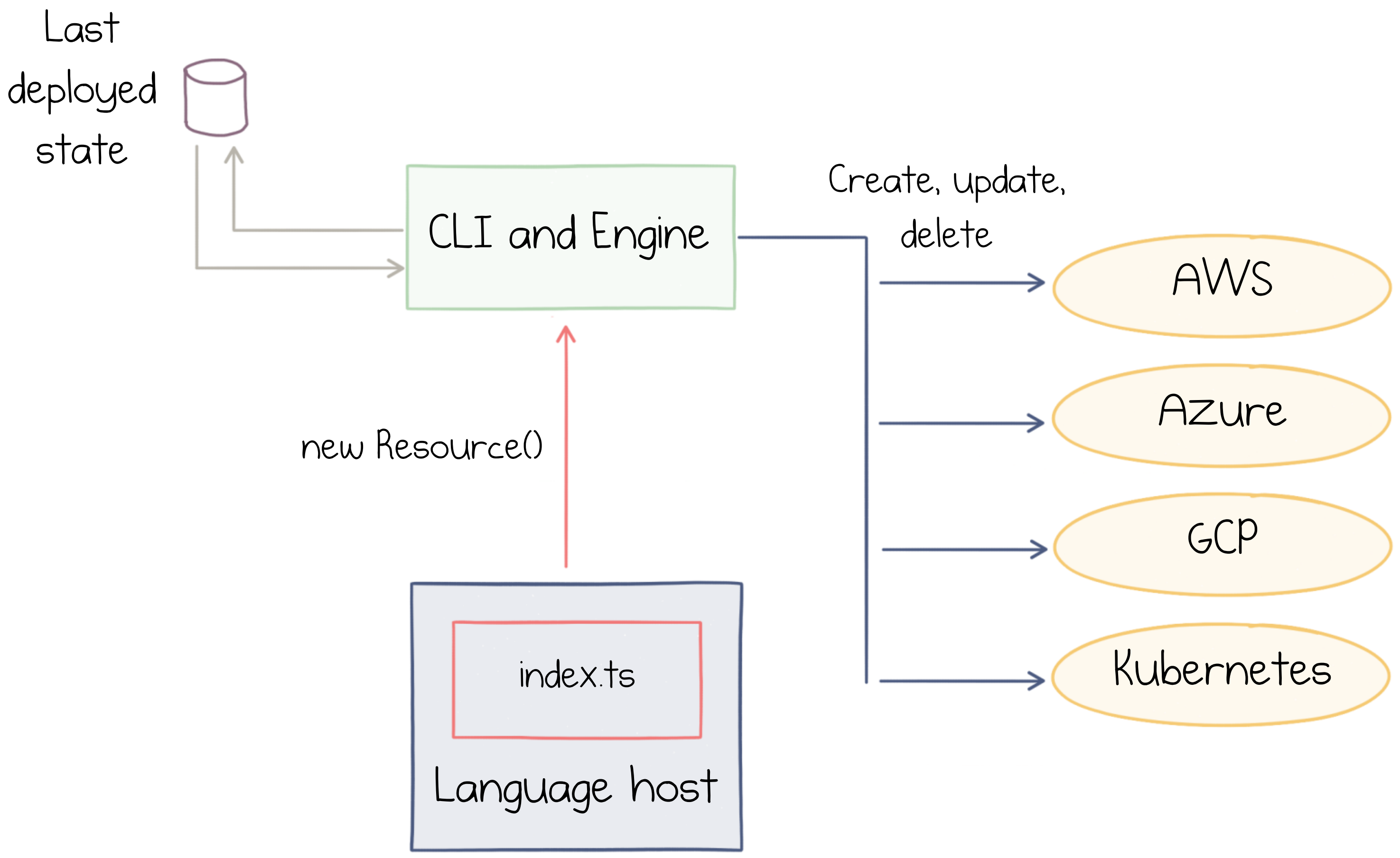
Pulumi

Demo

© Mikhail Shilkov

Pulumi CLI
Stacks
State

How Pulumi works

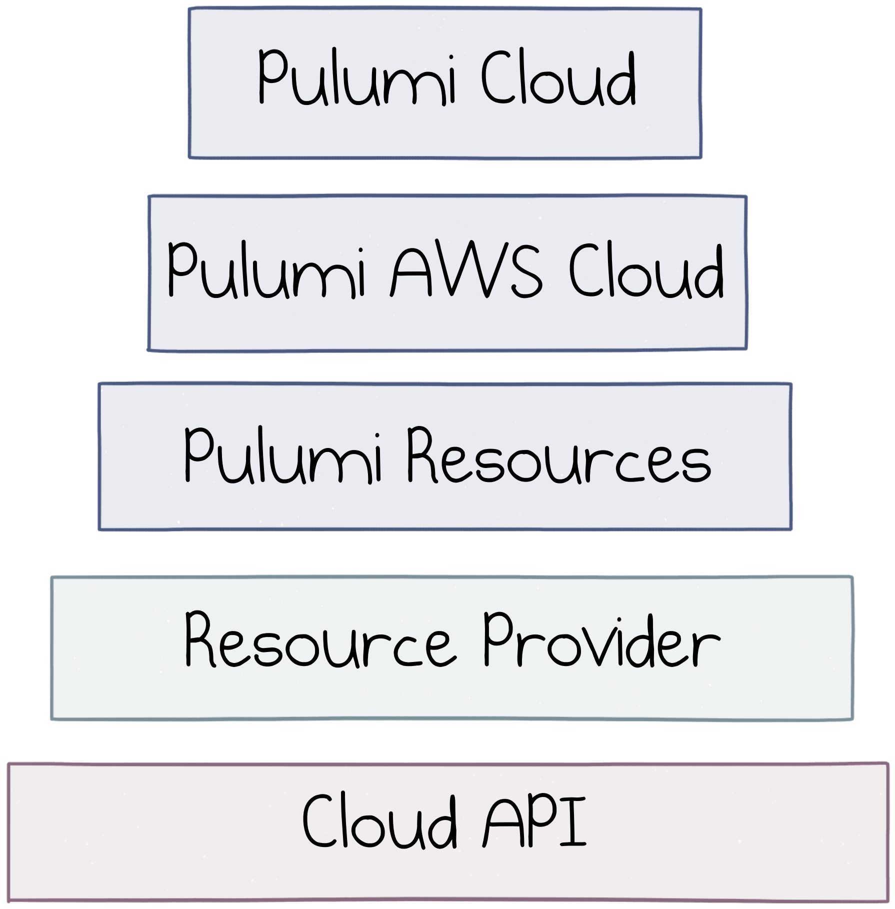


© Mikhail Shilkov

Demo

Types
IntelliSense
Compiler errors
Language Constructs
Reusable Components
Component Library
Blend of Infra and Code
Pulumi Cloud

Pulumi Layers



Conclusions

Making Cloud Apps?

USE INFRASTRUCTURE-AS-CODE

What Kind of "Code"?

Real Programming Language!

TypeScript

Useful Links

Slides and demos:
https://github.com/MikhailShilkov/fosdem2019

Pulumi:
https://pulumi.io/

Usage examples:
https://github.com/pulumi/examples

Thank you!

@MikhailShilkov

https://mikhail.io