Building Cloud Native applications with .NET Core

Johnny Hooyberghs

involved


involved
Cloud Native
The Cloud Native Computing Foundation

Cloud Native Technologies

Cloud native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds. Containers, service meshes, microservices, immutable infrastructure, and declarative APIs exemplify this approach.

These techniques enable loosely coupled systems that are resilient, manageable, and observable. Combined with robust automation, they allow engineers to make high-impact changes frequently and predictably with minimal toil.

The Cloud Native Computing Foundation seeks to drive adoption of this paradigm by fostering and sustaining an ecosystem of open source, vendor-neutral projects. We democratize state-of-the-art patterns to make these innovations accessible for everyone.


BORING

involved

Cloud Native Technologies

Cloud native is microservices hosted in containers and serverless apps, that can run in multi-cloud environments and are managed by DevOps processes

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

involved

Cloud Native Technologies

Cloud native is microservices hosted in containers and serverless apps, that can run in multi-cloud environments and are managed by DevOps processes

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

Microservices

A collection of loosely coupled services, where services are fine-grained and protocols are lightweight

involved

Containers

A standardized unit for storing, shipping and deploying a software package to run quickly and reliably, independently of computing environment

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

involved

Serverless

Thanks to cloud computing: scaling, capacity planning and maintenance can be hidden from the developer or operator

Focus on your application, not the infrastructure

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

involved

Multi-cloud

The use of multiple cloud computing and storage services in a single network architecture and the ability to be cloud-agnostic

involved

DevOps

A set of practices that combines software development and IT-operations, to shorten system development lifecycle and provides continues delivery

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

CSharpWars

Why use a simplified sample app when you can use a real app ☺


MADE A HELLO WORLD CONSOLE APPLICATION
NEXT BILL GATES

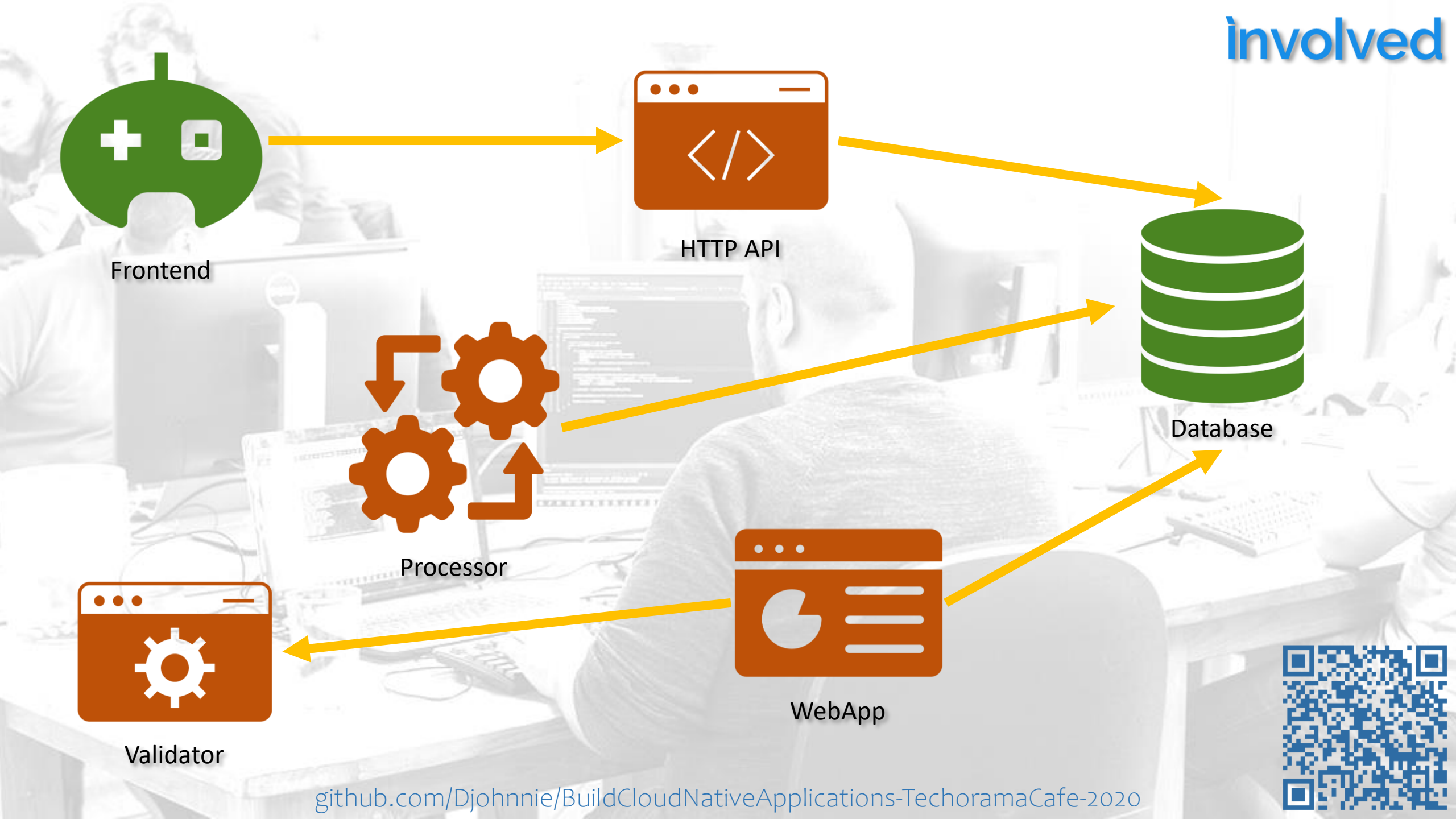

geert (geert)
killer (djohnnie)
geert (geert)

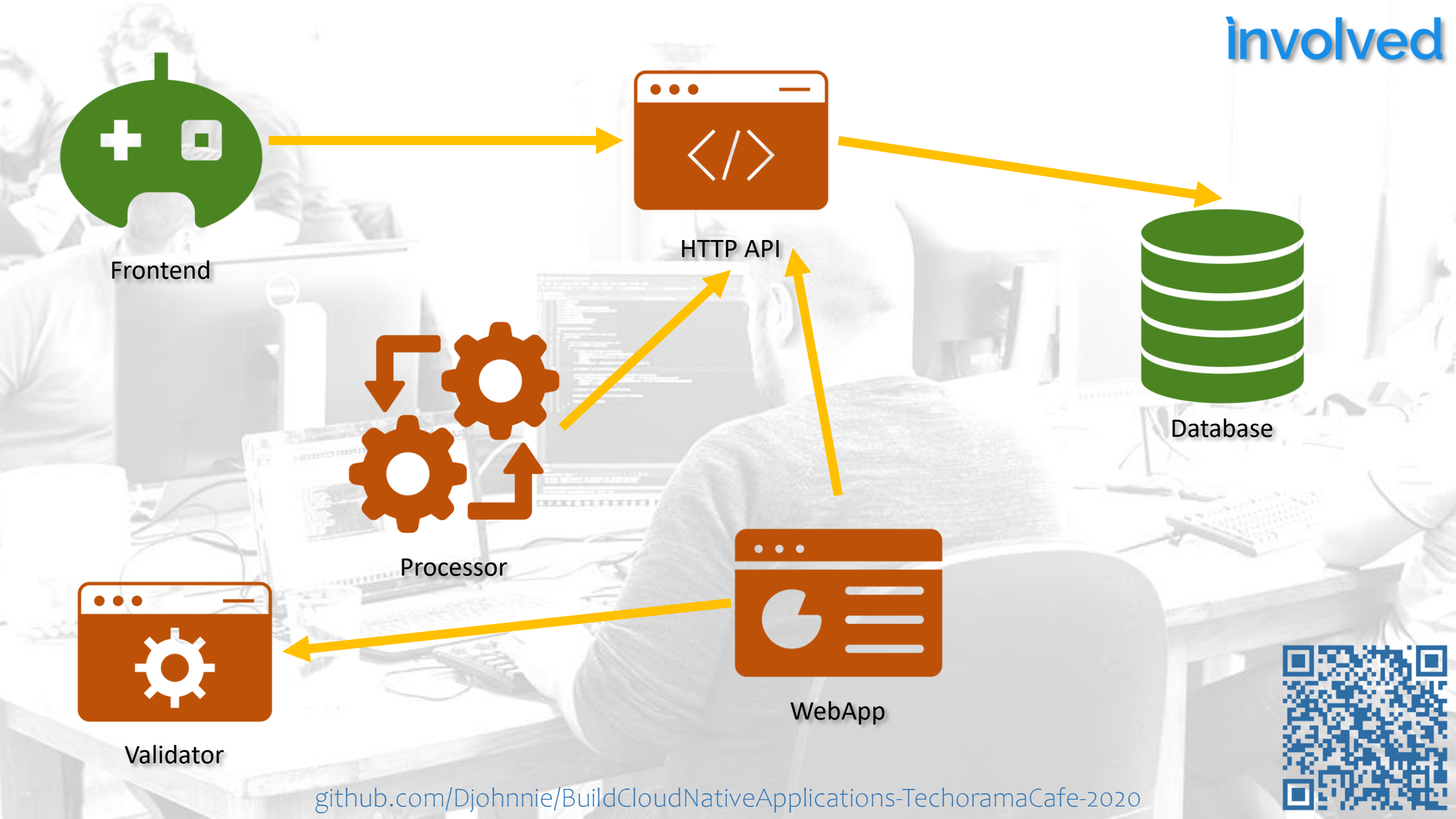
involved

CSharpWars Robot Scripting

```
var step = LoadFromMemory<Int32>("STEP");
if( step % 3 == 0 )
{
    TurnLeft();
}
else
{
    WalkForward();
}
step++;
StoreInMemory<Int32>("STEP", step);
```

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020


involved
Frontend
HTTP API
Database
Processor
Validator
WebApp
github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020


involved
Frontend
HTTP API
Database
Processor
WebApp
Validator
github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

.NET Core

What can .NET Core do to help with cloud native

involved

.NET Core vs. .NET Framework

- Platform independent
- High performance
- Lightweight
- Future-proof
- Cloud Native compatible
- The way to go for new apps

- Backwards compatible
- Restricted to Windows
- Better Windows-integration (*)
- Cloud Native compatible
- The way to go for legacy apps

(*) .NET Core will close this gap in the coming years from .NET 5 onwards

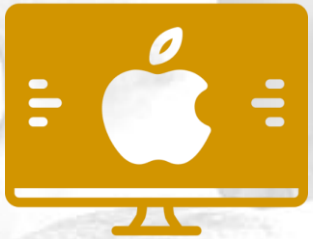
github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

involved

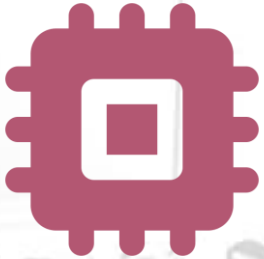
.NET Core is Platform Independent

Windows

Linux

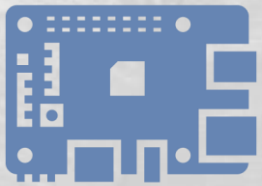


Mac



x86, x64, arm32, arm64

Cloud



IoT

Docker

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

involved

.NET Core SDK is not bound by tools

```
Welcome to .NET Core SDK CLI…
> dotnet new
> dotnet restore
> dotnet build
> dotnet publish
> dotnet test
> dotnet run
```

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

involved

Dependency Injection

```
[Route("[controller]")]
[ApiController]
public class ArenaController : ApiController<IArenaLogic>
{
    public ArenaController(IArenaLogic arenaLogic) : base(arenaLogic) { }

    // GET api/values
    [HttpGet]
    public Task<IActionResult> GetArena()
    {
        return Success(l => l.GetArena());
    }
}
```

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

involved

Configuration

```
public class ArenaLogic : IArenaLogic
{
    private readonly IConfiguration _configuration;

    public ArenaLogic(IConfiguration configuration) {
        _configuration = configuration;
    }

    public Task<ArenaDto> GetArena() {
        return Task.FromResult(new ArenaDto {
            Width = _configuration.GetValue<int>("ARENA_SIZE"),
            Height = _configuration.GetValue<int>("ARENA_SIZE")
        });
    }
}
```

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

involved

Logging

```
try {
    using var sw = new SimpleStopwatch();
    var middleware = scopedServiceProvider.GetService<IMiddleware>();
    await middleware.Process();
    _logger.LogInformation(
        "[ CSharpWars Script Processor - PROCESSING {ElapsedMilliseconds}ms! ]“
        , sw.ElapsedMilliseconds);
} catch (Exception ex) {
    _logger.LogError(ex,
        $"[ CSharpWars Script Processor - EXCEPTION - '{ex.Message}'! ]");
}
```

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

involved

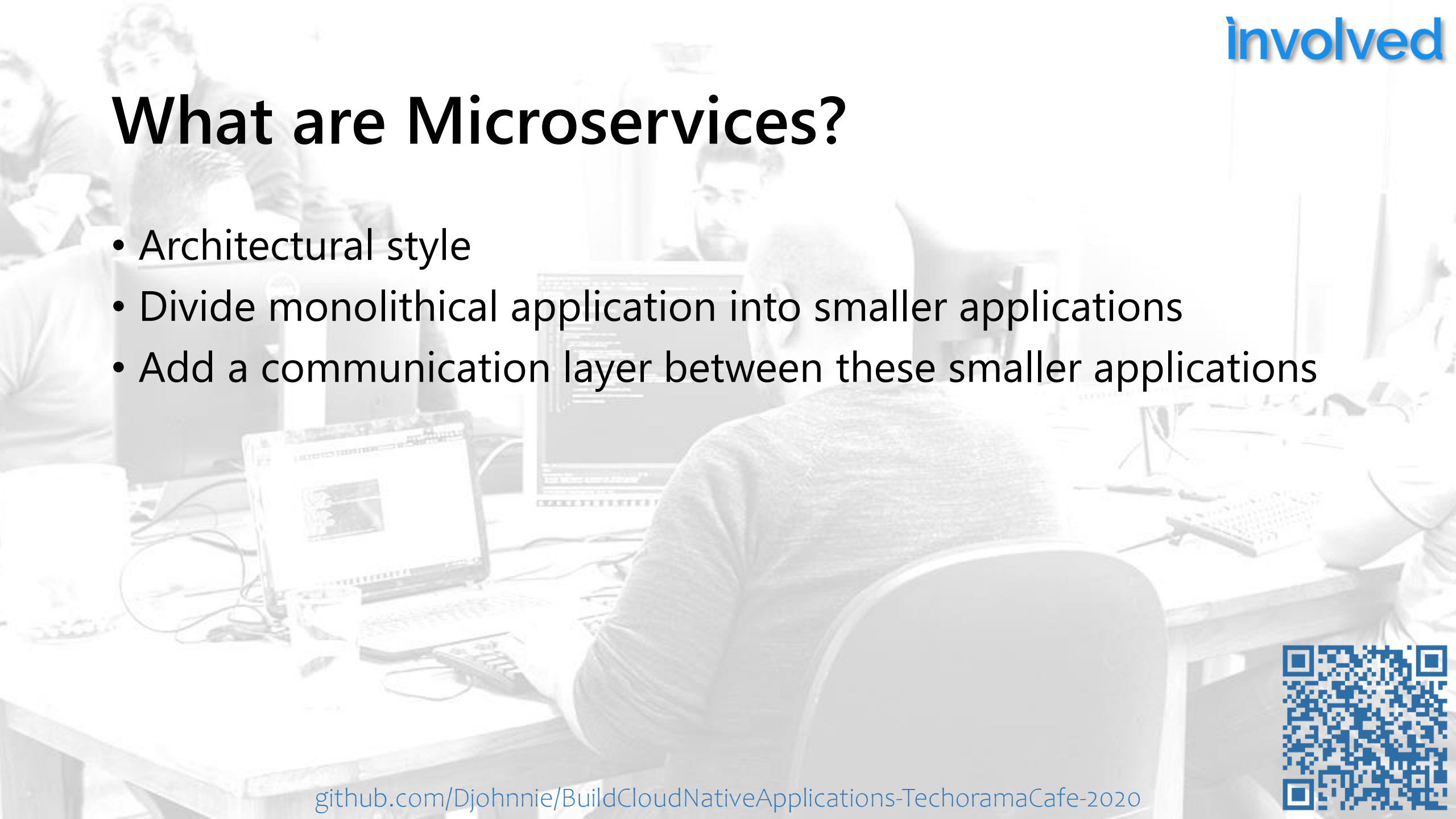
Microservices

A collection of loosely coupled services, where services are fine-grained and protocols are lightweight

involved

What are Microservices?

- Architectural style
- Divide monolithical application into smaller applications
- Add a communication layer between these smaller applications




SAY MICROSERVICE
ONE MORE TIME
memegenerator.net

involved

Why Microservices

- Increased performance
  - Easier to pinpoint a performance bottleneck in the system
  - Easier to scale out
- Increased manageability
  - Easier to upgrade part of the system in isolation
  - Easier to do feature-updates
- Increased velocity
  - Easier to scale out teams
- Increased flexibility
  - Easier to use different technologies
  - Easier to use different programming languages

involved

Should you use Microservices?

- Large applications
  - That (can) have clear defined boundaries
  - That should be scalable
- Large teams
  - Than can work on different parts of the application in parallel
  - To increase flexibility and velocity

YES

Should you use Microservices?

- Small or lightweight applications
- Distributed applications are hard
  - More chances of failing parts
  - Harder to work together as a team
  - More communication needed between different teams
- Difficult to define boundaries

NO


YOU CALLED MY SYSTEM A
MONOLITH

involved

.NET Core to develop Microservices

- ASP.NET Core WebApi for HTTP & JSON based communication
- ASP.NET Core gRPC for HTTP/2 & Binary based communication
- .NET Core Worker Services for background processing
- External messaging and pub/sub infrastructure & frameworks
  - Azure Service Bus
  - NServiceBus
  - Dapr
  - ...

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

involved

.NET Core to host Microservices

- ASP.NET Core has a built-in webserver called Kestrel
  - Runs on Windows, Linux and Mac
  - Runs on Raspberry Pi
  - ~~Runs on a potato~~
- Host in IIS
- Host in Azure App Service
- Host inside Docker container
- …

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

involved

ASP.NET Core WebApi (server)

```
[Route("[controller]")]
[ApiController]
public class ArenaController : ApiController<IArenaLogic>
{
    public ArenaController(IArenaLogic arenaLogic) : base(arenaLogic) { }

    // GET api/values
    [HttpGet]
    public Task<IActionResult> GetArena()
    {
        return Success(l => l.GetArena());
    }
}
```

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

involved

ASP.NET Core WebApi (client)

```
public static Arena GetArena()
{
    return Get<Arena>("arena");
}

private static TResult Get<TResult>(string resource) where Tresul
    : new()
{
    var client = new RestClient(_baseUrl);
    var request = new RestRequest(resource, Method.GET);
    var response = client.Execute<TResult>(request);
    return response.Data;
}
```

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

involved

ASP.NET Core gRPC (contract)

```
syntax = "proto3";
option csharp_namespace = "CSharpWars.Validator";
package Validator;

service ScriptValidator {
  rpc Validate (ScriptValidationRequest) returns (ScriptValidationResponse);
}

message ScriptValidationRequest {
  string Script = 1;
}
message ScriptValidationResponse {
    int64 CompilationTimeInMilliseconds = 1;
    int64 RunTimeInMilliseconds = 2;
    repeated ScriptValidationMessage ValidationMessages = 3;
}
```

```
message ScriptValidationMessage
{
    int32 LocationStart = 1;
    int32 LocationEnd = 2;
    string Message = 3;
}
```

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

involved

ASP.NET Core gRPC (server)

```
public class ScriptValidatorService
    : ScriptValidator.ScriptValidatorBase {

    private readonly IScriptValidationHelper _helper;

    public ScriptValidatorService(IScriptValidationHelper helper) {
        _helper = helper;
    }

    public override Task<ScriptValidationResponse> Validate(
        ScriptValidationRequest request, ServerCallContext context) {
            return _helper.Validate(request);
    }
}
```

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

involved

ASP.NET Core gRPC (client)

```
public async Task<ValidatedScriptDto> Validate(ScriptToValidateDto script)
{
    var request = new ScriptValidationRequest { Script = script.Script };
    var channel = GrpcChannel.ForAddress(_configuration.ValidationHost);
    var client = new ScriptValidator.ScriptValidatorClient(channel);
    var response = await client.ValidateAsync(request);

    return new ValidatedScriptDto
    {
        Script = script.Script,
        CompilationTimeInMilliseconds = response.CompilationTimeInMilliseconds,
        RunTimeInMilliseconds = response.RunTimeInMilliseconds,
    };
}
```

involved

Worker Services

```
public class Worker : BackgroundService {

    private readonly IMiddleware _middleware;
    private readonly ILogger<Worker> _logger;

    public Worker(IMiddleware middleware, ILogger<Worker> logger) {
        _scopeFactory = scopeFactory;
        _logger = logger;
    }

    protected override async Task ExecuteAsync(CancellationToken stoppingToken) {
        while (!stoppingToken.IsCancellationRequested) {
            await _middleware.Process();
            _logger.LogInformation($"[ PROCESSING! ]");
        }
    }
}
```

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

involved

Containers

A standardized unit for developing, shipping and deploying a software package to run quickly and reliably, independently of computing environment

involved

What are containers?

- OS-level virtualization
- Software packages
- Includes dependencies, libraries and configuration
- Isolated from one another
- Communication via well defined channels
- More lightweight than Virtual Machines
  - Single operating system kernel, multiple containers
- Resource limiting


BONAVENTURE
KENDAL


tex

What about application state?

- Containers should not hold state!
- Use environment variables for configuration
- Use container volume mapping if needed
- Use external caching services like Redis
- Use external storage services like databases

involved

Building containers

```
FROM mcr.microsoft.com/dotnet/core/aspnet:3.1
WORKDIR /app
COPY bin/Release/publish /app
EXPOSE 5000
ENV TZ=Europe/Brussels
ENV KEY_VAULT=...
ENV CLIENT_ID=...
ENV CLIENT_SECRET=...
ENV CERTIFICATE_KEY=...
ENV ARENA_SIZE=10
ENTRYPOINT ["dotnet", "CSharpWars.Web.Api.dll"]
```

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

involved

.NET Core and Docker

https://hub.docker.com/_/microsoft-dotnet-core-runtime/

https://mcrflowprodcentralus.cdn.mscr.io/mcrprod/dotnet/core/runtime?P1=1583833908&P2=1&P3=1&P4=zHgCwD%2FlU5XzXxBZ7vY6escQfYTeKS5tKYjMcCODqQw%3D&se=2020-03-10T09%3A51%3A48Z&sig=5Mve%2FkEj3iq8Wkb7DQyr%2BwDiqUeFLDZ4eO8bItngvJY%3D&sp=r&sr=b&sv=2015-02-21

involved

Serverless

Thanks to cloud computing, scaling, capacity planning and maintenance can be hidden from the developer or operator

Focus on your application, not the infrastructure

involved

What is Serverless?

- The cloud provider is responsible to execute your piece of code
- Resources can be allocated dynamically
- You are charged for the resources you need (have consumed)
- Run as stateless containers
- Triggered by a variety of events (http, queueing, jobs, …)
- Latency due to cold starts


SERVERLESS
IS MADE OF SERVERS

involved

Multi-cloud

The use of multiple cloud computing and storage services in a single network architecture and the ability to be cloud-agnostic

involved

Why Multi-cloud?

- Build apps that work across multiple providers
- Avoid vendor lock-in
- Each provider has strenghts and weaknesses
- A level of resiliency that is not available on a single provider


CLOUD COMPUTING
CLOUD COMPUTING EVERYWHERE
memegenerator.net

Configuration

```
public static IHostBuilder CreateHostBuilder(string[] args) =>
  Host.CreateDefaultBuilder(args)
    .ConfigureWebHostDefaults(webBuilder => {
      webBuilder.ConfigureAppConfiguration(configBuilder => {
        var keyVault = GetEnvironmentVariable("KEY_VAULT");
        var clientId = GetEnvironmentVariable("CLIENT_ID");
        var clientSecret = GetEnvironmentVariable("CLIENT_SECRET");
        configBuilder.AddAzureKeyVault(keyVault, clientId, clientSecret);
      });
      webBuilder.ConfigureKestrel((ctx, options) => {
        var key = GetEnvironmentVariable("CERTIFICATE_KEY");
        var data = ctx.Configuration.GetValue<string>(key);
        var certificate = new X509Certificate2(Convert.FromBase64String(data));
        options.Listen(IPAddress.Any, 5000, listenOptions => {
          listenOptions.UseHttps(certificate); });
        }
      });
      webBuilder.UseStartup<Startup>();
});
```

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

involved

Logging

```
public static IHostBuilder CreateHostBuilder(string[] args) =>
  Host.CreateDefaultBuilder(args)
      ConfigureLogging((hostContext, logging) => {
        var elasticUri = hostContext.Configuration.GetValue<string>("elastic-uri");
        if (!string.IsNullOrEmpty(elasticUri)) {
          Log.Logger = new LoggerConfiguration()
              .Enrich.FromLogContext()
              .Enrich.WithExceptionDetails()
              .WriteTo.Elasticsearch(new ElasticsearchSinkOptions(new Uri(elasticUri))
{
                 AutoRegisterTemplate = true
              }).CreateLogger();
              logging.AddSerilog();
        }
      });
});
```

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020


involved
DevOps
A set of practices that combines software development and IT-operations
to shorten systems development lifecycle and provides continues delivery

involved

What is DevOps

- Working together
- Automation (with tools)
  - Building
  - Testing
  - Deploying
  - Updating and upgrading
  - Scaling
  - Monitoring
- Scripting (with tools)
  - Configuration as code
  - Source control!

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020


WORKED FINE IN DEV
OPS PROBLEM NOW
memegenerator.net

involved

Azure DevOps Pipelines

CloudNative-CSharpWars-Web
Run Docker Image

CloudNative-CSharpWars-Processor
Run Docker Image

CloudNative-CSharpWars-Validator
Run Docker Image

CloudNative-CSharpWars-Api
Run Docker Image

Recently run pipelines

Pipeline	Last run		
CloudNative-CSharpWars-Processor	#20200309.5 • Refactored pipelines to four separate builds Manually triggered master		
CloudNative-CSharpWars-Validator	#20200309.2 • Refactored pipelines to four separate builds Manually triggered master	3h ago 3m 39s	
CloudNative-CSharpWars-Web	#20200309.3 • Refactored pipelines to four separate builds Manually triggered master	4h ago 3m 50s	
CloudNative-CSharpWars-Api	#20200309.3 • Refactored pipelines to four separate builds Manually triggered master	4h ago 4m 6s	

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

Johnny Hooyberghs

@djohnnieke

www.involved-it.be

github.com/Djohnnie/BuildCloudNativeApplications-TechoramaCafe-2020

ìnvolved