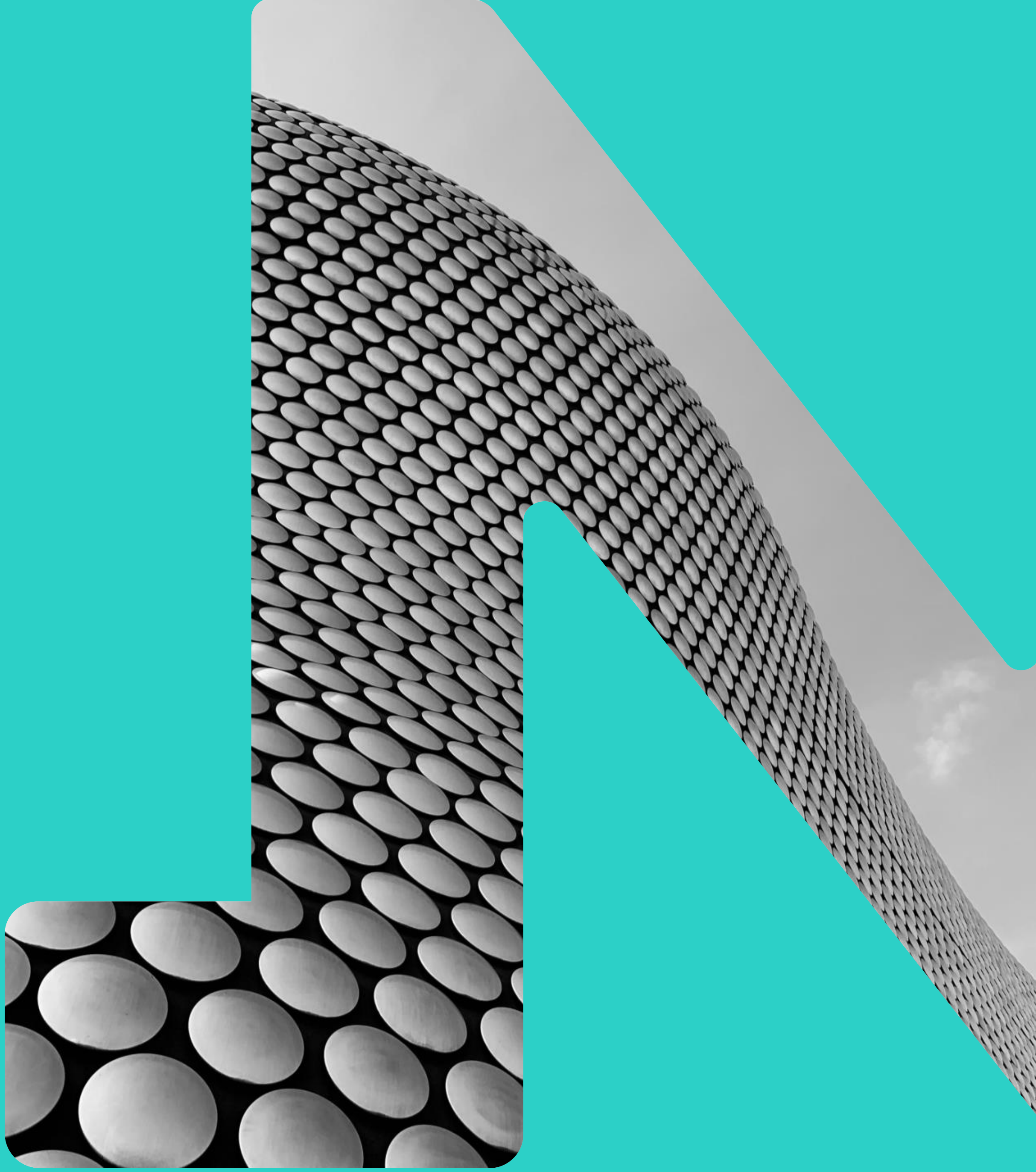
NOEST

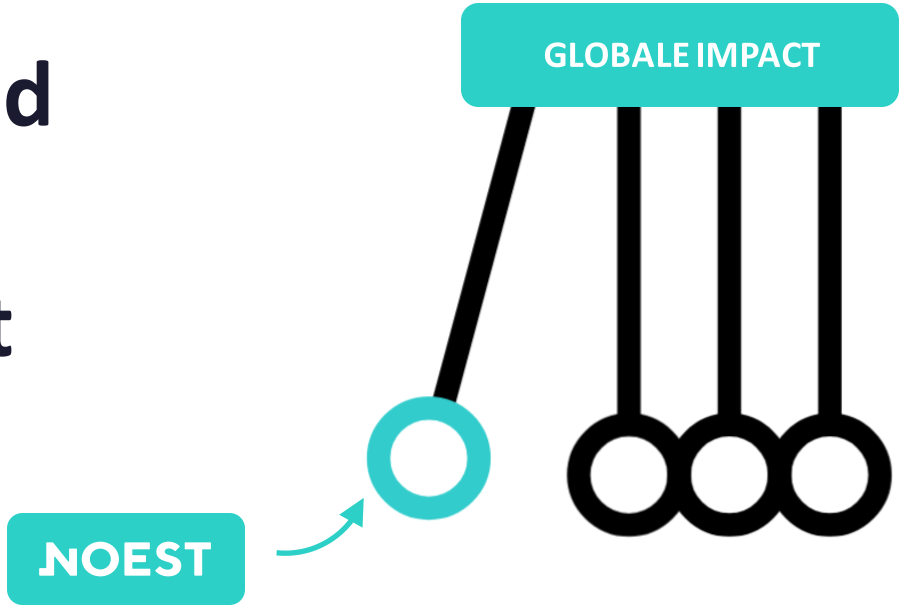
# Software Architecture Showcase

NIELS STUBBE & ANGELO DEJAEGHERE

25/04/2024



# Wij maken jouw wereld slimmer en efficiënter door lokale kracht met globale impact



# Software Architecture Showcase

Part 1

**Modernizing Legacy Applications**

**- Niels Stubbe**

Part 2

**Modular Monolith**

**- Angelo Dejaeghere**

# Modernizing Legacy Applications

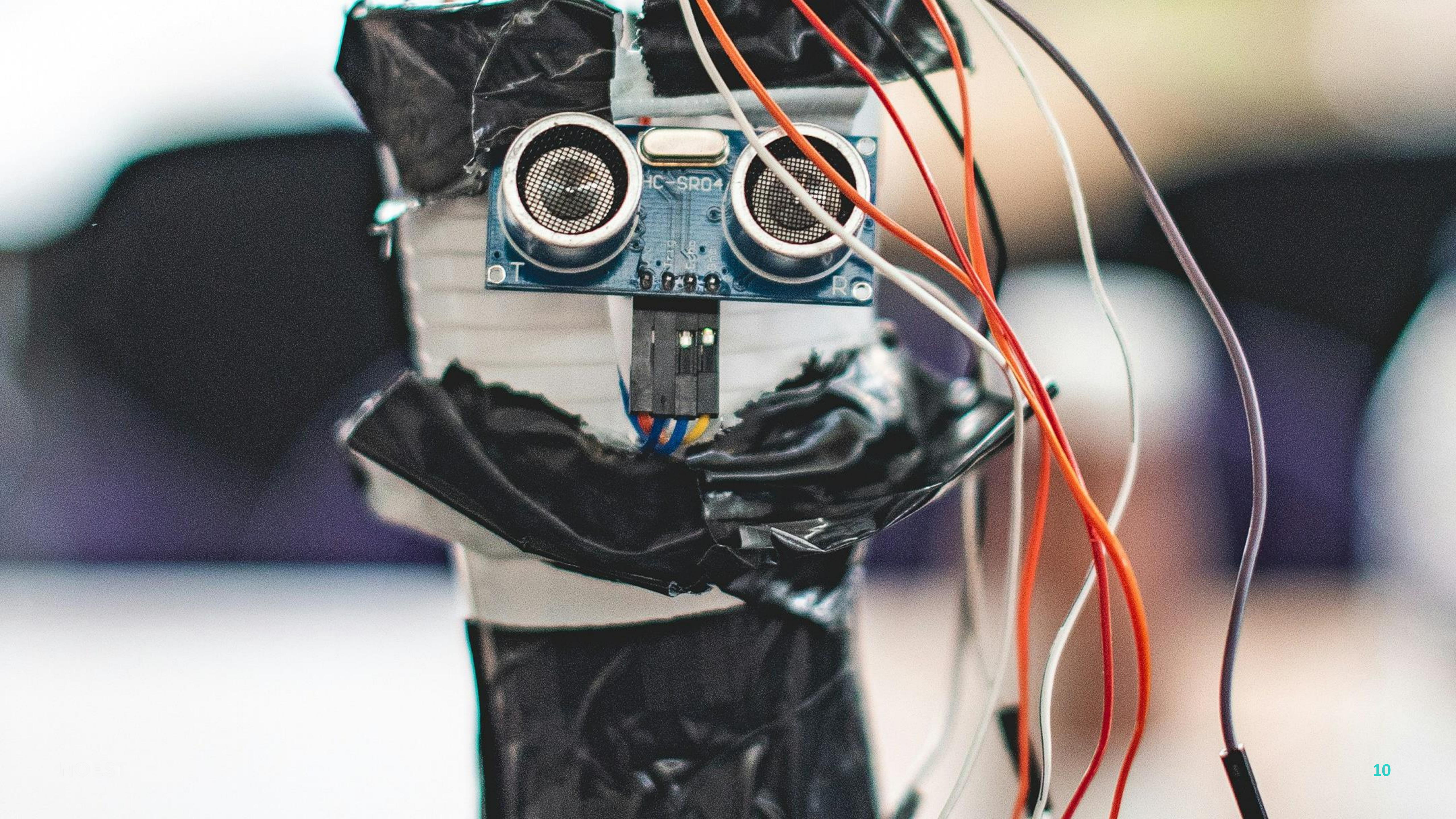
ARCHITECTURE AS AN ENGINE FOR CHANGE

NOEST

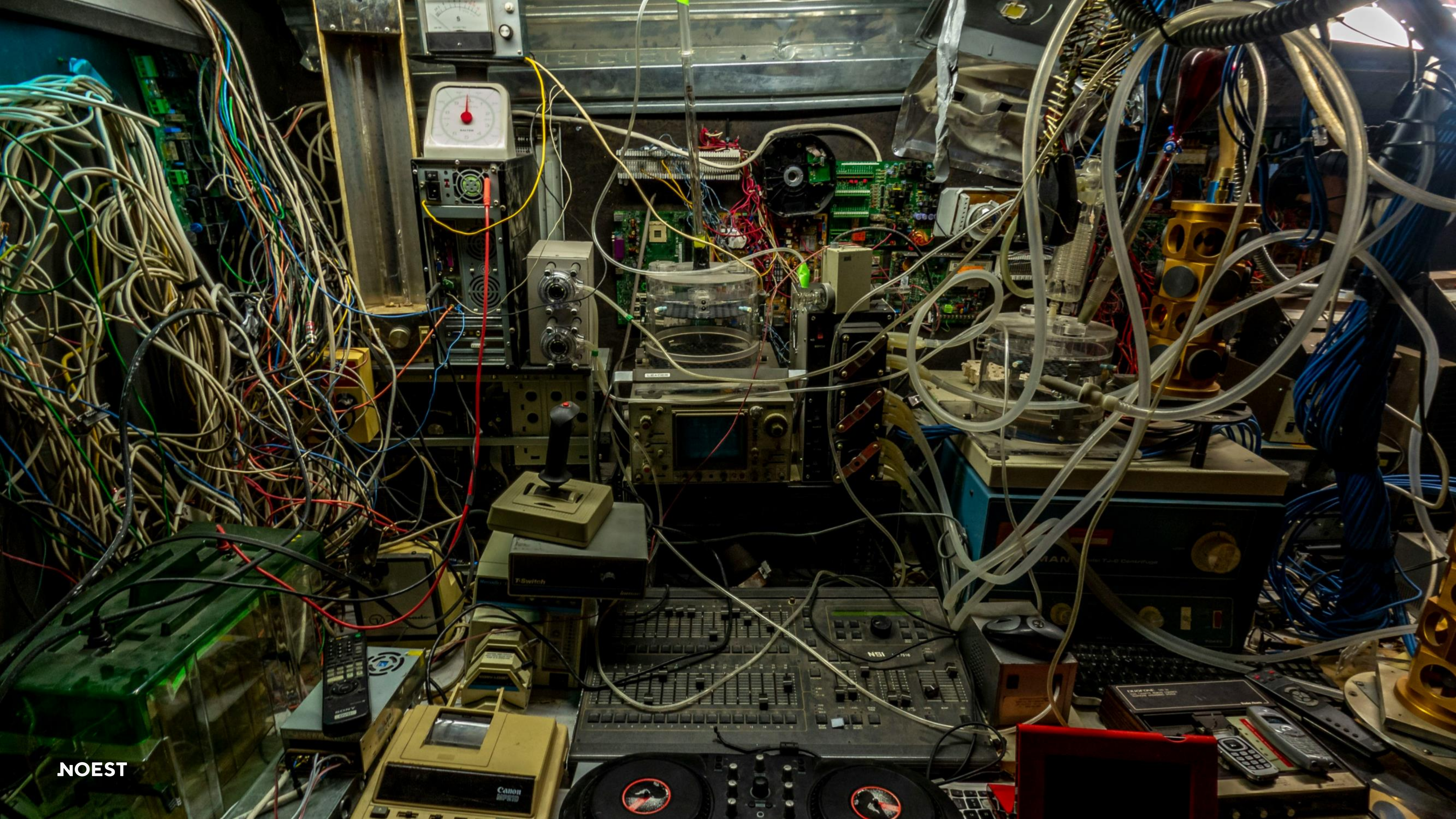
# 10 years ago…

# 5 years ago...

# 1 year ago...


HC-SR04
T
R
Trig
Echo
Gnd

# Today



.NOEST

NOEST

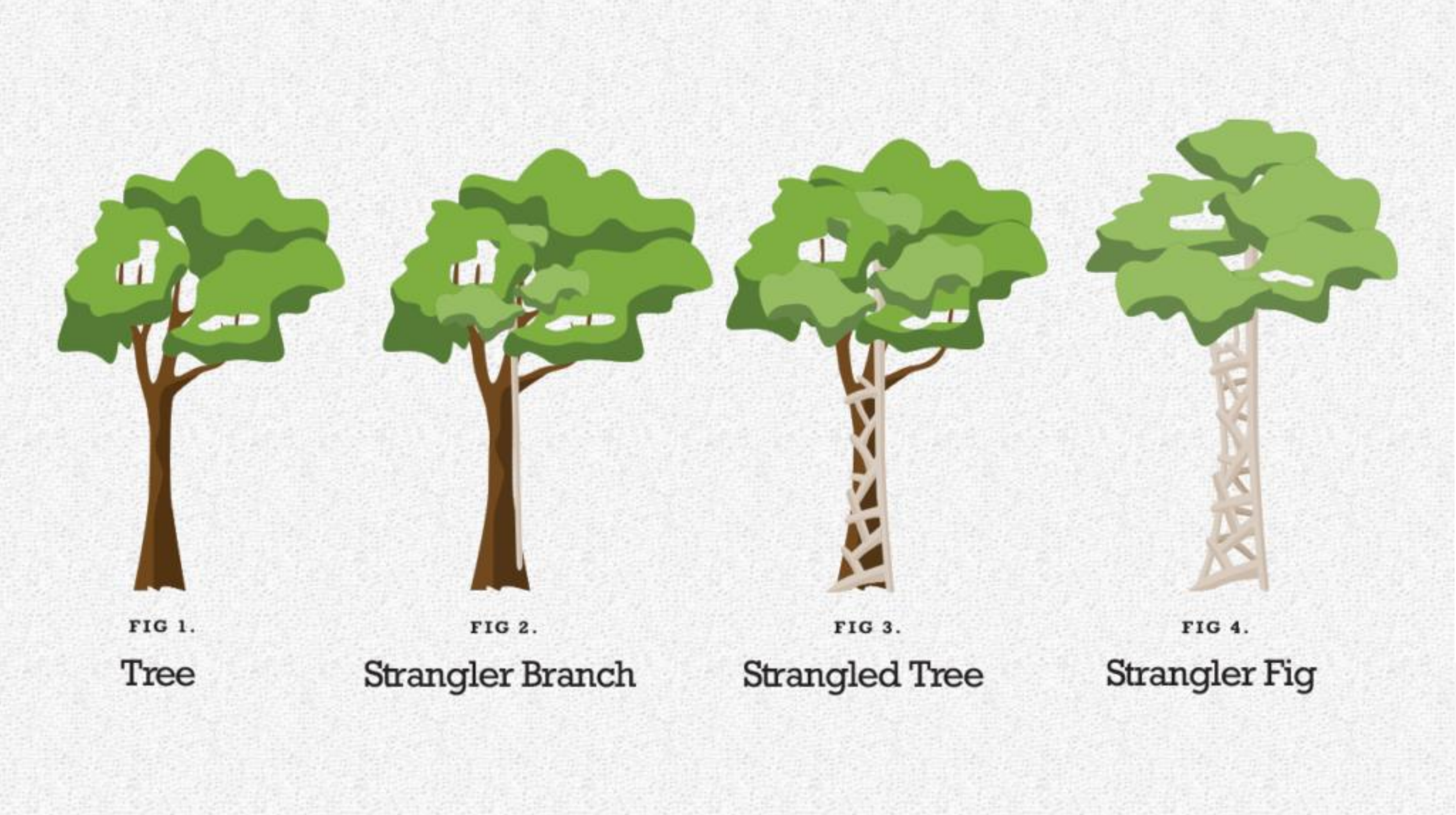
# Herkenbaar?

# Rewrite

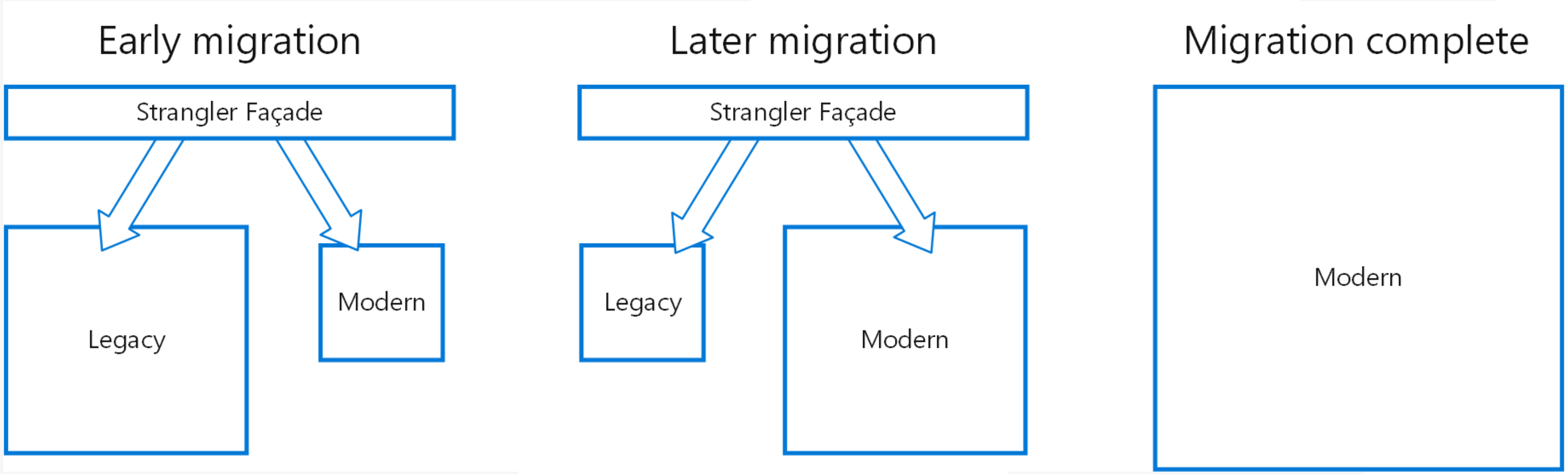
- Te duur
- Geen tijd, we moeten features ontwikkelen!
- Het zal weer hetzelfde resultaat zijn
- Dubbel ontwikkelwerk

# Is there any alternative?

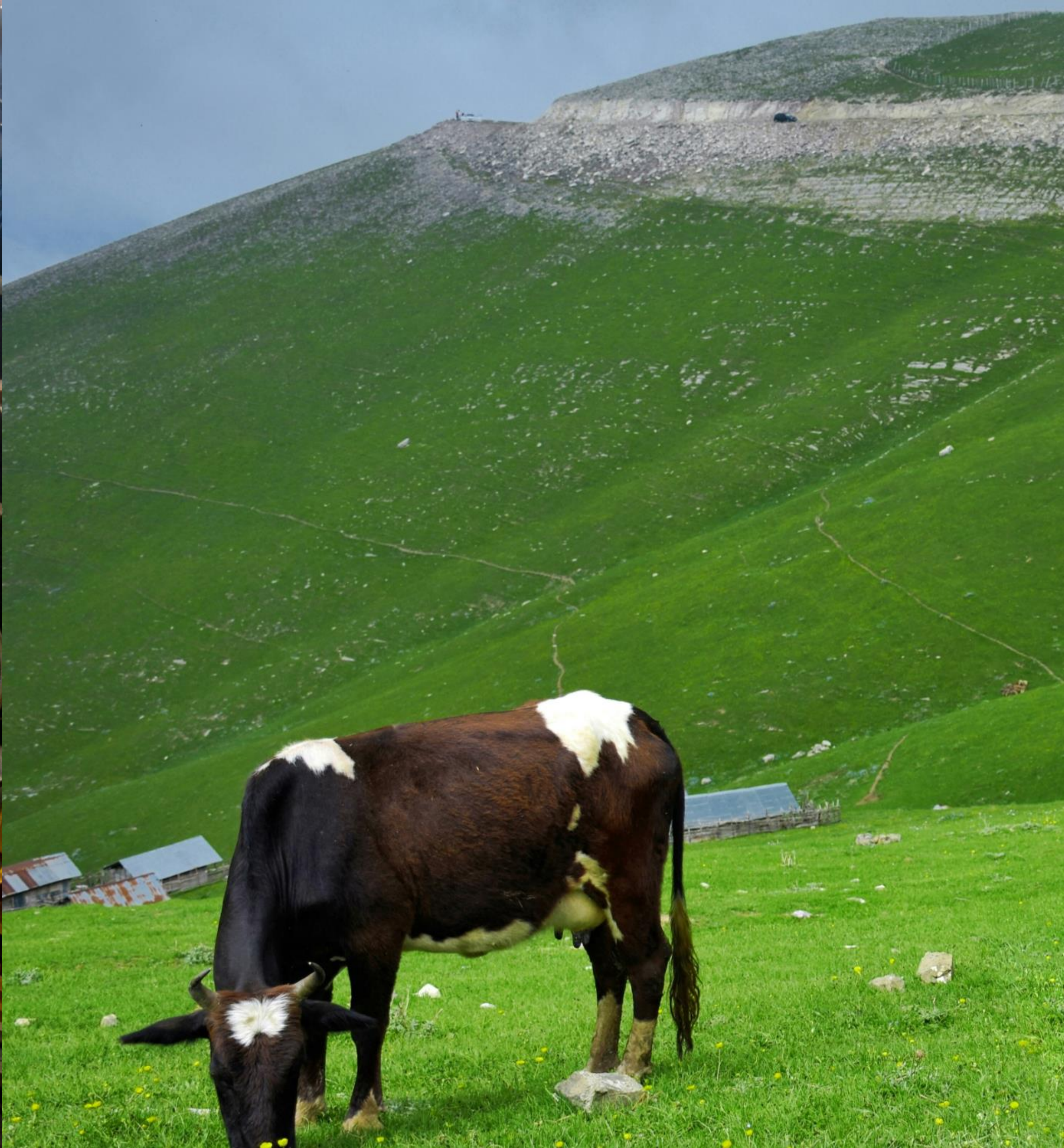
# Strangler (vine/fig) Pattern



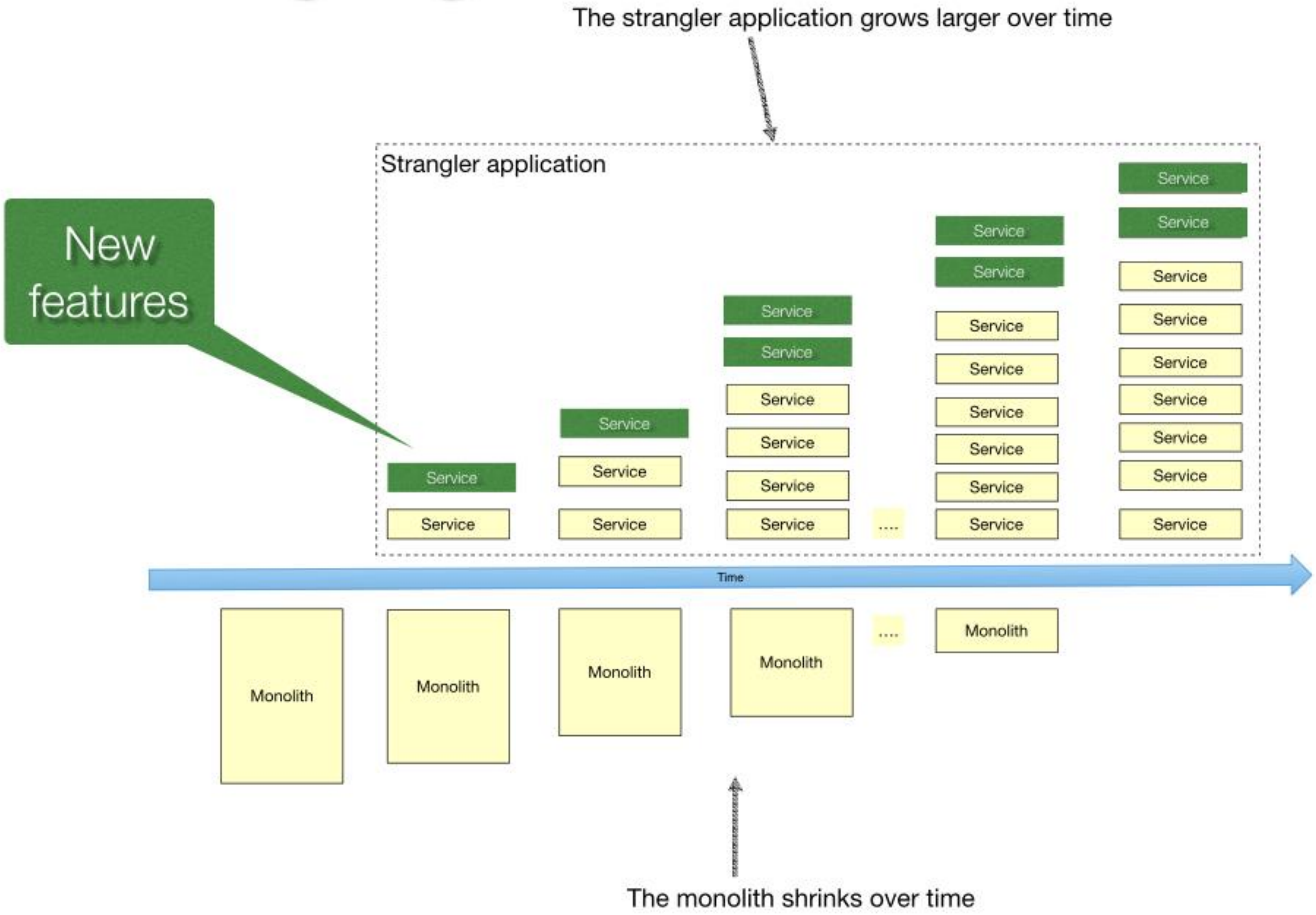
# Strangler Pattern



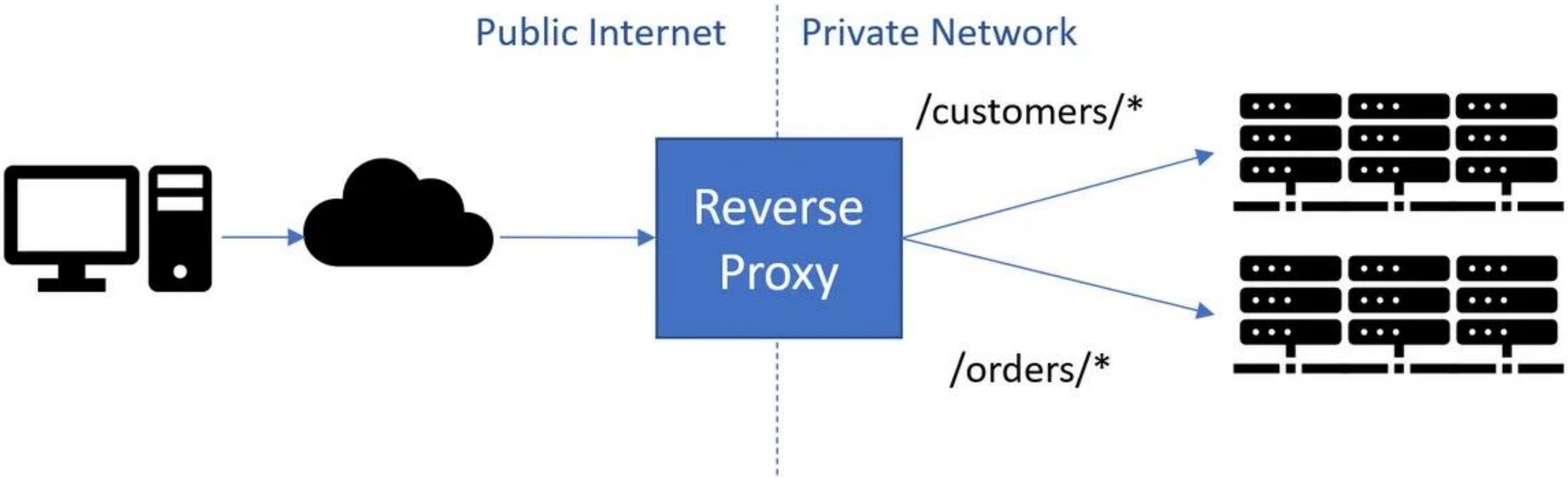
NSI



# Strangling the monolith



# Reverse Proxy



# Reverse Proxy

## On-prem

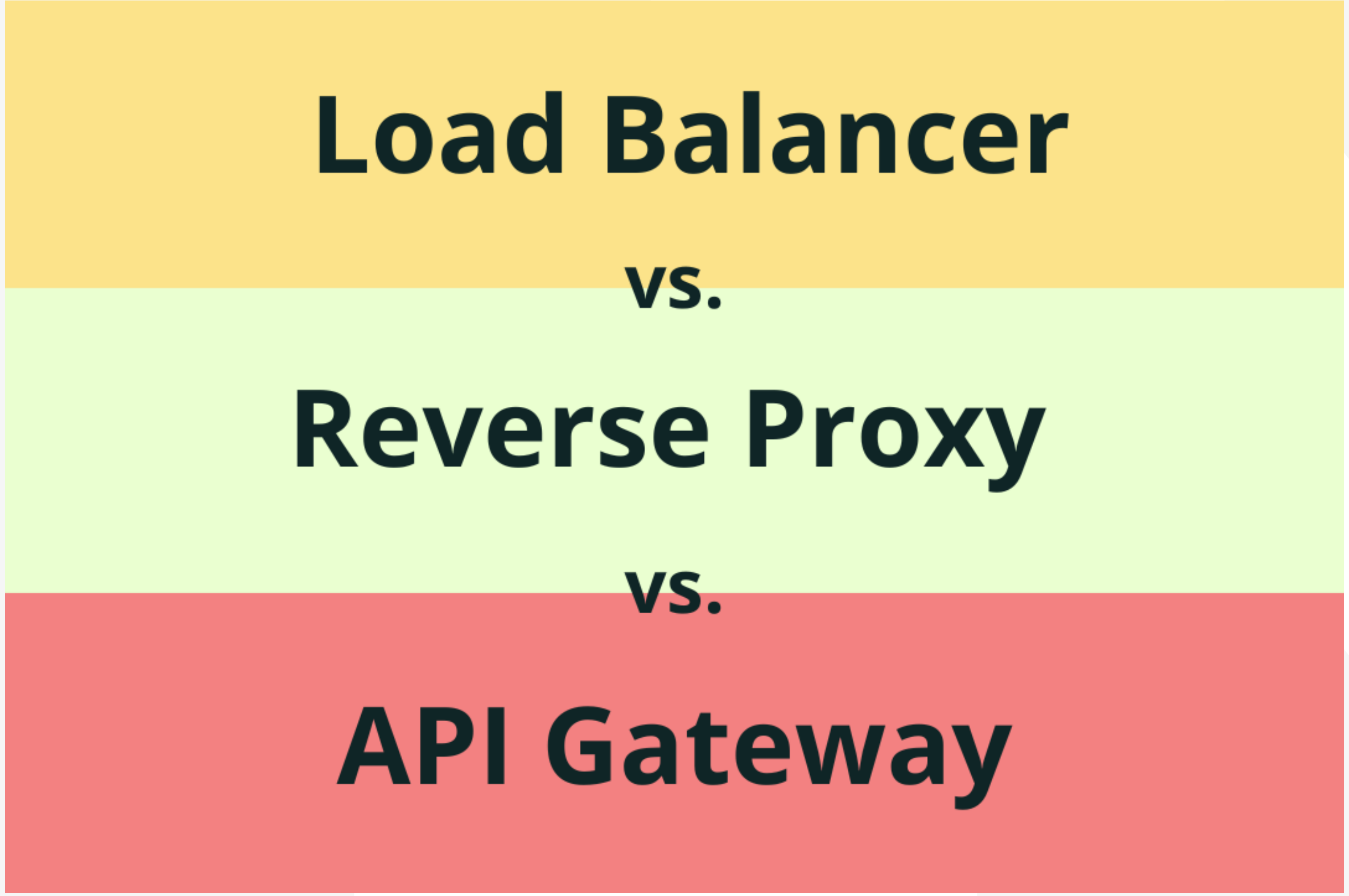
- Nginx
- IIS
- YARP

## Azure

- Azure API Management
- YARP

Somerfield

# (More than) Yet Another Reverse Proxy



Load Balancer

vs.

Reverse Proxy

vs.

API Gateway

# (More than) Yet Another Reverse Proxy

Load Balancer

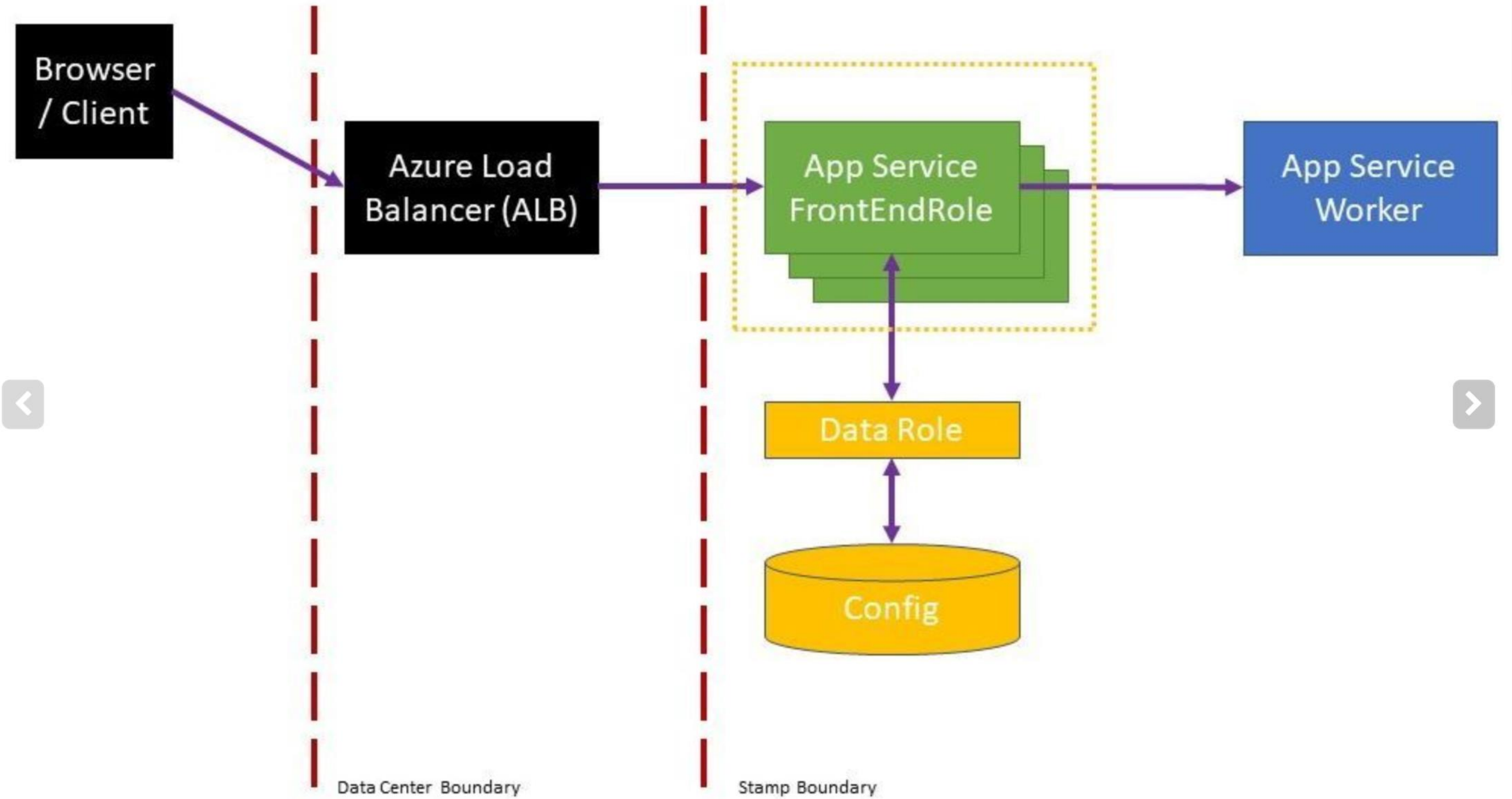
vs.

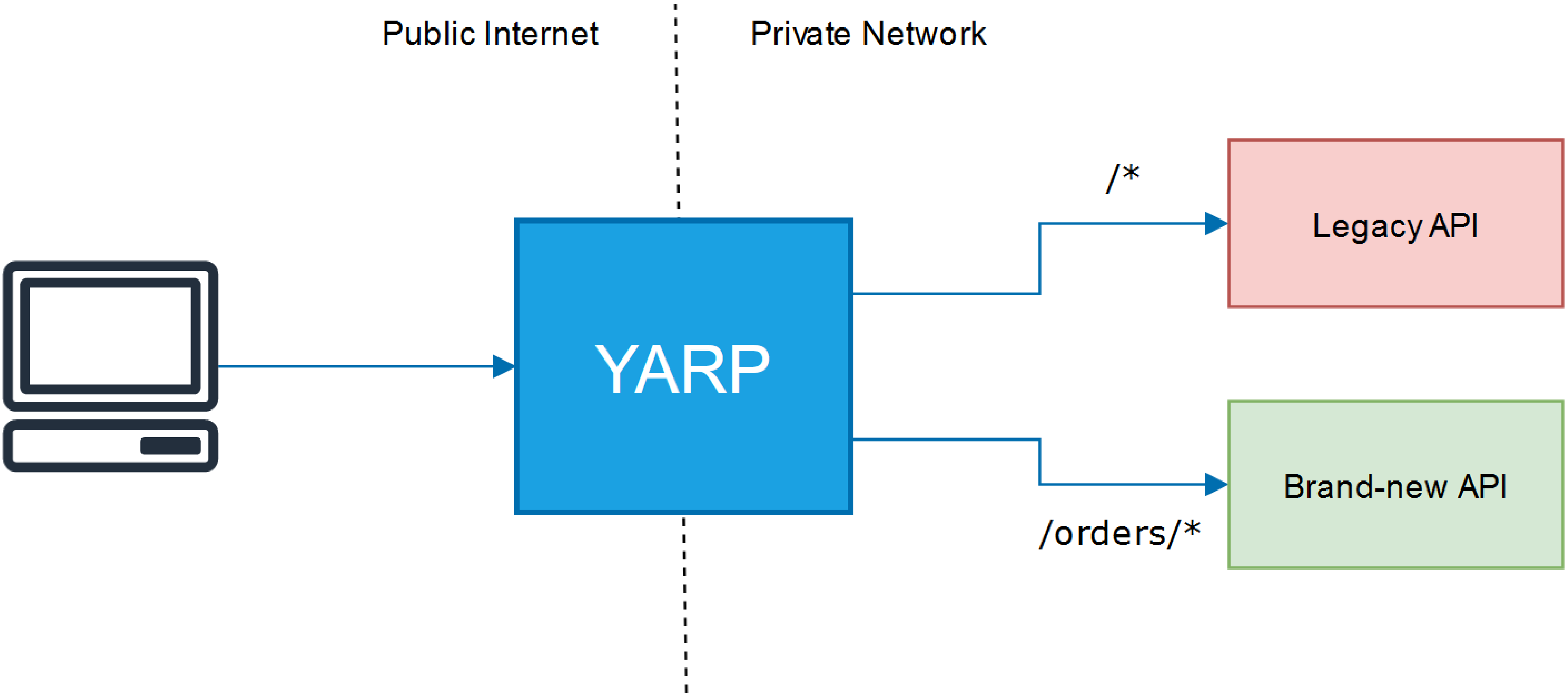
Reverse Proxy

vs.

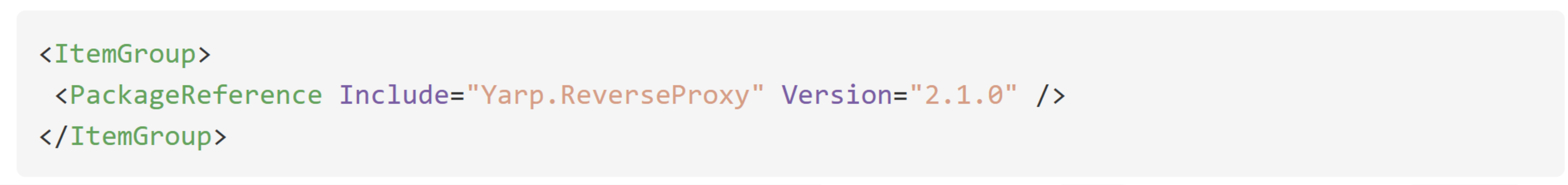
API Gateway

YARP

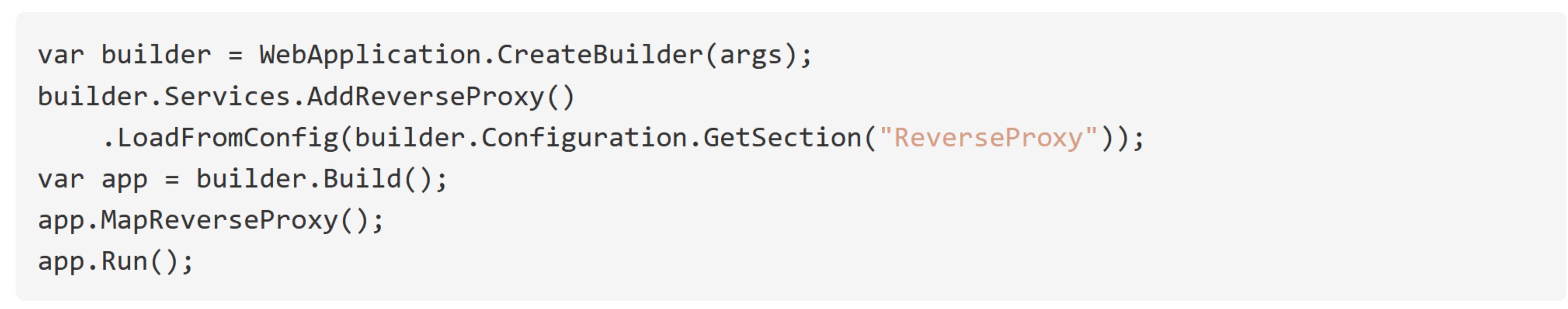

Browser / Client
Azure Load Balancer (ALB)
App Service FrontEndRole
App Service Worker
Data Role
Config
Data Center Boundary
Stamp Boundary


Public Internet
Private Network
YARP
/*
Legacy API
/orders/*
Brand-new API

# YARP Configuration



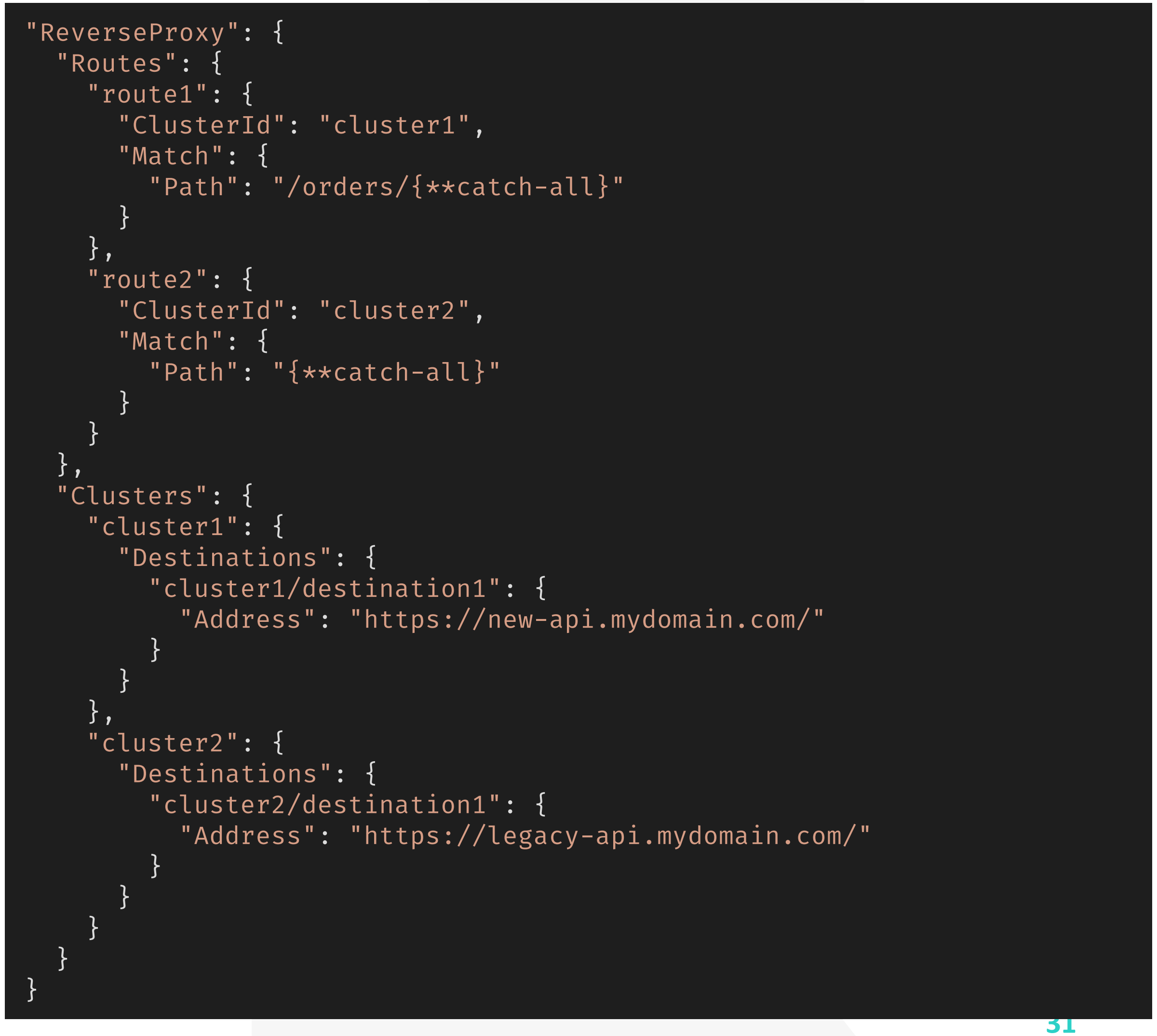
```
<ItemGroup>
 <PackageReference Include="Yarp.ReverseProxy" Version="2.1.0" />
</ItemGroup>
```



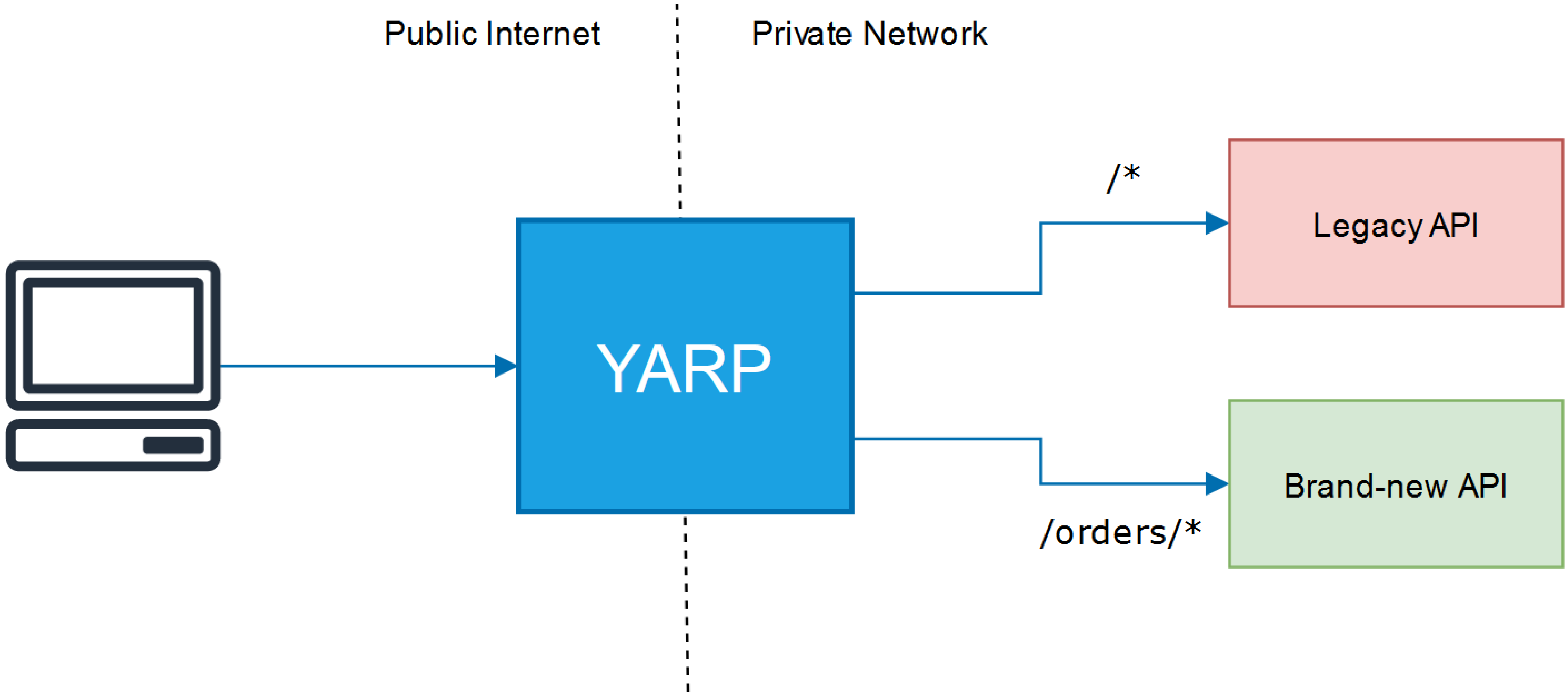
```
var builder = WebApplication.CreateBuilder(args);
builder.Services.AddReverseProxy()
    .LoadFromConfig(builder.Configuration.GetSection("ReverseProxy"));
var app = builder.Build();
app.MapReverseProxy();
app.Run();
```

# YARP Configuration

**Route**: matches incoming requests
**Cluster**: group of destinations
**Destination:** points to a specific address

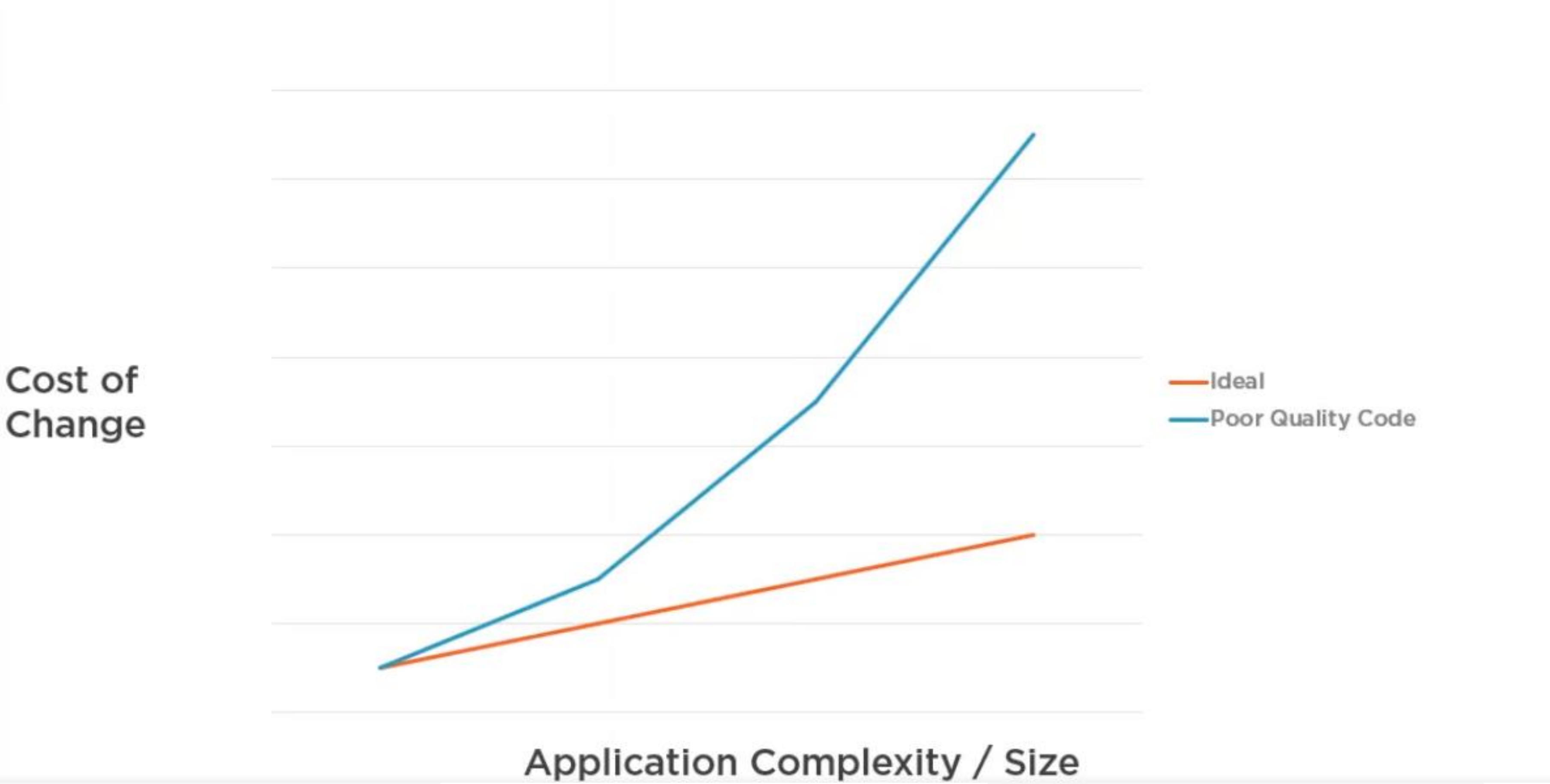


```json
"ReverseProxy": {
  "Routes": {
    "route1": {
      "ClusterId": "cluster1",
      "Match": {
        "Path": "/orders/{**catch-all}"
      }
    },
    "route2": {
      "ClusterId": "cluster2",
      "Match": {
        "Path": "{**catch-all}"
      }
    }
  },
  "Clusters": {
    "cluster1": {
      "Destinations": {
        "cluster1/destination1": {
          "Address": "https://new-api.mydomain.com/"
        }
      }
    },
    "cluster2": {
      "Destinations": {
        "cluster2/destination1": {
          "Address": "https://legacy-api.mydomain.com/"
        }
      }
    }
  }
}
```

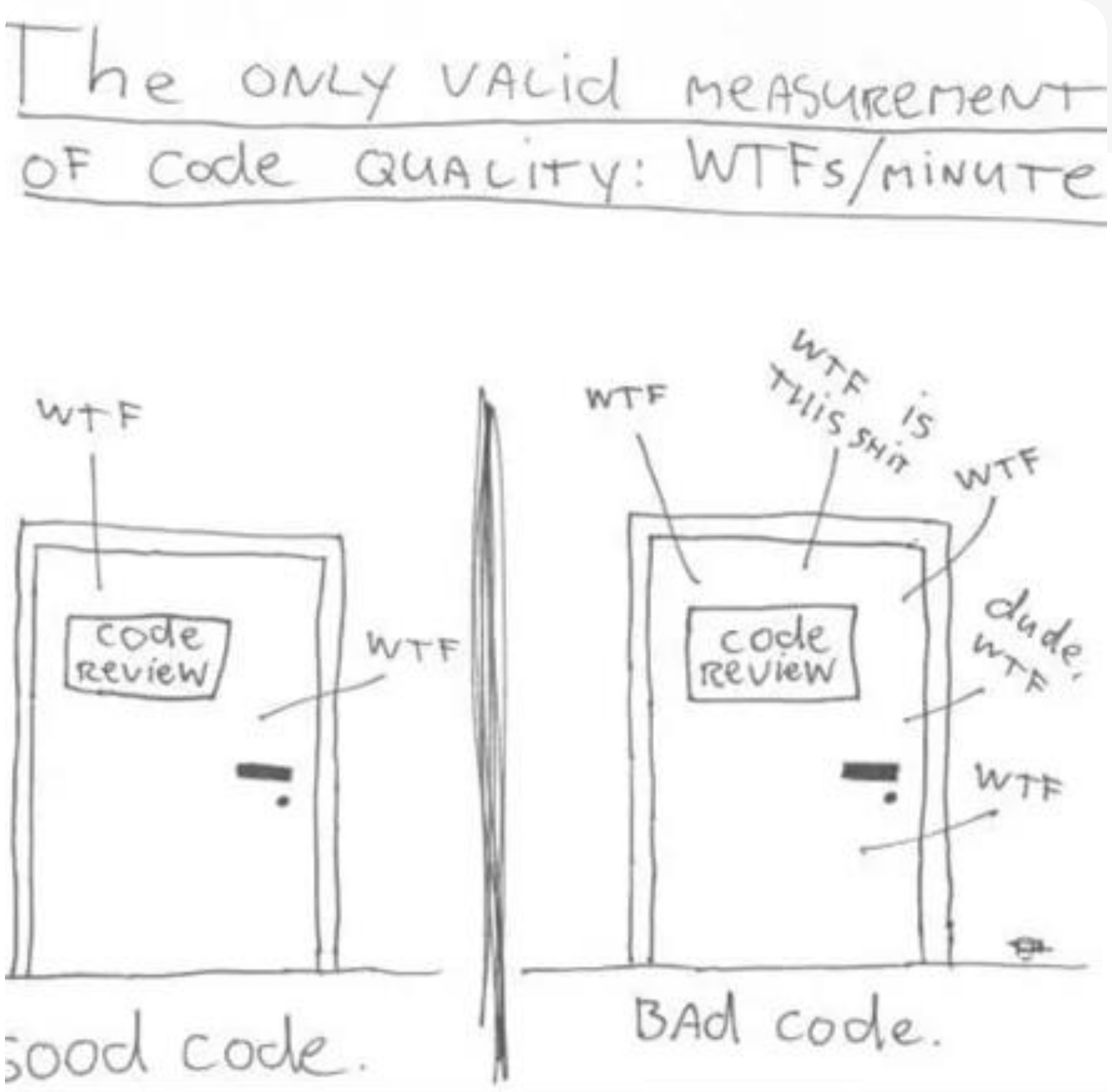

Public Internet
Private Network
YARP
/*
Legacy API
Brand-new API
/orders/*

# Time to migrate!

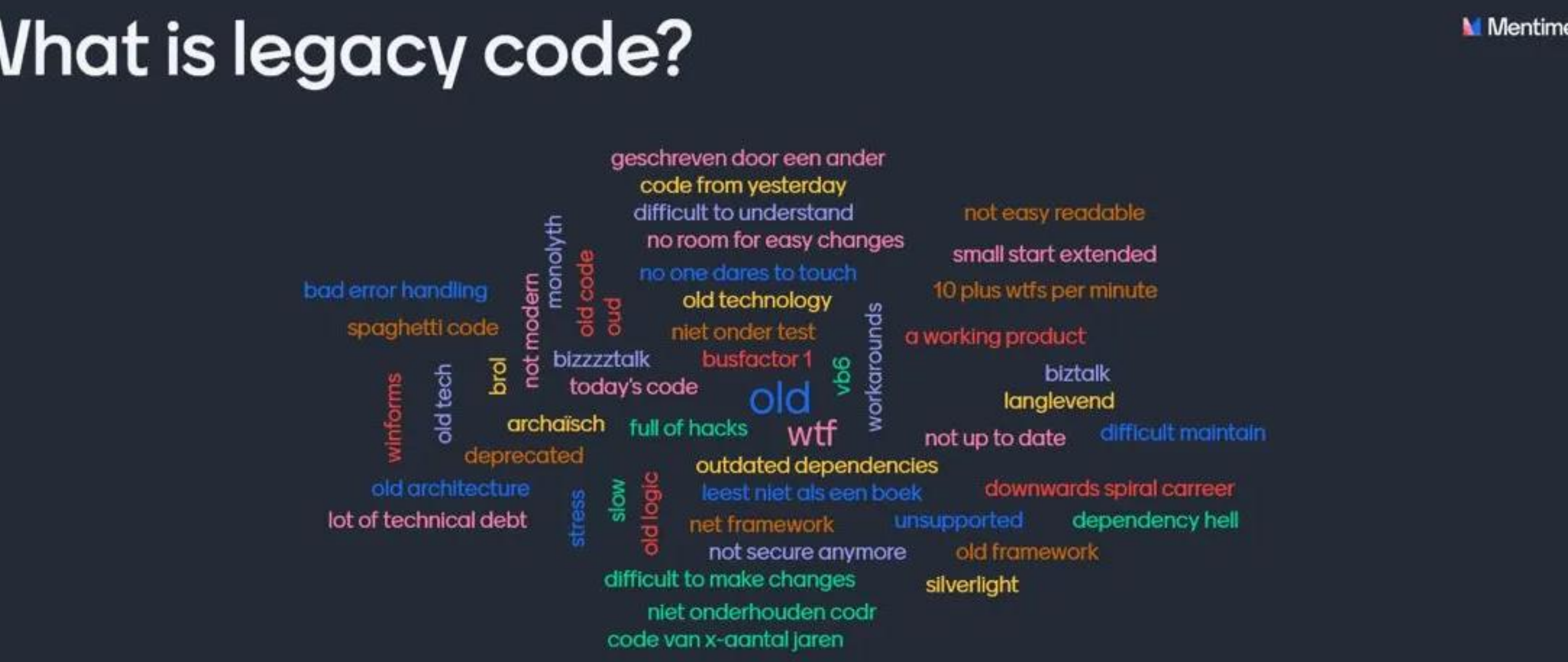
# Technical debt



# What is legacy code?



2008 Focus Shift/OSNews/Thom Holwerda - http://www.osnews.com/comi



# What is legacy code?

geschreven door een ander
code from yesterday
difficult to understand
not easy readable
no room for easy changes
monolyth
small start extended
old code
no one dares to touch
bad error handling
not modern
oud
old technology
10 plus wtfs per minute
spaghetti code
workarounds
niet onder test
a working product
bizzzztalk
busfactor 1
vb6
brol
biztalk
today's code
winforms
old tech
old
langlevend
archaïsch
full of hacks
wtf
not up to date
difficult maintain
deprecated
outdated dependencies
old architecture
slow
old logic
leest niet als een boek
downwards spiral carreer
stress
lot of technical debt
net framework
unsupported
dependency hell
not secure anymore
old framework
difficult to make changes
silverlight
niet onderhouden codr
code van x-aantal jaren

# What is legacy code?

Code that you’re afraid to change

- Breaks easily
- No idea what it does exactly
- **No tests** to tell you if you broke something

# Migratieproces

### Write tests

Unit tests are not always possible

Use high-level integration tests to get started:

- Test HTTP calls
- Test the database

### Refactor

We can quickly test thanks to automation

We are less likely to introduce bugs

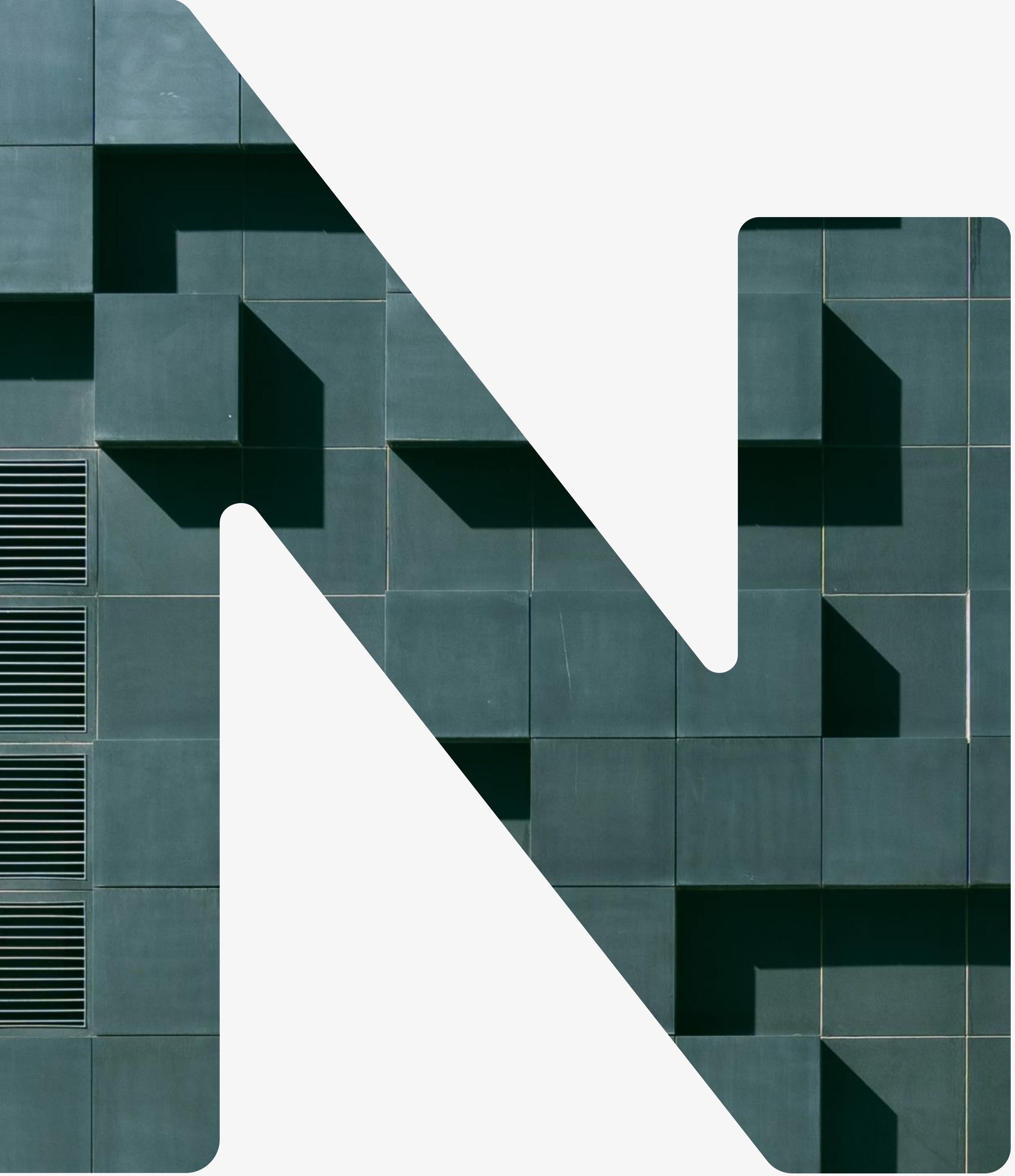
### Improve tests

Integration tests are slow and can be flaky

Unit tests are much better

**Write more tests!**

# What’s next?



# Modular Monolith

“DEFINITION”

# What is a Modular Monolith

- Think Microservices
- Single solution
- Multiple modules
  - That can be reused
- 1 module is on logical island
- Benefits of microservices without the overhead

FUNCTIONAL REQUIREMENTS

# Real Life example

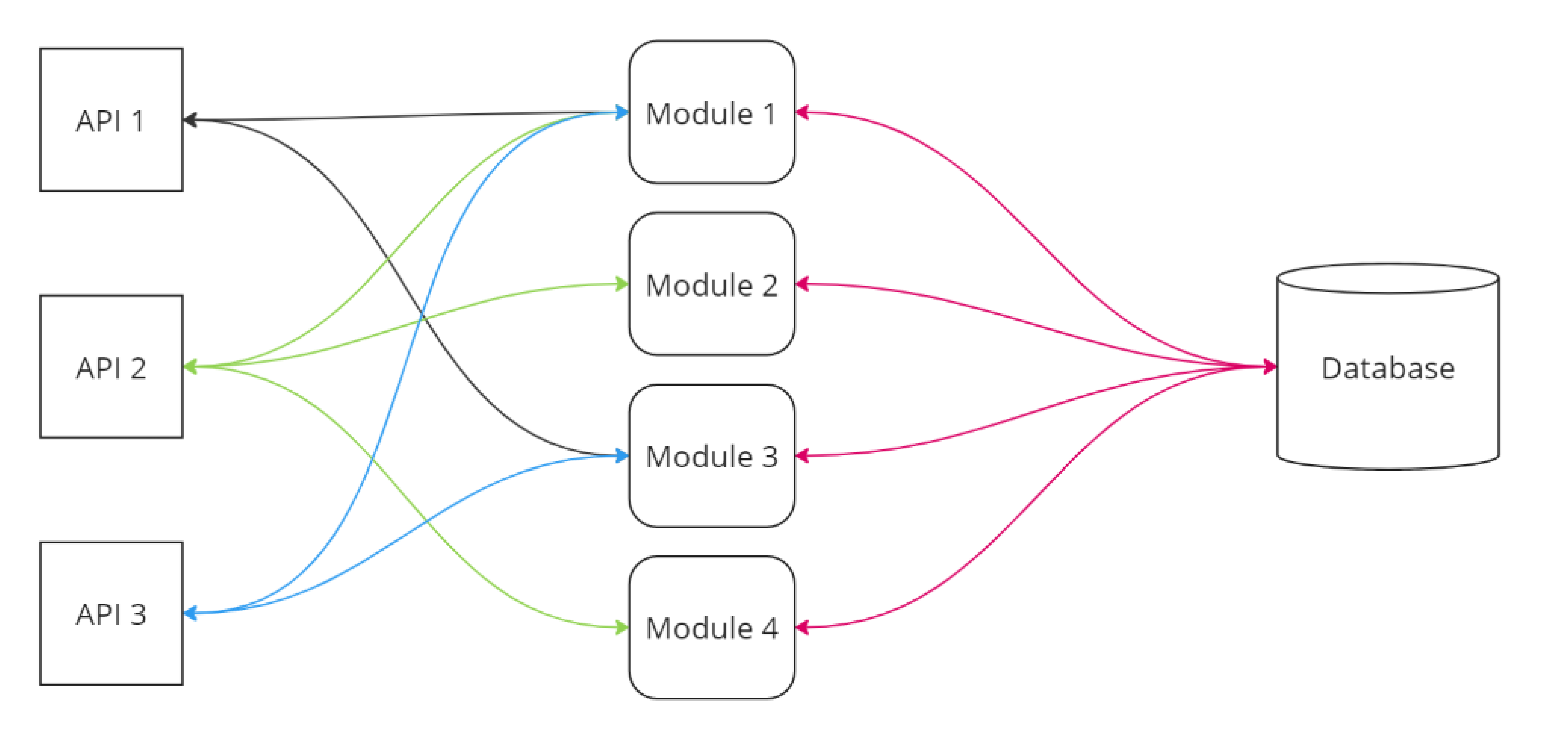
- Data platform
- Shared API’s
- Multiple API’s
- Security concerns

TECHNICAL SOLUTION

# Scalable solution

- Reusability
- Simple in setup
- Easy to use
- Multiple teams in one codebase

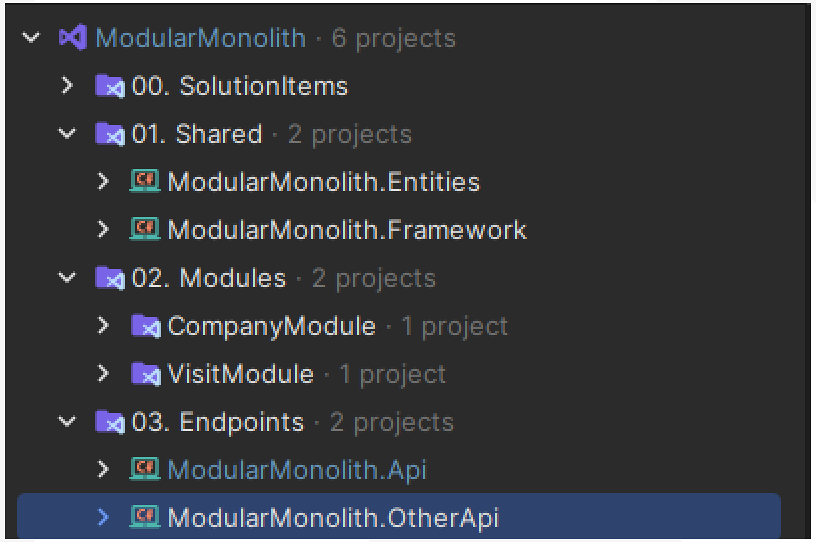
SCHEMATICS



SCHEMATICS

# Solution

- Shared entities
- Central framework (can be NuGet)
- Modules
- Endpoints

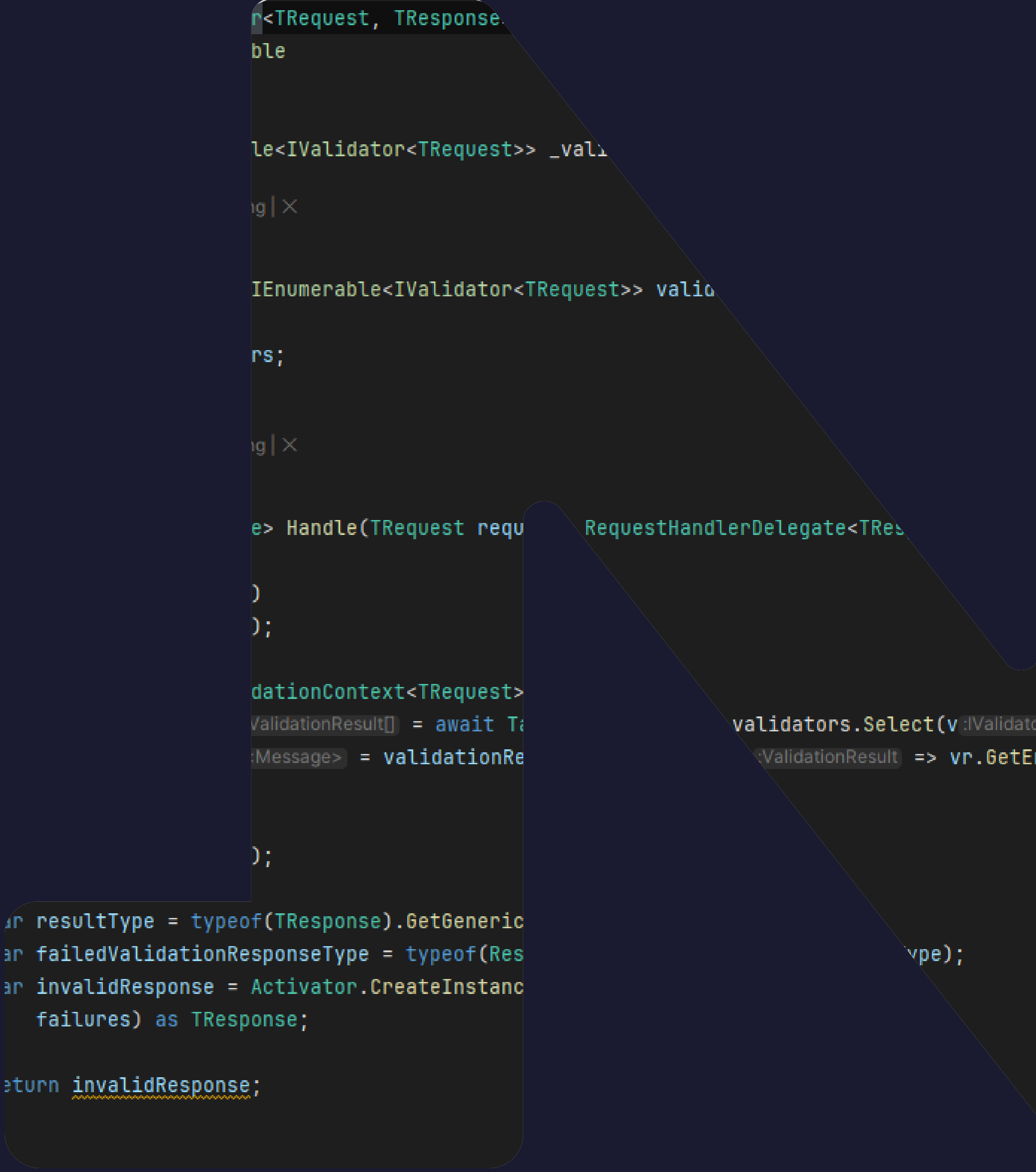


```
ModularMonolith · 6 projects
  00. SolutionItems
  01. Shared · 2 projects
    ModularMonolith.Entities
    ModularMonolith.Framework
  02. Modules · 2 projects
    CompanyModule · 1 project
    VisitModule · 1 project
  03. Endpoints · 2 projects
    ModularMonolith.Api
    ModularMonolith.OtherApi
```

NOEST

# And now CODE

**DEMO TIME**



```
r<TRequest, TResponse
ble

le<IValidator<TRequest>> _vali

ng|×

IEnumerable<IValidator<TRequest>> valid

rs;

ng|×

e> Handle(TRequest requ     RequestHandlerDelegate<TRes

)
);

dationContext<TRequest>
ValidationResult[] = await Ta                     validators.Select(v:IValidato
Message> = validationRe                           :ValidationResult => vr.GetEr

);

ar resultType = typeof(TResponse).GetGeneric
ar failedValidationResponseType = typeof(Res                                 ype);
ar invalidResponse = Activator.CreateInstanc
   failures) as TResponse;

eturn invalidResponse;
```

# Stiefwel merci!