Т БАНК

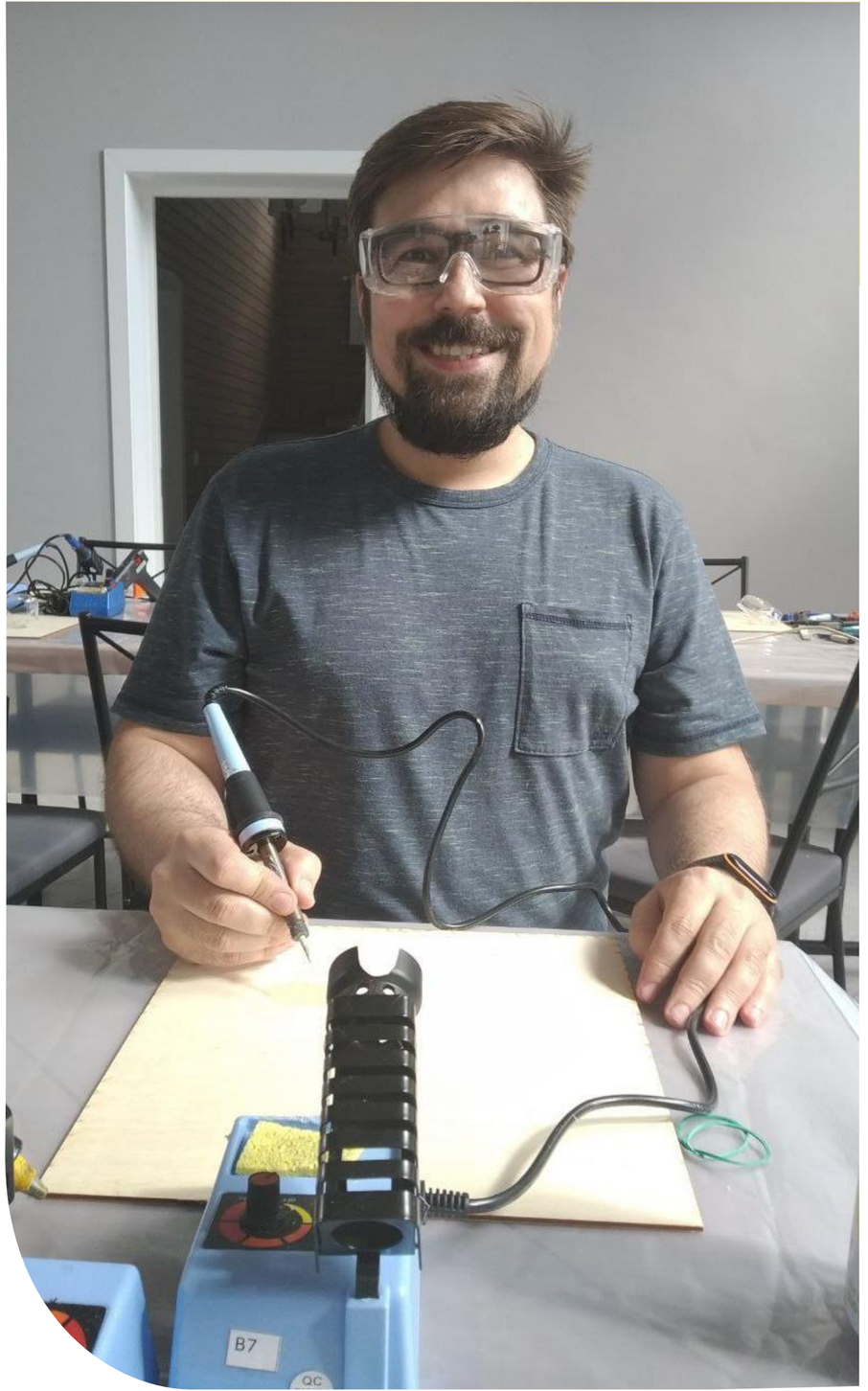
.NET для чайников

Александр Бусыгин

11.09.2024

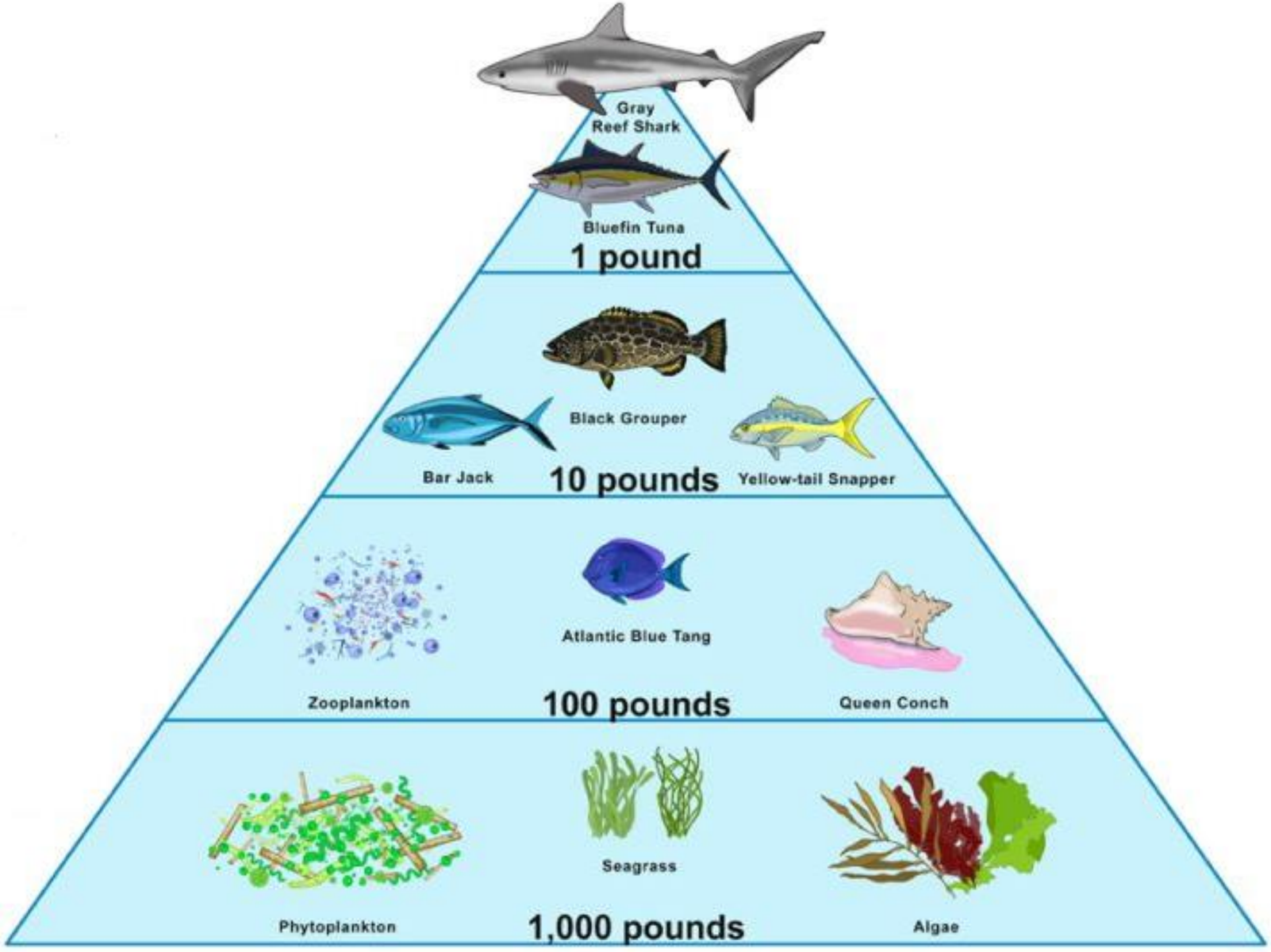
Александр Бусыгин

.NET разработчик

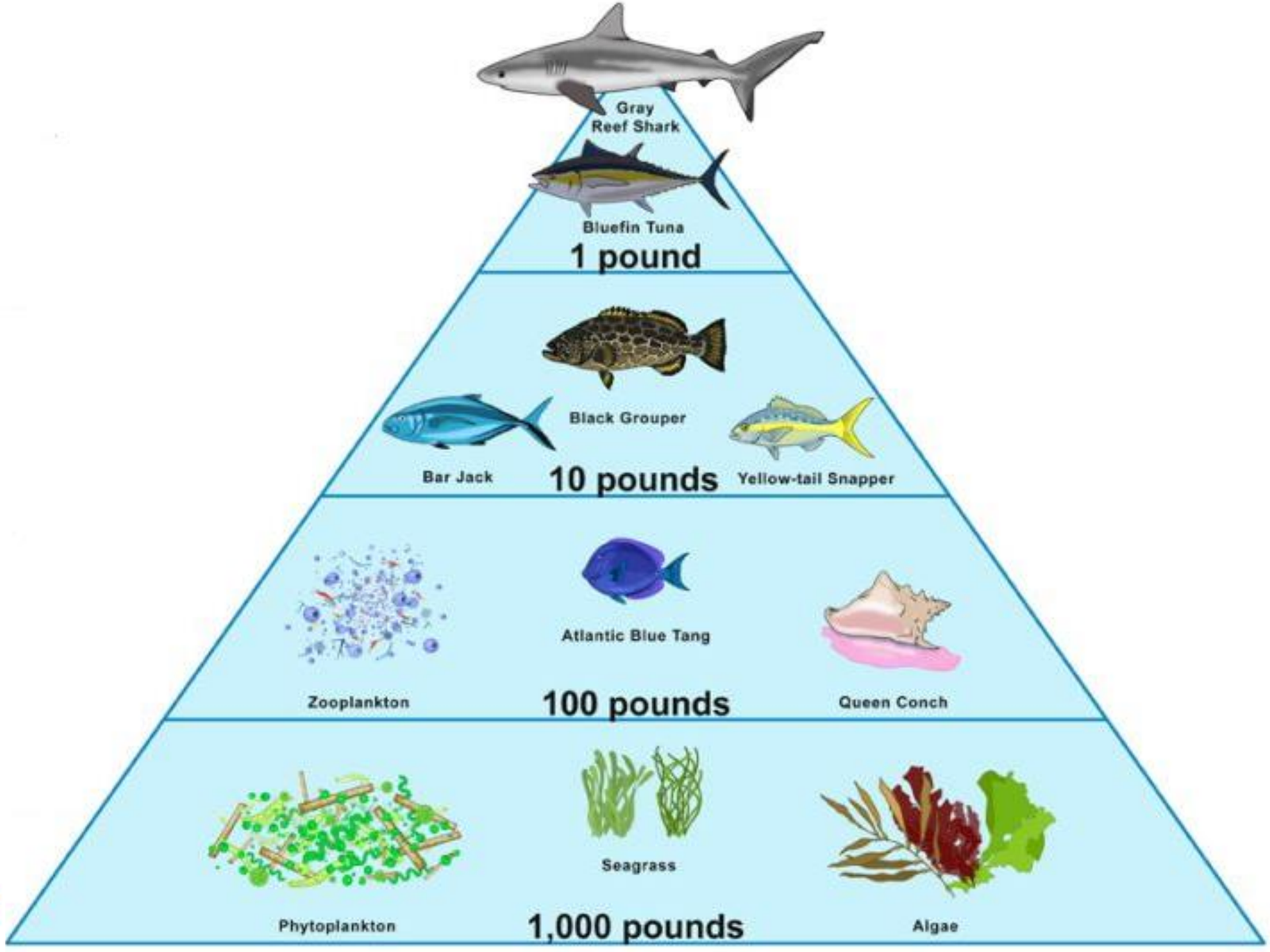


@AlexanderBusygin

“Экологическая пирамида”



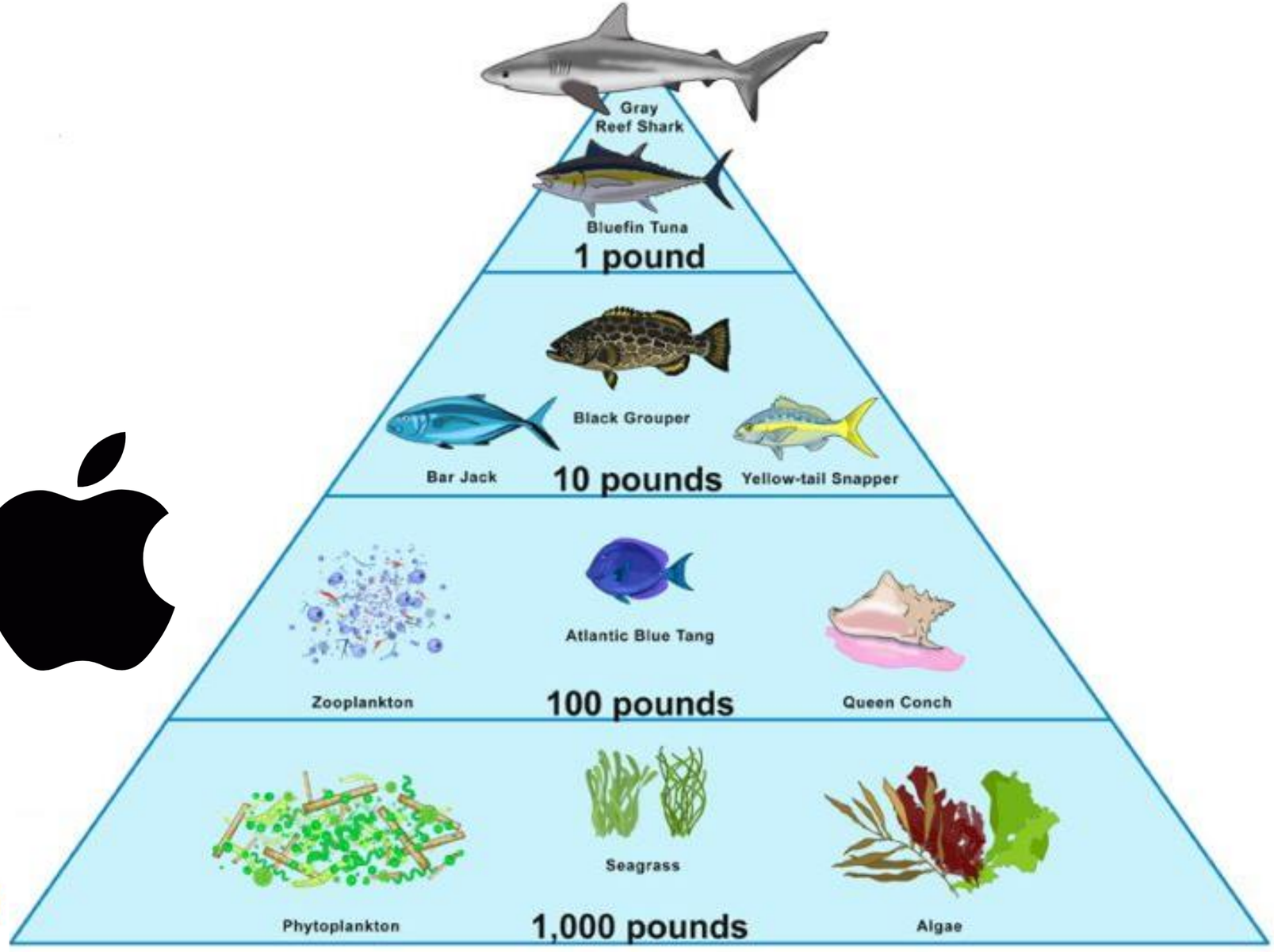
“Экологическая пирамида”

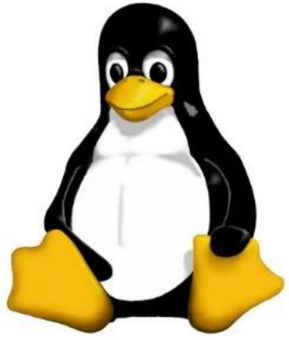


Microsoft Windows xp

Microsoft .net

“Экологическая пирамида”



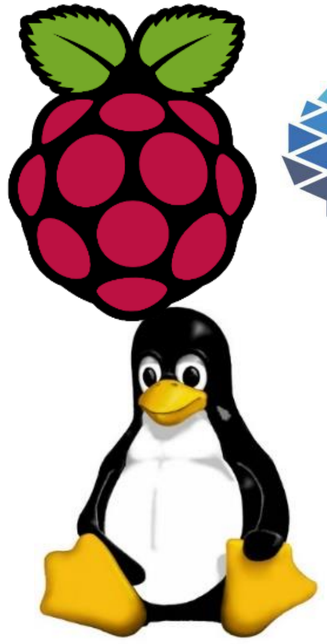


Microsoft
Windows xp

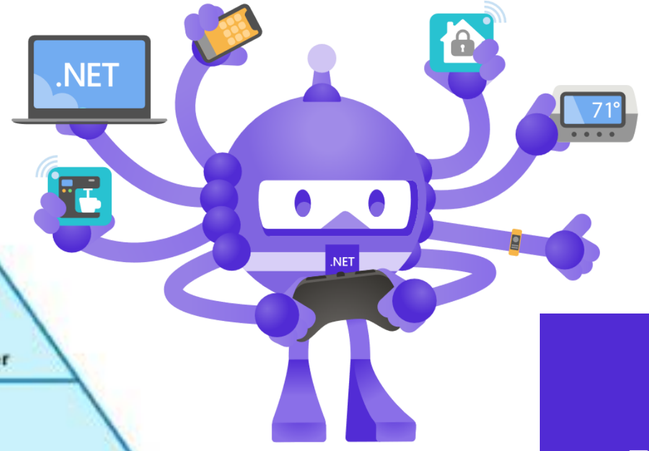
.NET

Microsoft
.net

“Экологическая пирамида”



PINE64



.NET

Microsoft .net

Microsoft Windows xp

“Экологическая пирамида”

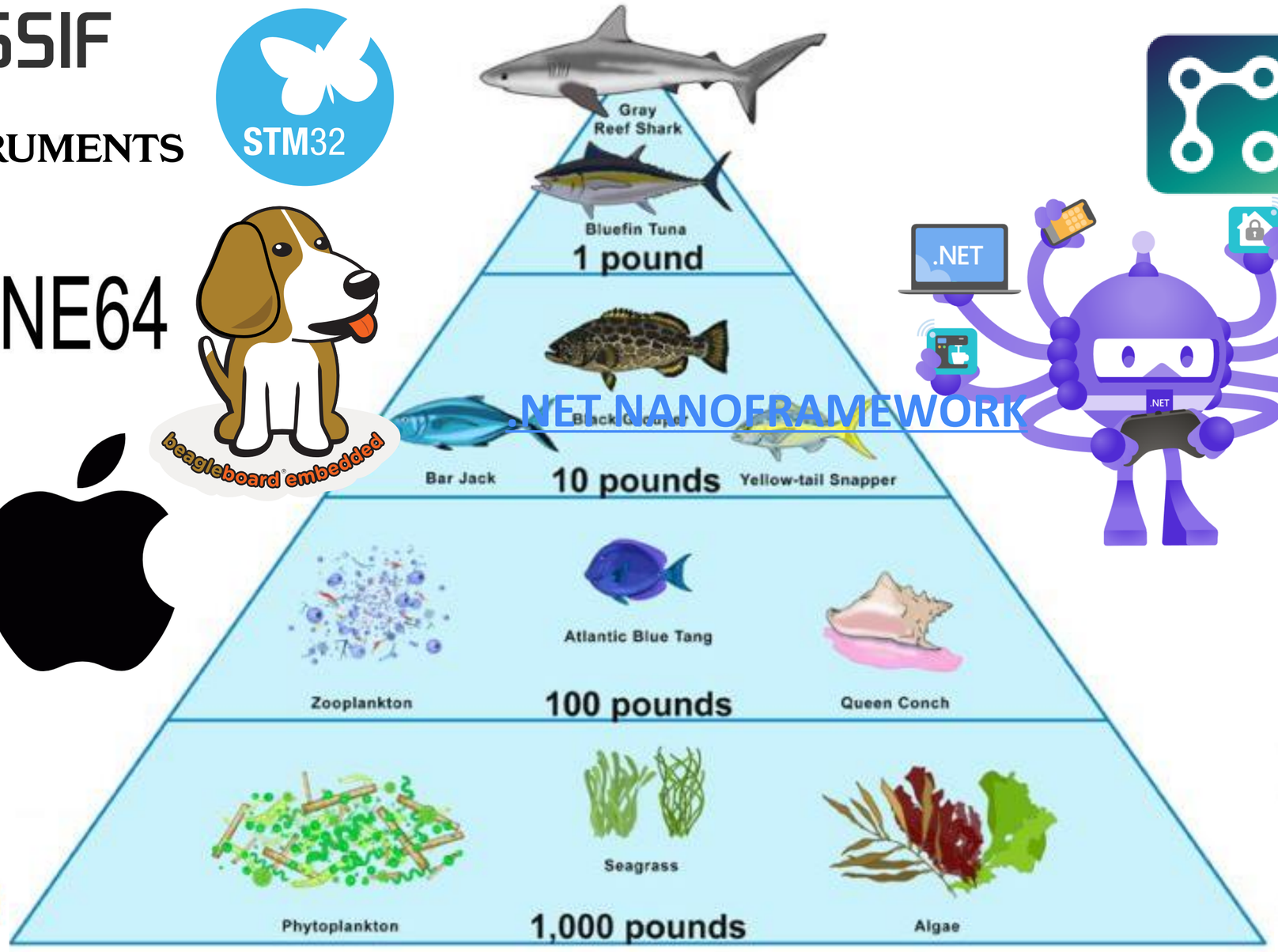
ESPRESSIF

TEXAS INSTRUMENTS

STM32

PINE64

beagleboard embedded



nano FRAMEWORK

.NET NANOFRAMEWORK

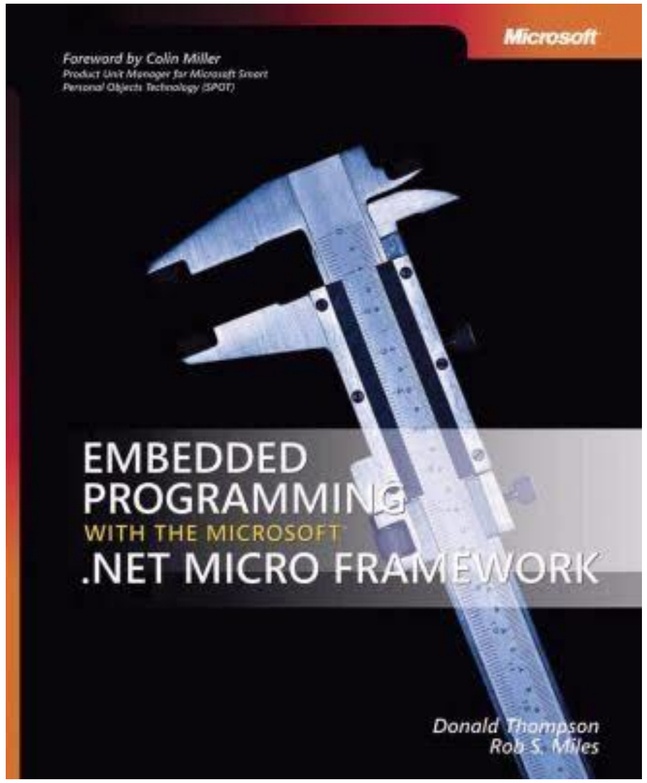
.NET

Microsoft .net

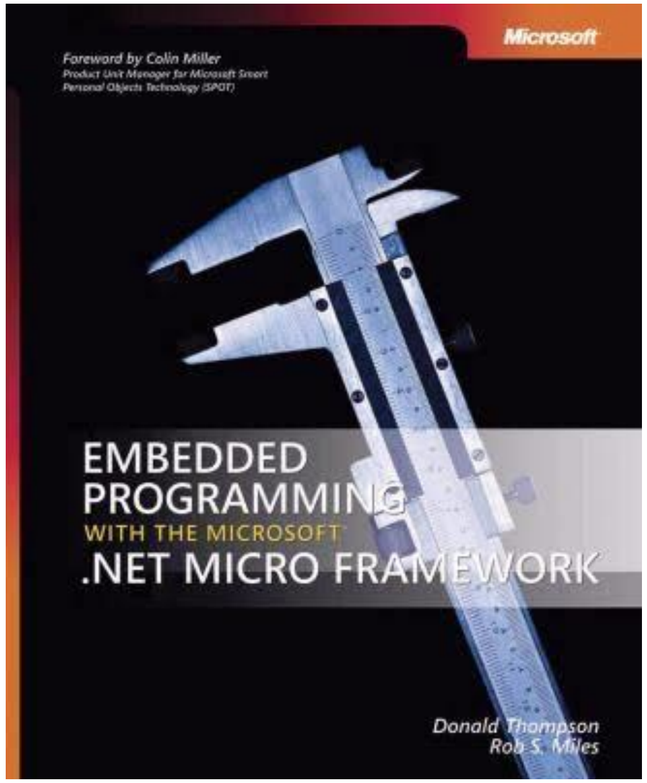
Microsoft Windows xp

.NET NANOFRAMEWORK

.NET NANOFRAMEWORK

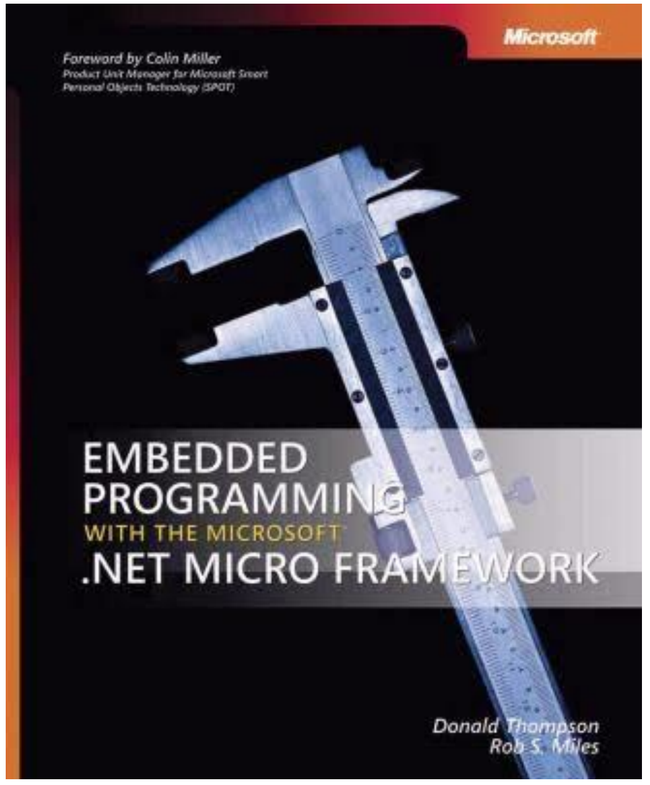


.NET NANOFRAMEWORK



.NET NANOFRAMEWORK

.NET micro framework



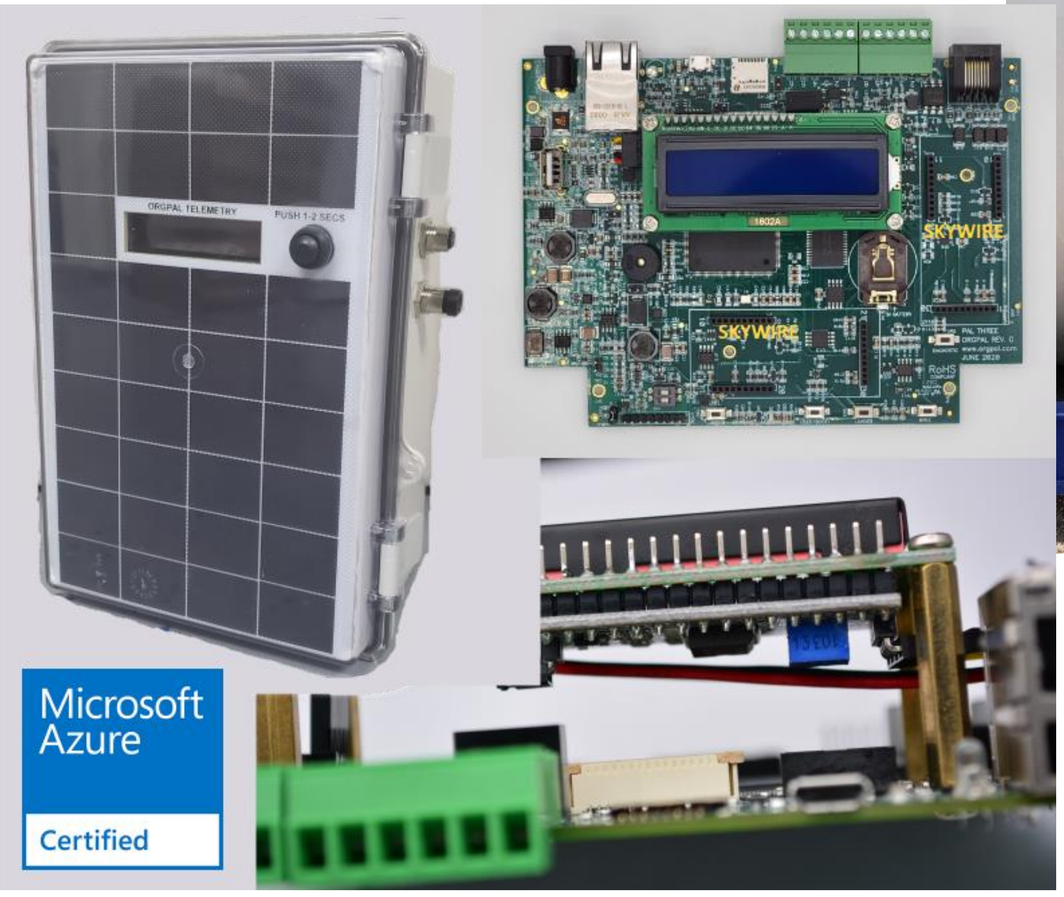
.NET Core

nano FRAMEWORK

.NET NANOFRAMEWORK

OrgPal.IoT

smart automation solutions



.NET NANOFRAMEWORK

NESHTEC
NEW ENERGY FROM NORTHERN GERMANY

PV solar power plant

Neshtec-Control

Neshtec-Control uses .NET nanoFramework on STM32 controller

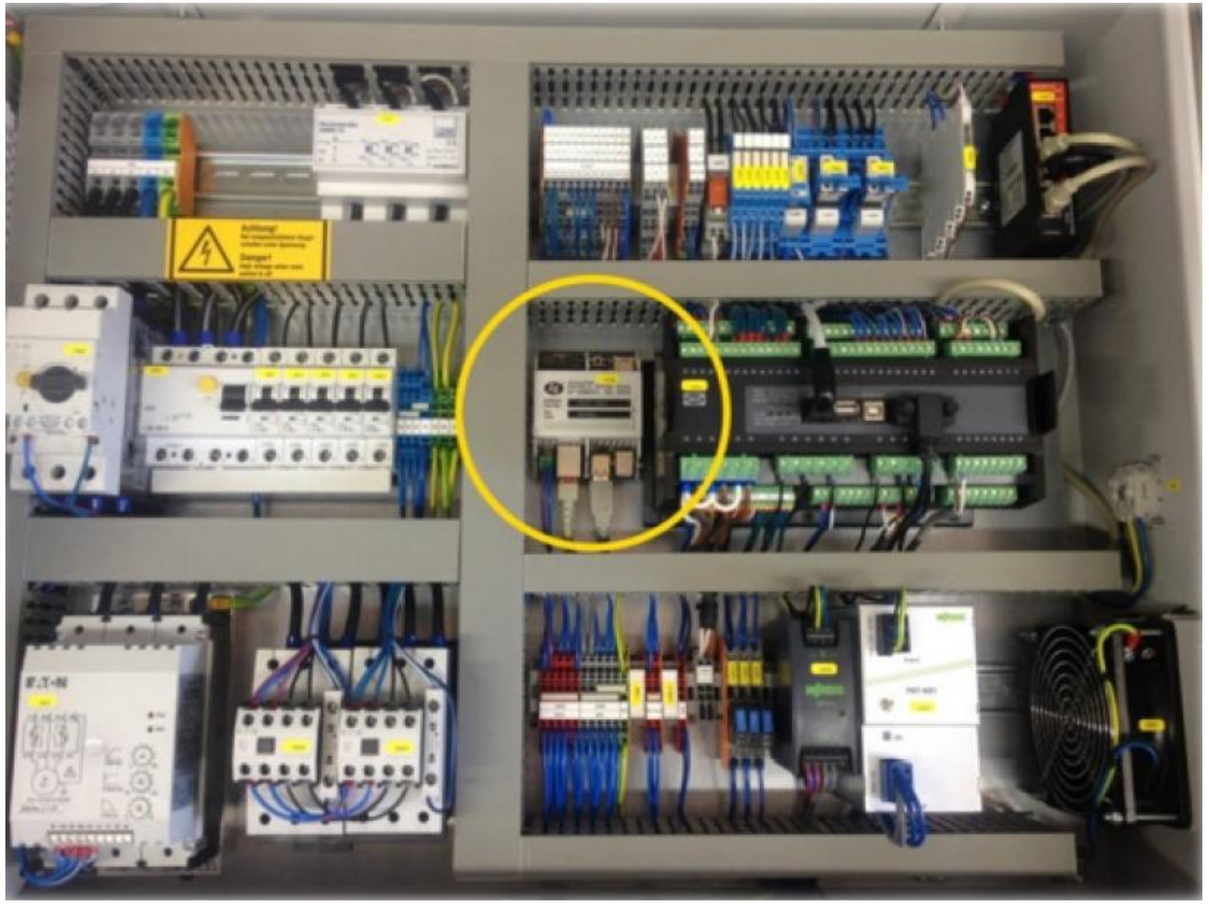
Engergy Supplier

Neshtec-Monitoring & Control

Azure IoT Hub

Web App

.NET NANOFRAMEWORK



.NET NANOFRAMEWORK

- 256kB of flash and 64kB of RAM.

.NET NANOFRAMEWORK

- 256kB of flash and 64kB of RAM.
- GPIO, UART, SPI, I2C, USB

.NET NANOFRAMEWORK

- 256kB of flash and 64kB of RAM.
- GPIO, UART, SPI, I2C, USB
- Multithreading

.NET NANOFRAMEWORK

- 256kB of flash and 64kB of RAM.
- GPIO, UART, SPI, I2C, USB
- Multithreading
- Deep sleep

.NET NANOFRAMEWORK

- 256kB of flash and 64kB of RAM.
- GPIO, UART, SPI, I2C, USB
- Multithreading
- Deep sleep
- Managed (C#) & native (C/C++) code

.NET NANOFRAMEWORK

- 256kB of flash and 64kB of RAM.
- GPIO, UART, SPI, I2C, USB
- Multithreading
- Deep sleep
- Managed (C#) & native (C/C++) code
- IDE. Visual Studio, Visual Studio Code

.NET NANOFRAMEWORK

- 256kB of flash and 64kB of RAM.
- GPIO, UART, SPI, I2C, USB
- Multithreading
- Deep sleep
- Managed (C#) & native (C/C++) code
- IDE. Visual Studio, Visual Studio Code
- Debugger

.NET NANOFRAMEWORK

- 256kB of flash and 64kB of RAM.
- GPIO, UART, SPI, I2C, USB
- Multithreading
- Deep sleep
- Managed (C#) & native (C/C++) code
- IDE. Visual Studio, Visual Studio Code
- Debugger
- Garbage collector

.NET NANOFRAMEWORK

- 256kB of flash and 64kB of RAM.
- GPIO, UART, SPI, I2C, USB
- Multithreading
- Deep sleep
- Managed (C#) & native (C/C++) code
- IDE. Visual Studio, Visual Studio Code
- Debugger
- Garbage collector
- DI

.NET NANOFRAMEWORK

- 256kB of flash and 64kB of RAM.
- GPIO, UART, SPI, I2C, USB
- Multithreading
- Deep sleep
- Managed (C#) & native (C/C++) code
- IDE. Visual Studio, Visual Studio Code
- Debugger
- Garbage collector
- DI
- Unit tests

.NET NANOFRAMEWORK

- 256kB of flash and 64kB of RAM.
- GPIO, UART, SPI, I2C, USB
- Multithreading
- Deep sleep
- Managed (C#) & native (C/C++) code
- IDE. Visual Studio, Visual Studio Code
- Debugger
- Garbage collector
- DI
- Unit tests
- Free & Open source

.NET NANOFRAMEWORK

Generics

.NET NANOFRAMEWORK

Generics

Ограниченная поддержка enum

.NET NANOFRAMEWORK

Generics

Ограниченная поддержка enum

Многомерные массивы

.NET NANOFRAMEWORK

Generics

Ограниченная поддержка enum

Многомерные массивы

Фиксированные форматы String.Format

и numeric.ToString()

.NET NANOFRAMEWORK

Generics

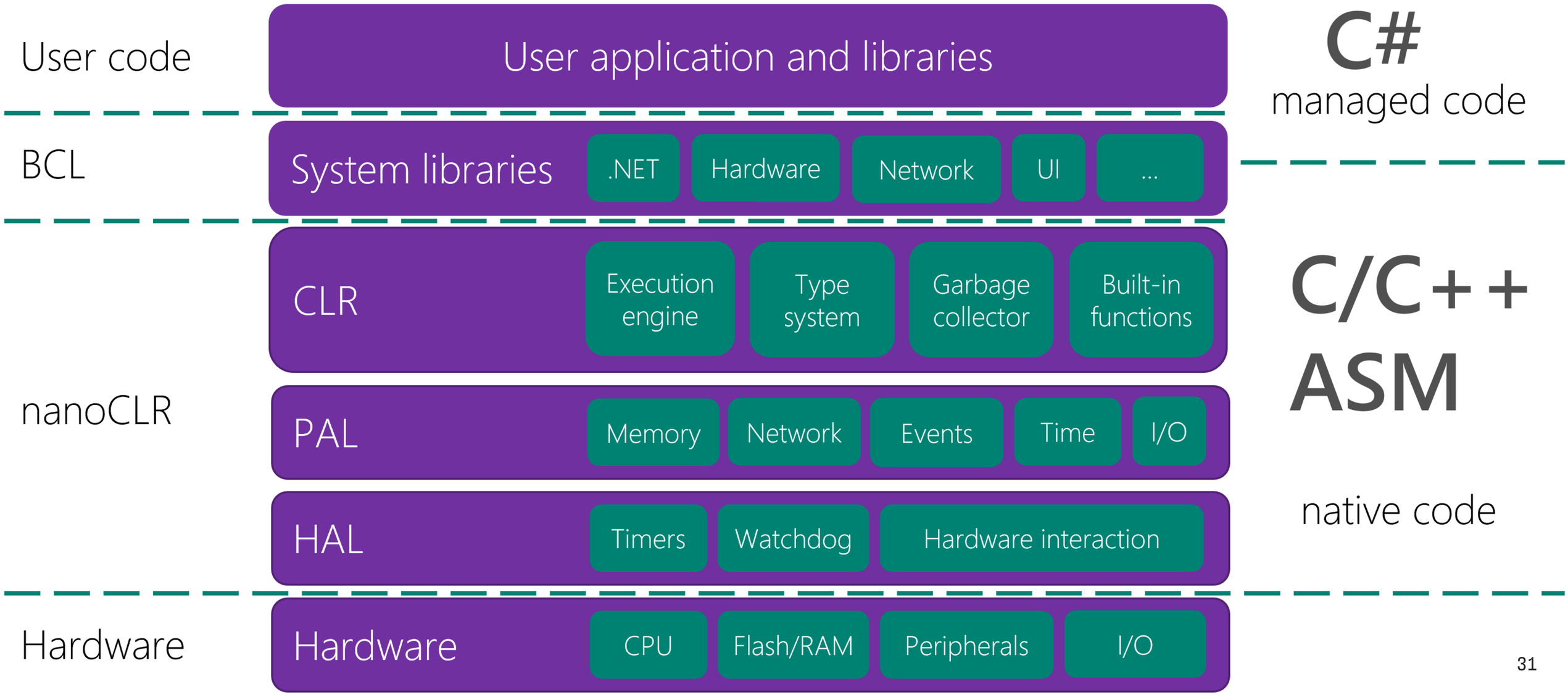
Ограниченная поддержка enum

Многомерные массивы

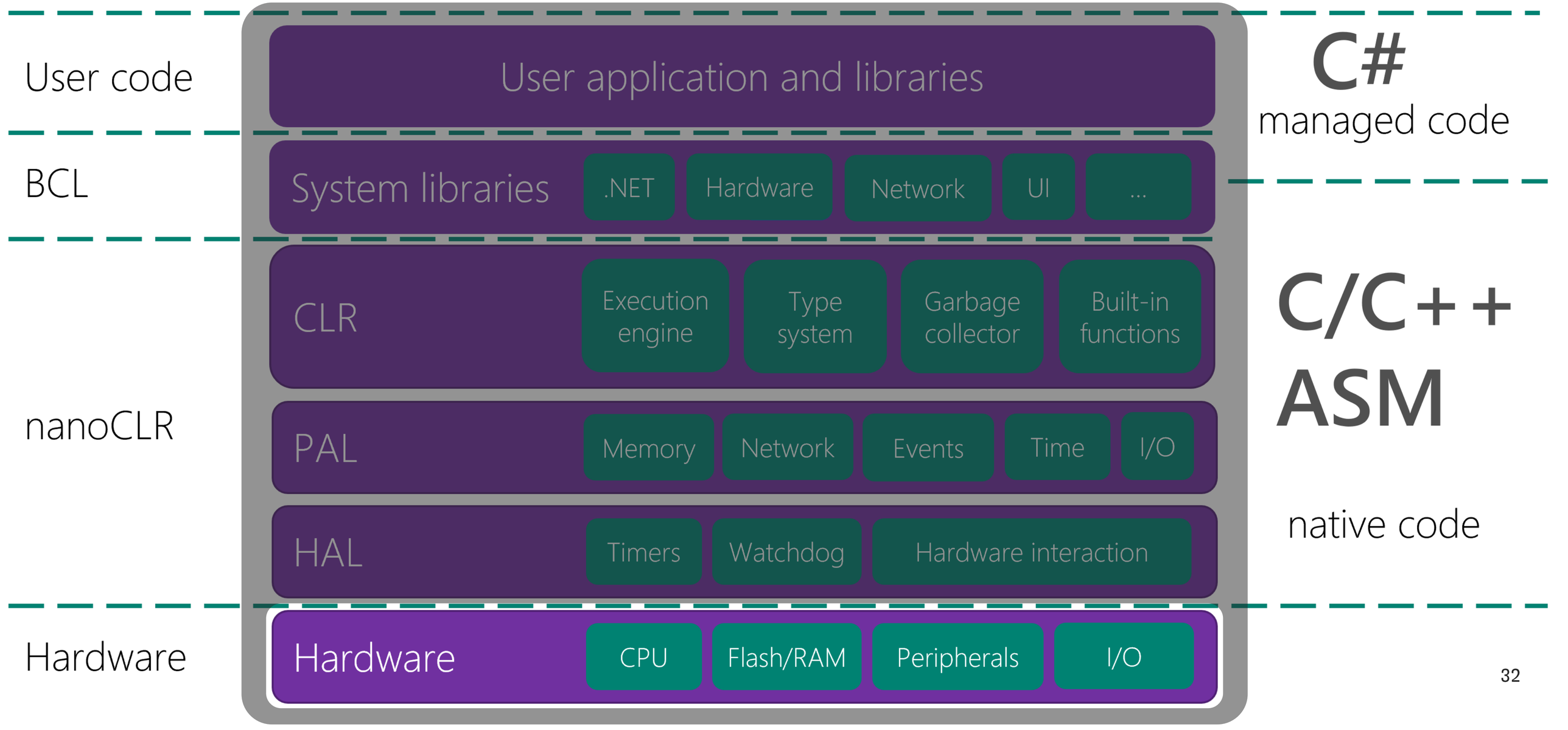
Фиксированные форматы String.Format и numeric.ToString()

По умолчанию double 4 байта. Флаг DP_FLOATINGPOINT

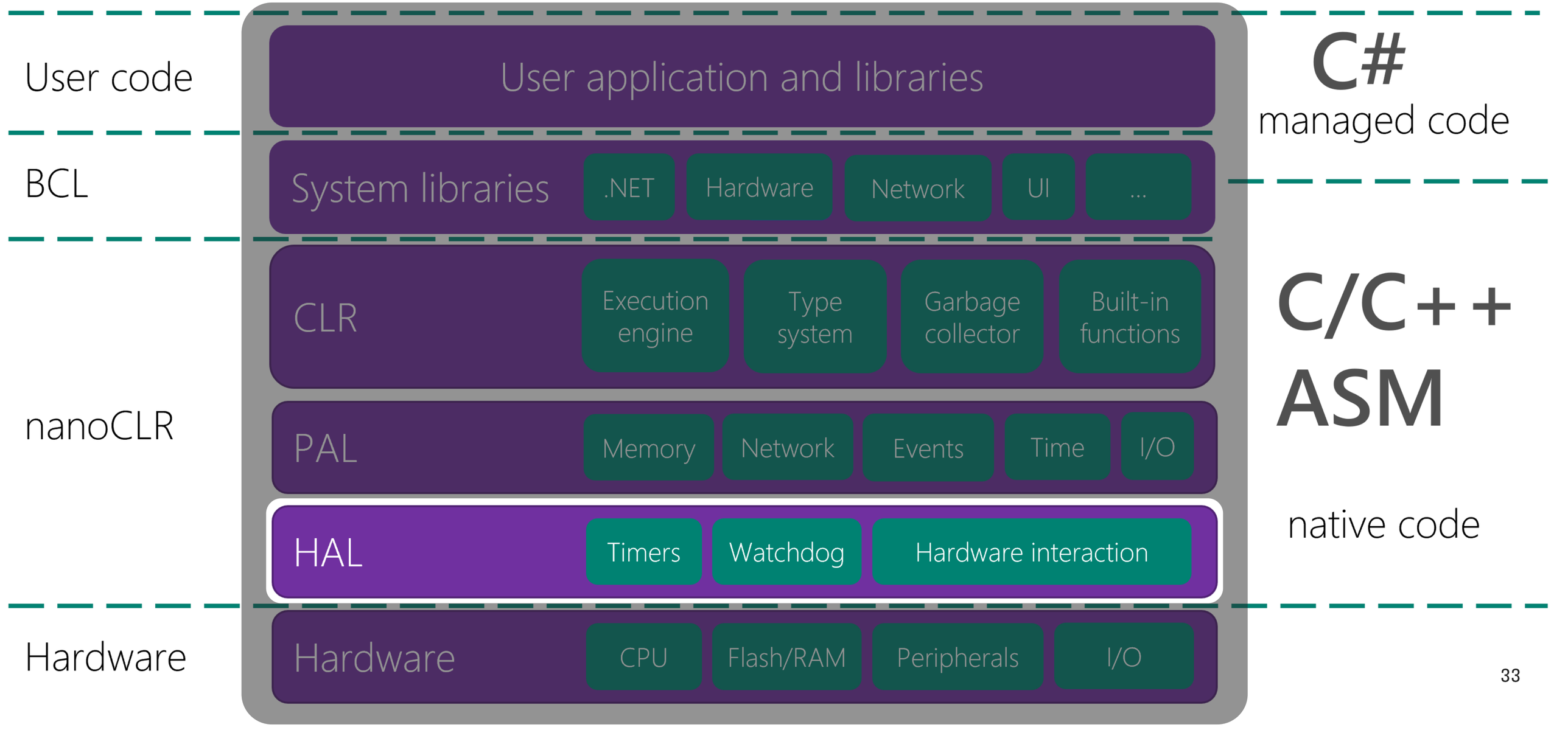
.NET NANOFRAMEWORK



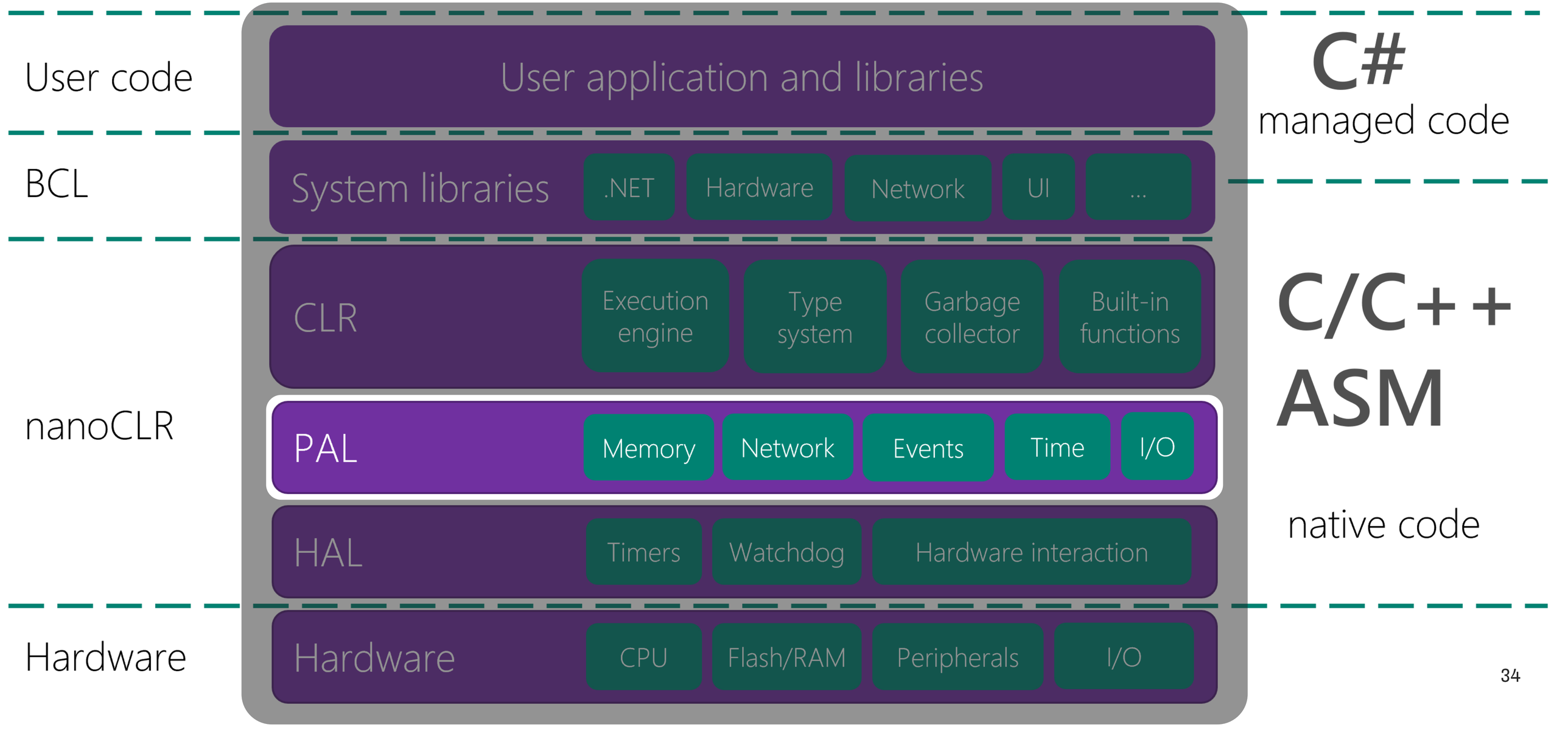
.NET NANOFRAMEWORK



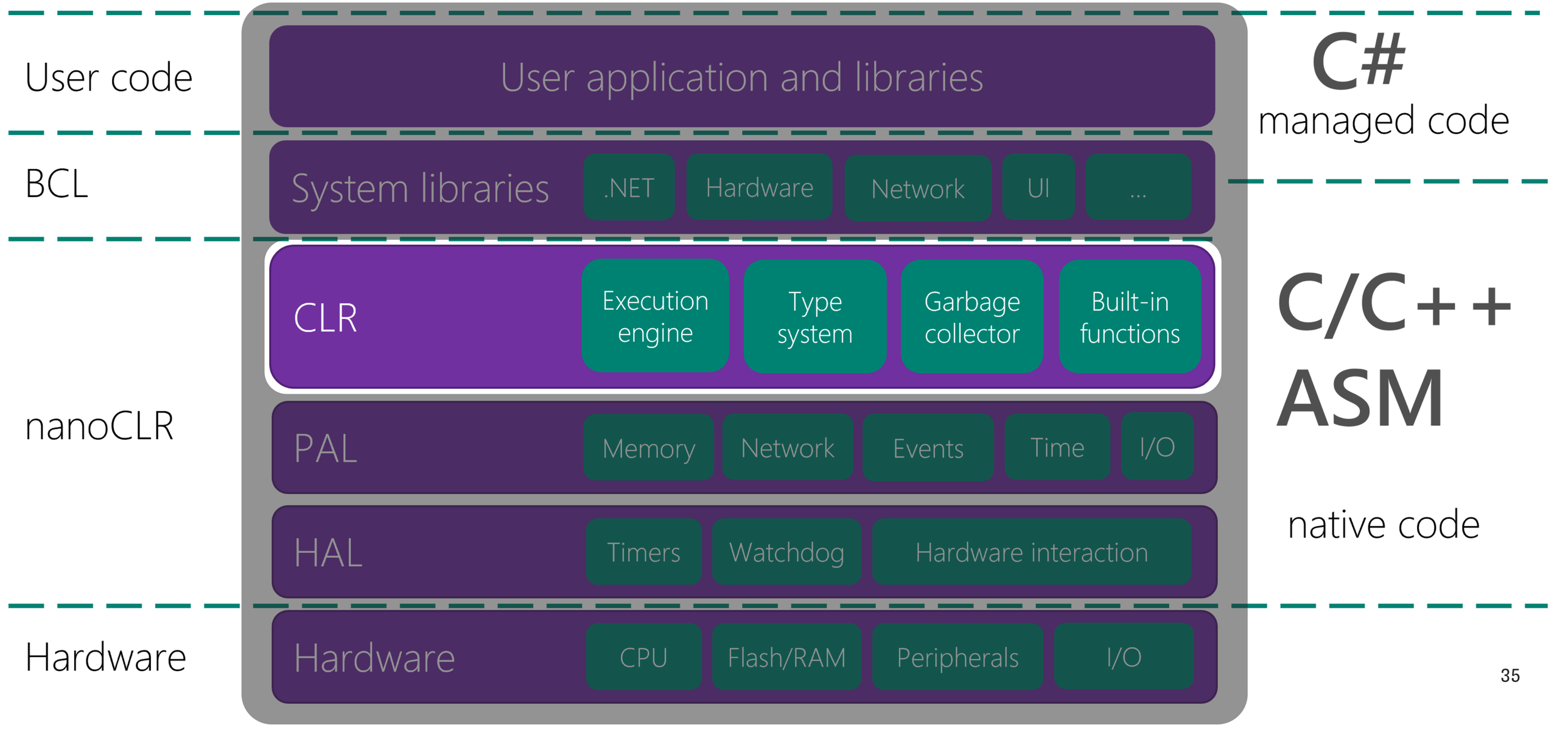
.NET NANOFRAMEWORK



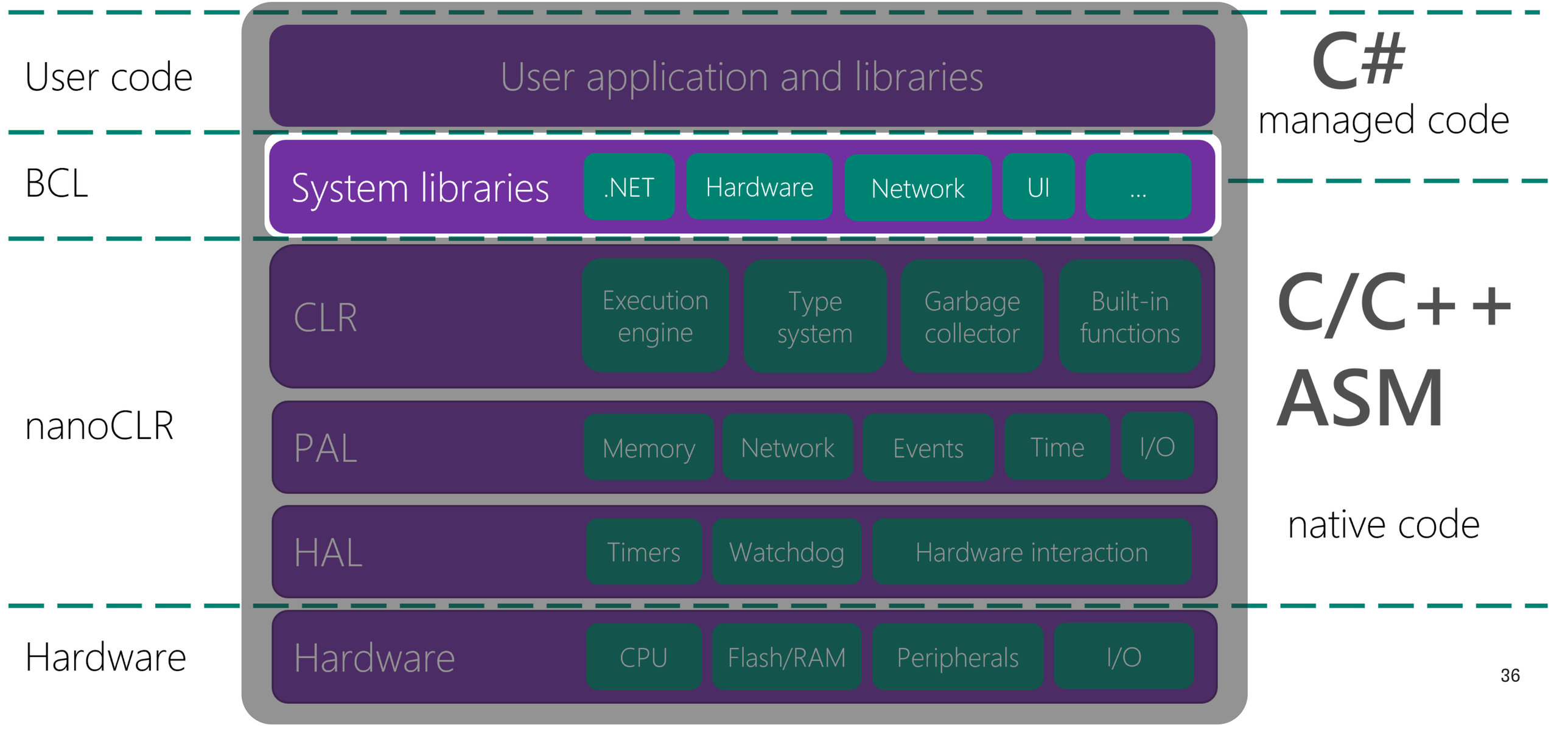
.NET NANOFRAMEWORK



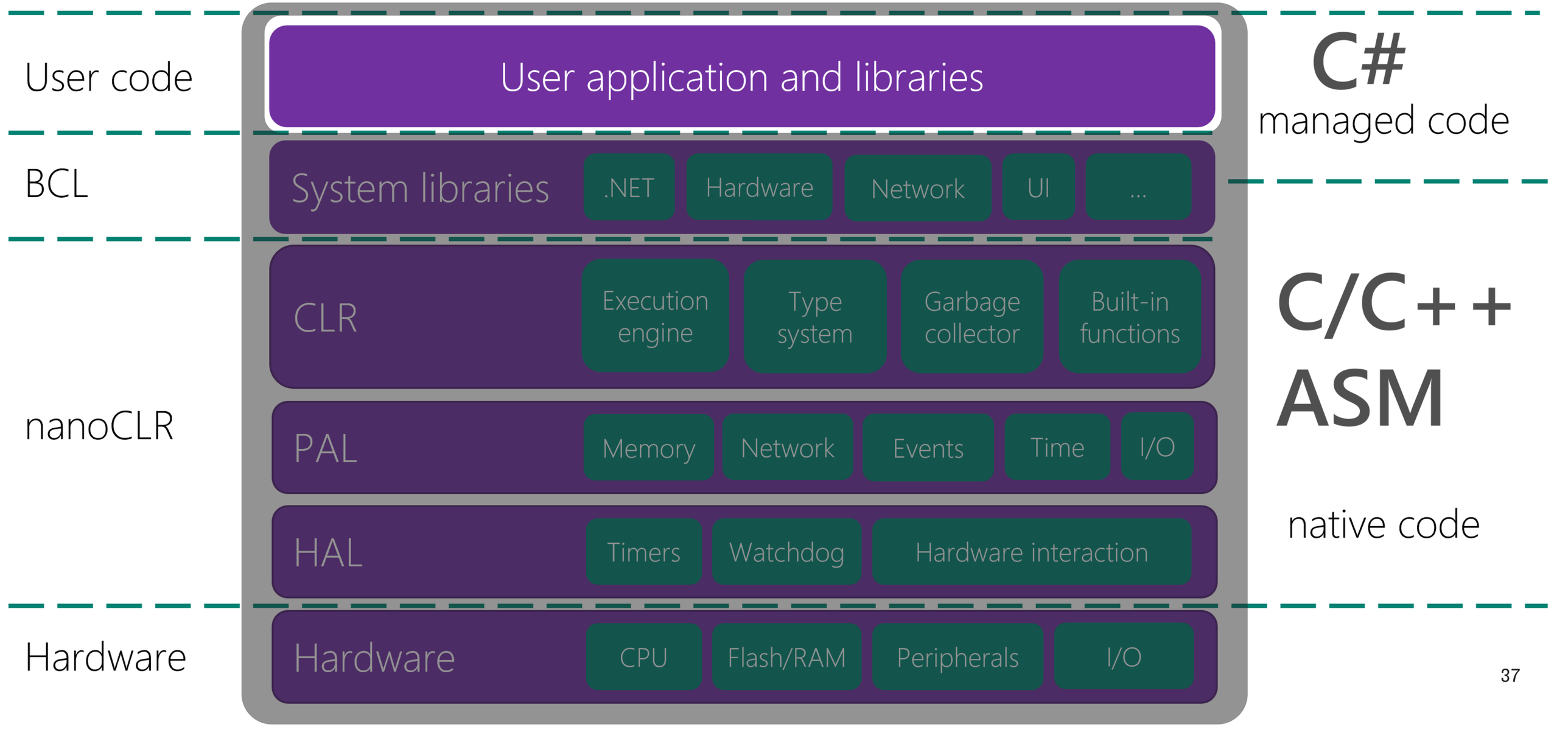
.NET NANOFRAMEWORK



.NET NANOFRAMEWORK



.NET NANOFRAMEWORK



ESP32

2 ядра 240MHz 32bit

520 KB RAM

WIFI

Bluetooth

34 GPIO

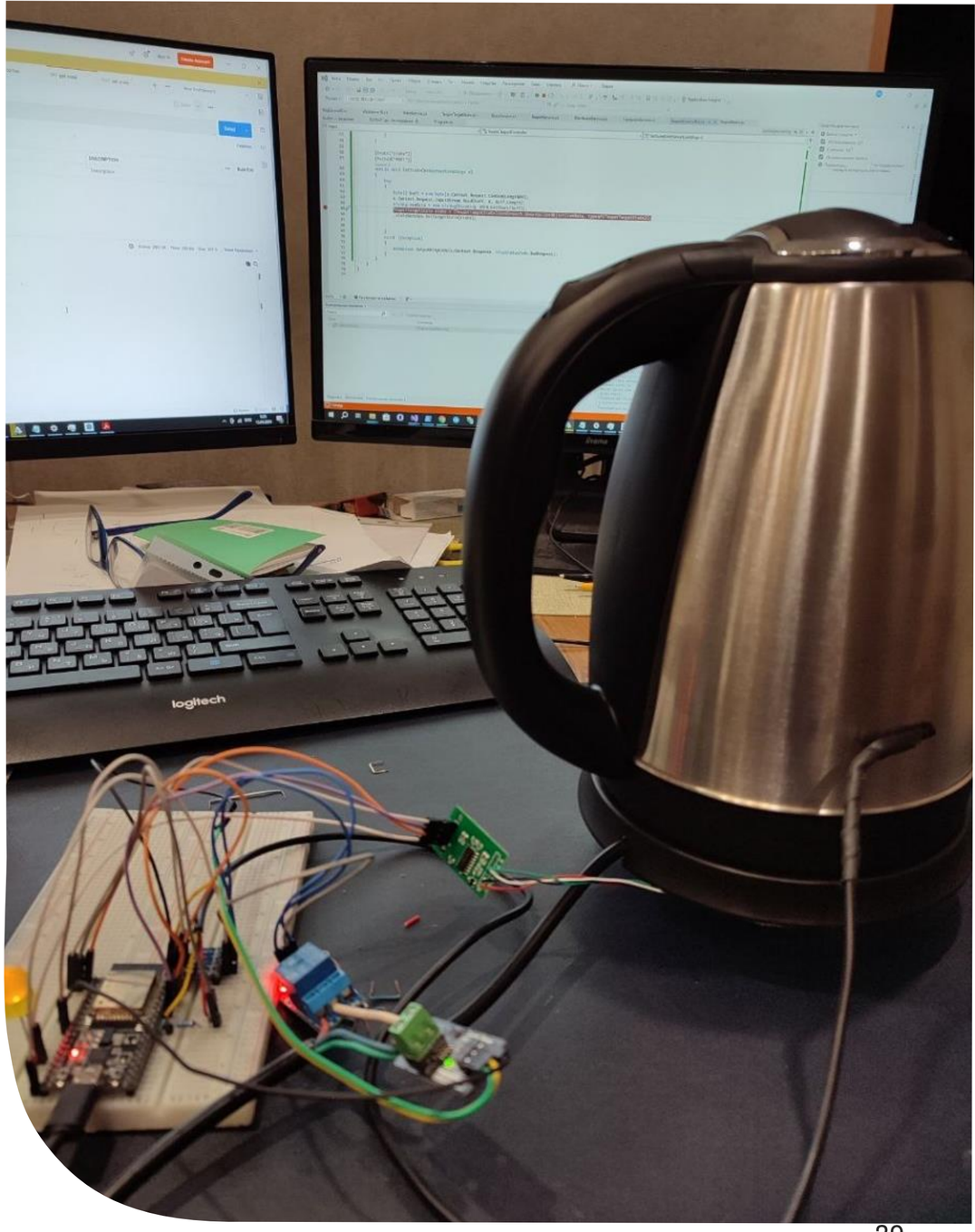
18 x 12-bit ADC, 2 x of 8-bit DAC 4 x SPI, 2 x I^2C,
2 x I^2S, 3 x UART

Free RTOS

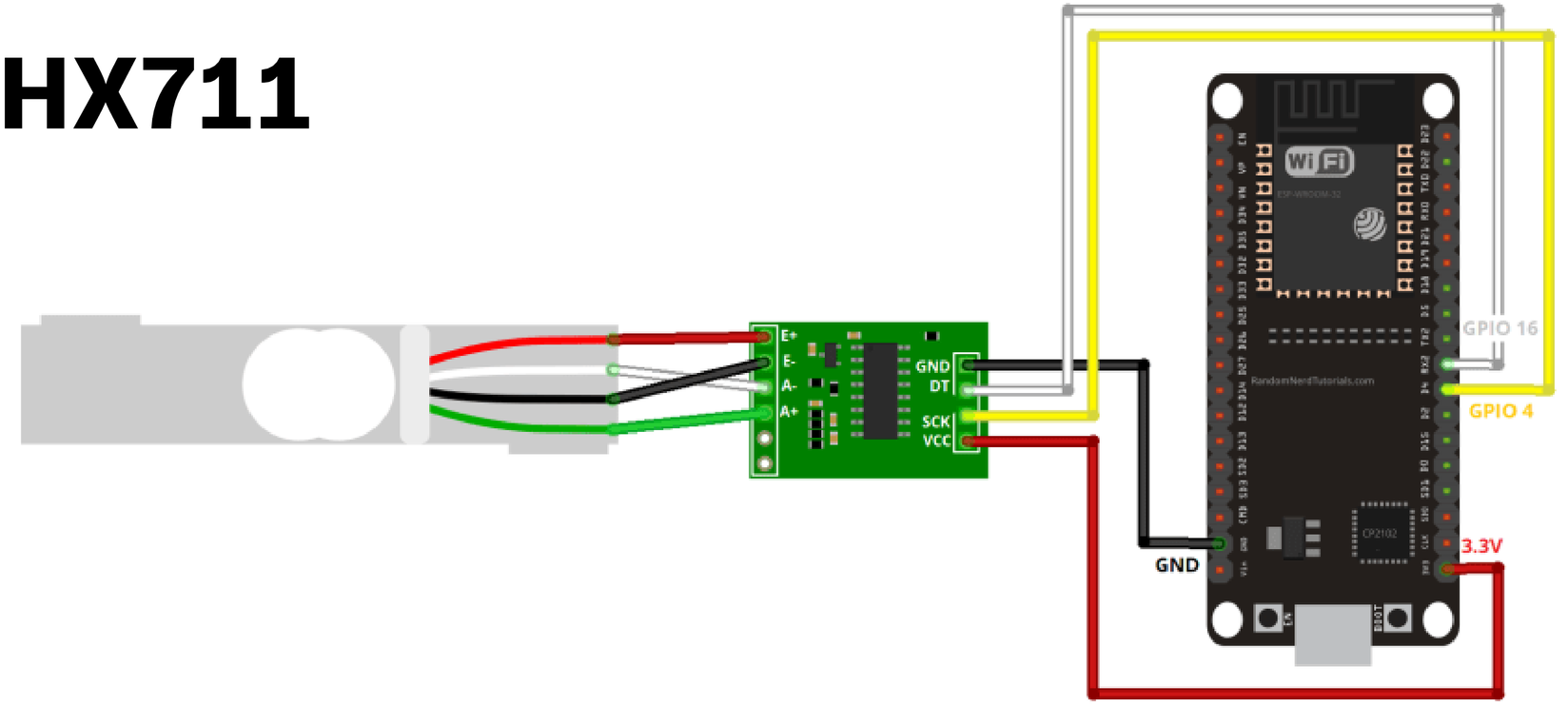
Проект

- Получать состояние устройства
  - уровень воды
  - температура
  - состояние нагревателя

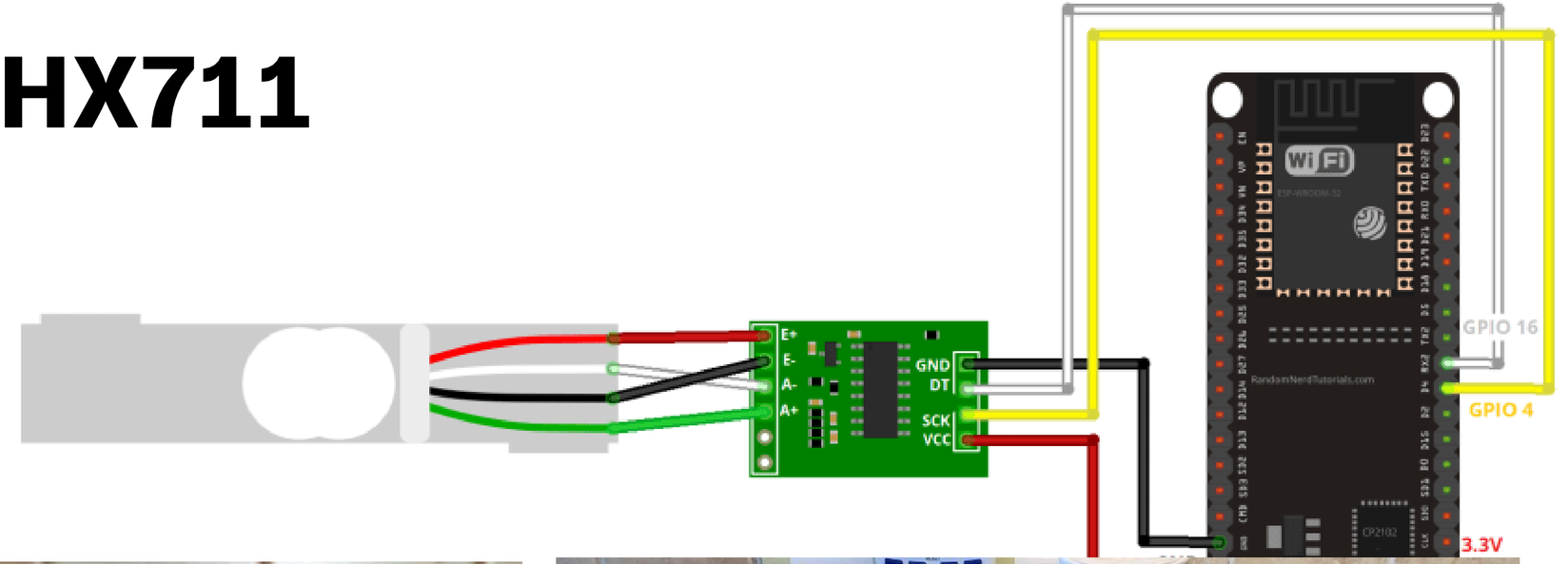
- Управлять устройством
  - целевое состояние

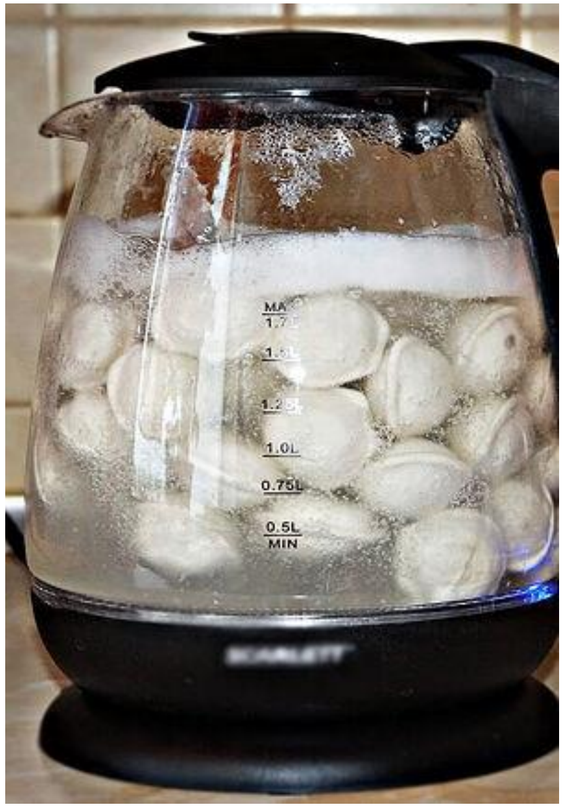


Тензодатчик + HX711



Тензодатчик + HX711





Тензодатчик + HX711

```
//scale init
Configuration.SetPinFunction(21, DeviceFunction.SPI1_MOSI);
Configuration.SetPinFunction(22, DeviceFunction.SPI1_MISO);
Configuration.SetPinFunction(23, DeviceFunction.SPI1_CLOCK);

var spiSettings = new SpiConnectionSettings(1)
{
    ClockFrequency = Scale.DefaultClockFrequency
};
var spidev = SpiDevice.Create(spiSettings);
var scale = new Scale(spidev);

var weight = scale.Read();
```

Тензодатчик + HX711

```
//scale init
Configuration.SetPinFunction(21, DeviceFunction.SPI1_MOSI);
Configuration.SetPinFunction(22, DeviceFunction.SPI1_MISO);
Configuration.SetPinFunction(23, DeviceFunction.SPI1_CLOCK);

var spiSettings = new SpiConnectionSettings(1)
{
    ClockFrequency = Scale.DefaultClockFrequency
};
var spidev = SpiDevice.Create(spiSettings);
var scale = new Scale(spidev);

var weight = scale.Read();
```

Тензодатчик + HX711

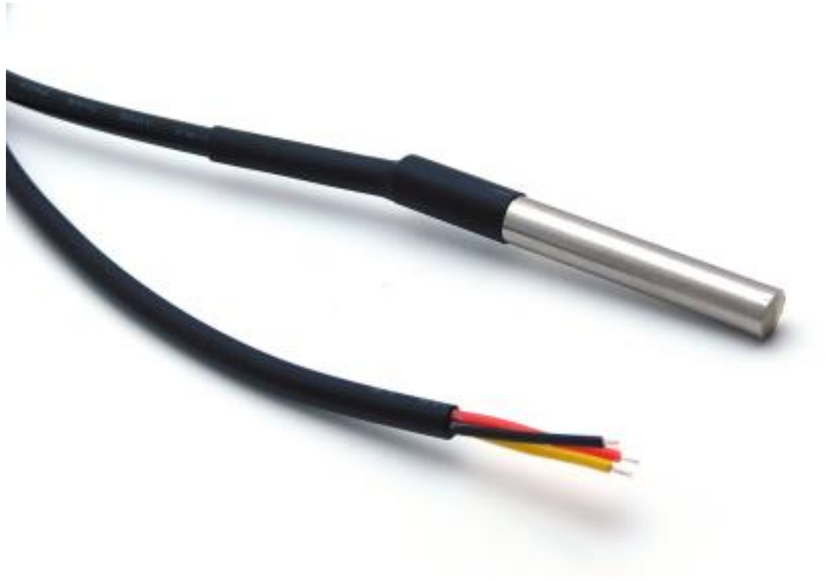
```
//scale init
Configuration.SetPinFunction(21, DeviceFunction.SPI1_MOSI);
Configuration.SetPinFunction(22, DeviceFunction.SPI1_MISO);
Configuration.SetPinFunction(23, DeviceFunction.SPI1_CLOCK);

var spiSettings = new SpiConnectionSettings(1)
{
    ClockFrequency = Scale.DefaultClockFrequency
};
var spidev = SpiDevice.Create(spiSettings);
var scale = new Scale(spidev);

var weight = scale.Read();
```

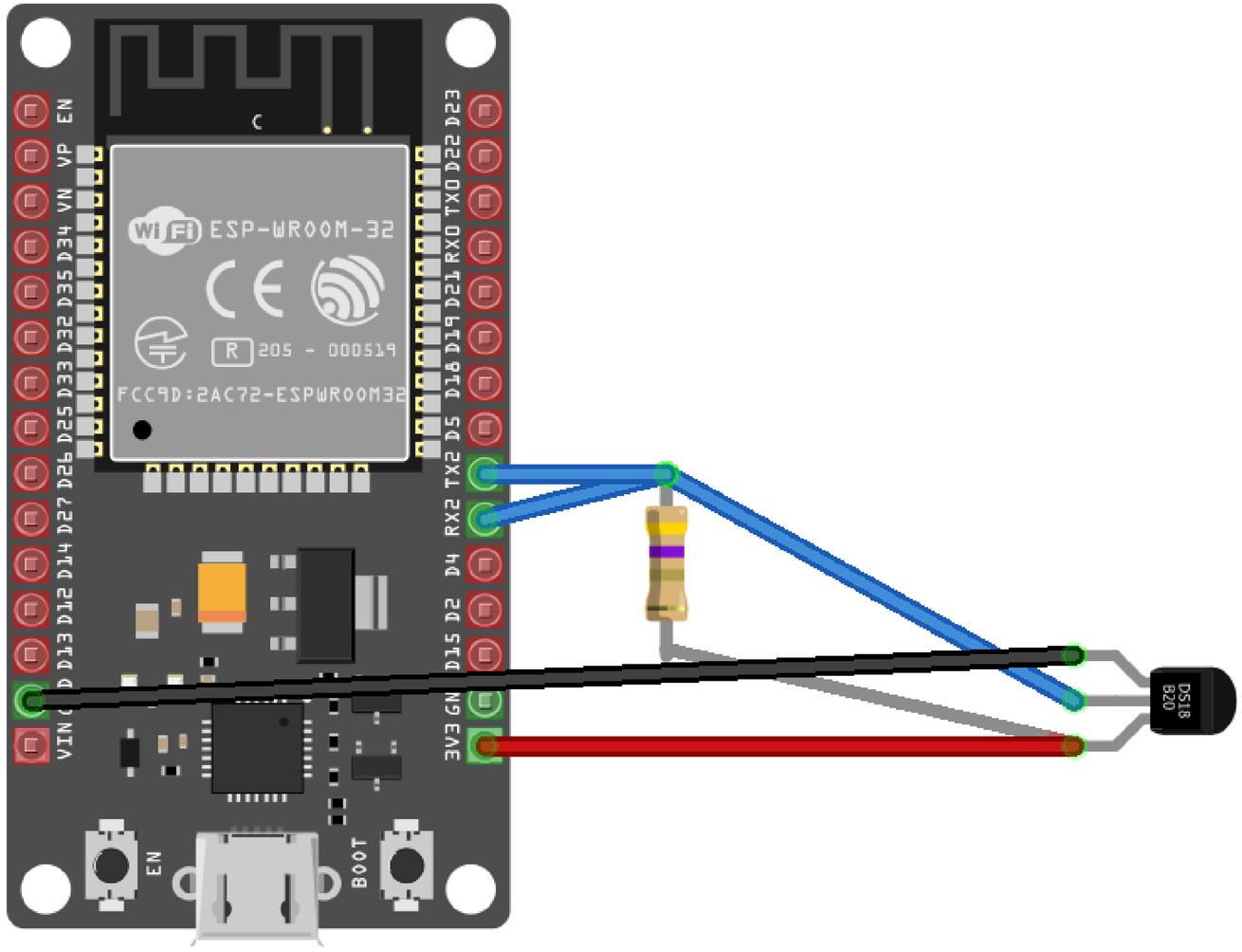
Тензодатчик + HX711

```
//scale init
Configuration.SetPinFunction(21, DeviceFunction.SPI1_MOSI);
Configuration.SetPinFunction(22, DeviceFunction.SPI1_MISO);
Configuration.SetPinFunction(23, DeviceFunction.SPI1_CLOCK);

var spiSettings = new SpiConnectionSettings(1)
{
    ClockFrequency = Scale.DefaultClockFrequency
};
var spidev = SpiDevice.Create(spiSettings);
var scale = new Scale(spidev);
```

```
var weight = scale.Read();
```

Датчик температуры DS1820





Датчик температуры DS1820

```
//ds18b20 init
Configuration.SetPinFunction(18, DeviceFunction.COM3_RX);
Configuration.SetPinFunction(19, DeviceFunction.COM3_TX);

var oneWire = new OneWireHost();
var ds18b20 = new Ds18b20(oneWire)
{
    IsAlarmSearchCommandEnabled = false
};
ds18b20.Initialize();
double temperature;
if (ds18b20.TryReadTemperature(out var currentTemperature))
{
    temperature = currentTemperature.DegreesCelsius;
}
```

Датчик температуры DS1820

```
//ds18b20 init
Configuration.SetPinFunction(18, DeviceFunction.COM3_RX);
Configuration.SetPinFunction(19, DeviceFunction.COM3_TX);

var oneWire = new OneWireHost();
var ds18b20 = new Ds18b20(oneWire)
{
    IsAlarmSearchCommandEnabled = false
};
ds18b20.Initialize();
double temperature;
if (ds18b20.TryReadTemperature(out var currentTemperature))
{
    temperature = currentTemperature.DegreesCelsius;
}
```

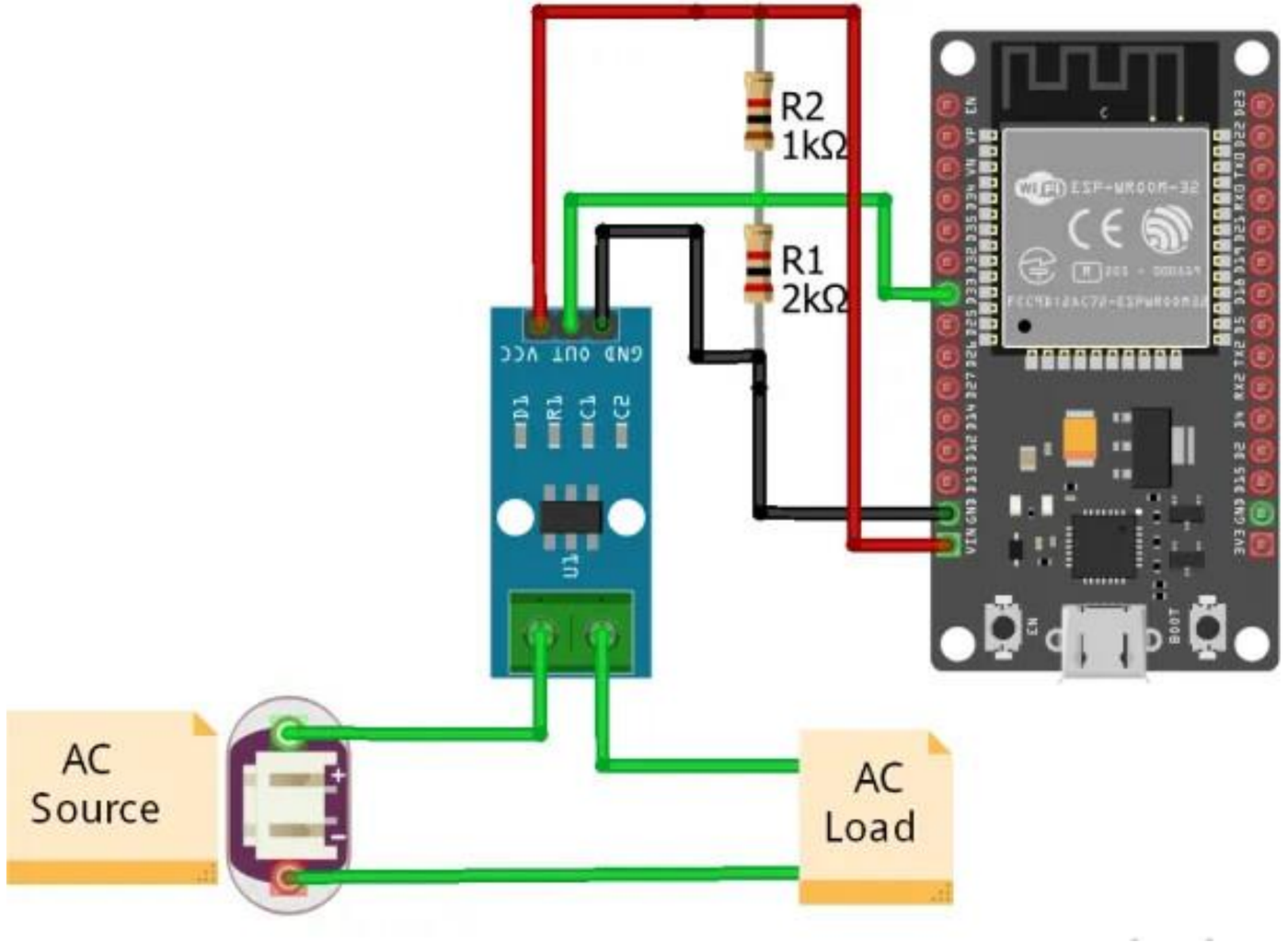
Датчик температуры DS1820

```
//ds18b20 init
Configuration.SetPinFunction(18, DeviceFunction.COM3_RX);
Configuration.SetPinFunction(19, DeviceFunction.COM3_TX);

var oneWire = new OneWireHost();
var ds18b20 = new Ds18b20(oneWire)
{
    IsAlarmSearchCommandEnabled = false
};
ds18b20.Initialize();
double temperature;
if (ds18b20.TryReadTemperature(out var currentTemperature))
{
    temperature = currentTemperature.DegreesCelsius;
}
```

Датчик температуры DS1820

```
//ds18b20 init
Configuration.SetPinFunction(18, DeviceFunction.COM3_RX);
Configuration.SetPinFunction(19, DeviceFunction.COM3_TX);

var oneWire = new OneWireHost();
var ds18b20 = new Ds18b20(oneWire)
{
    IsAlarmSearchCommandEnabled = false
};
ds18b20.Initialize();
double temperature;
if (ds18b20.TryReadTemperature(out var currentTemperature))
{
    temperature = currentTemperature.DegreesCelsius;
}
```

Датчик тока ACS712



Датчик тока ACS712

```
//ADC init
var adc = new AdcController();
var adcChannel = adc.OpenChannel(4);

var adcRawValue = adcChannel.ReadValue();
```

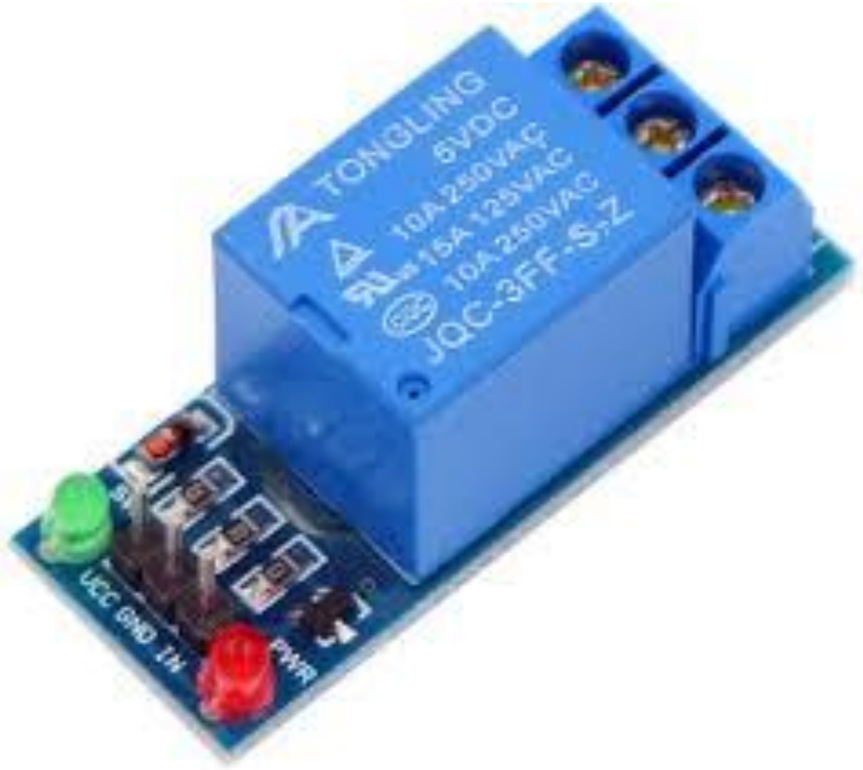
Датчик тока ACS712

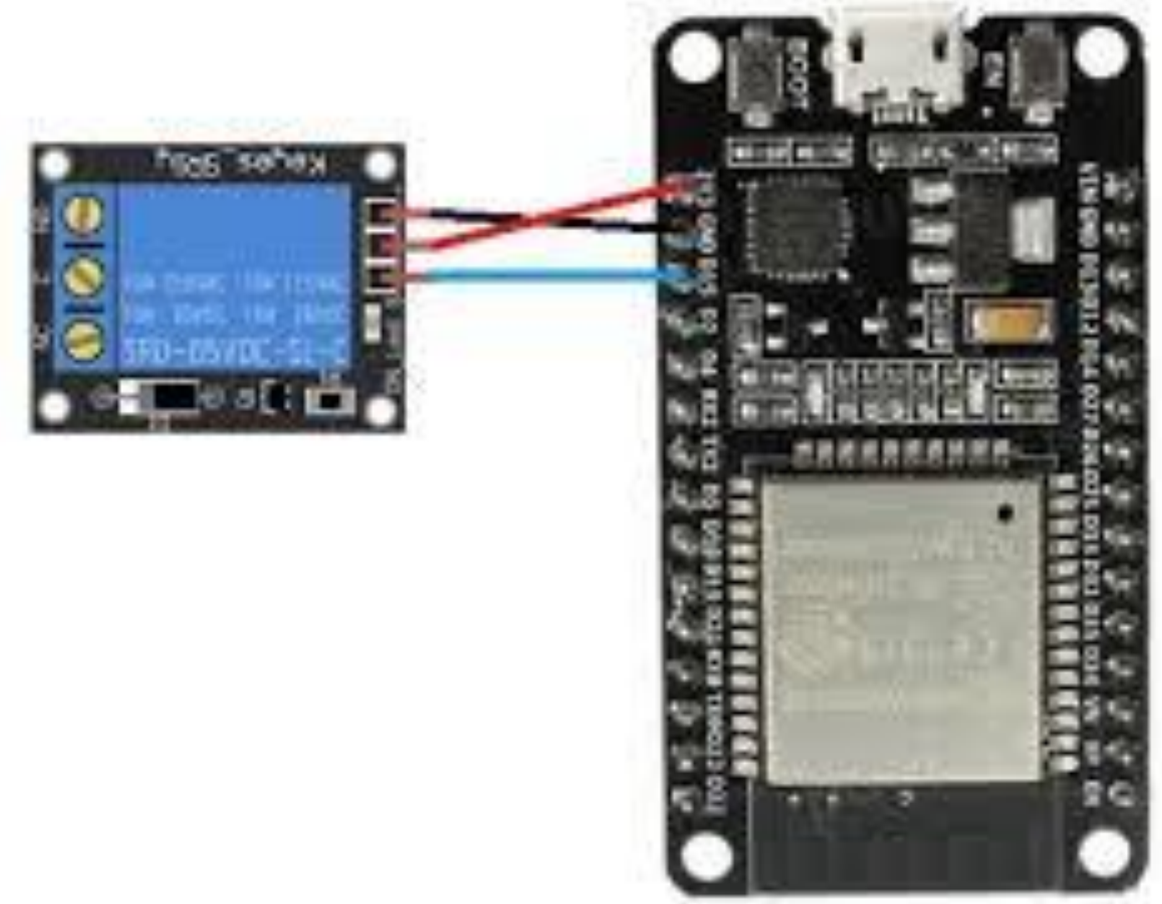
```
//ADC init
var adc = new AdcController();
var adcChannel = adc.OpenChannel(4);

var adcRawValue = adcChannel.ReadValue();
```

Датчик тока ACS712

```
//ADC init
var adc = new AdcController();
var adcChannel = adc.OpenChannel(4);

var adcRawValue = adcChannel.ReadValue();
```

Реле


TONGLING
5VDC
10A 250VAC
15A 125VAC
10A 250VAC
JQC-3FF-S-Z
VCC GND IN
PWR



Реле

```
//relay init
var gpioController = new GpioController();
var led = gpioController.OpenPin(25, PinMode.Output);

led.Write(PinValue.High);
Thread.Sleep(1000);
led.Write(PinValue.Low);
```

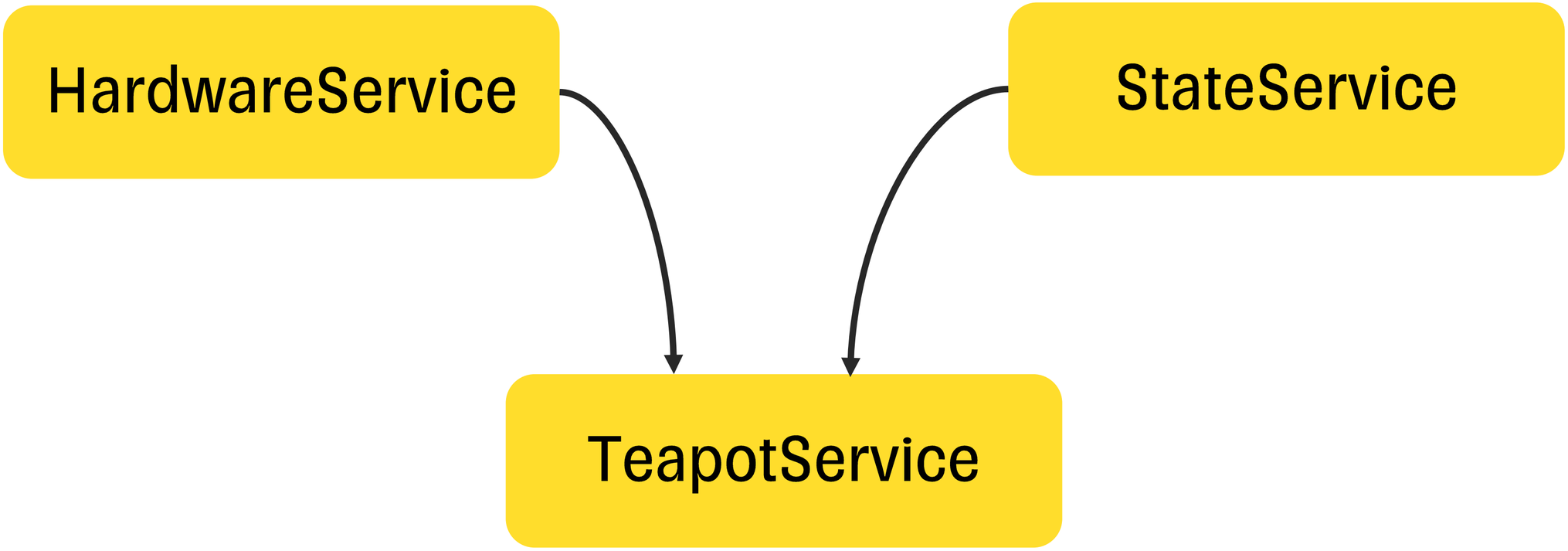
Приложение

```
//ds18b20 init
Configuration.SetPinFunction(18, DeviceFunction.COM3_RX);
Configuration.SetPinFunction(19, DeviceFunction.COM3_TX);
_oneWire = new OneWireHost();
_ds18b20 = new Ds18b20(_oneWire);
_ds18b20.IsAlarmSearchCommandEnabled = false;
_ds18b20.Initialize();

//scale init
Configuration.SetPinFunction(21, DeviceFunction.SPI1_MOSI);
Configuration.SetPinFunction(22, DeviceFunction.SPI1_MISO);
Configuration.SetPinFunction(23, DeviceFunction.SPI1_CLOCK);
var spisettings = new SpiConnectionSettings(1)
{
    ClockFrequency = Scale.DefaultClockFrequency
};
var spidev = SpiDevice.Create(spisettings);
_scale = new Scale(spidev);

while (true)
{
    int temperature;
    if (_ds18b20.TryReadTemperature(out var currentTemperature))
    {
        temperature = (int)currentTemperature.DegreesCelsius;
    }
    var weight = (int)_scale.Read();
    int adcRawValue = _adcChannel.ReadValue();

    Thread.Sleep(1000);
```

Приложение

```
//ds18b20 init
Configuration.SetPinFunction(18, DeviceFunction.COM3_RX);
Configuration.SetPinFunction(19, DeviceFunction.COM3_TX);
_oneWire = new OneWireHost();
_ds18b20 = new Ds18b20(_oneWire);
_ds18b20.IsAlarmSearchCommandEnabled = false;
_ds18b20.Initialize();

//scale init
Configuration.SetPinFunction(21, DeviceFunction.SPI1_MOSI);
Configuration.SetPinFunction(22, DeviceFunction.SPI1_MISO);
Configuration.SetPinFunction(23, DeviceFunction.SPI1_CLOCK);
var spisettings = new SpiConnectionSettings(1)
{
    ClockFrequency = Scale.DefaultClockFrequency
};
var spidev = SpiDevice.Create(spisettings);
_scale = new Scale(spidev);

while (true)
{
    int temperature;
    if (_ds18b20.TryReadTemperature(out var currentTemperature))
    {
        temperature = (int)currentTemperature.DegreesCelsius;
    }
    var weight = (int)_scale.Read();
    int adcRawValue = _adcChannel.ReadValue();

    Thread.Sleep(1000);
```

Сервисы



Сервисы

HardwareService

```
namespace Teapot.Services
{
    public interface IHardwareService
    {
        TeapotState GetState();

        void SetRelay(bool value);
    }
}
```

Сервисы

StateService

```
public interface IStateService
{
    public TeapotState GetState();
    public void SetState(TeapotState state);

    public TeapotTargetState GetTargetState();
    public void SetTargetState(TeapotTargetState state);
}
```

Сервисы

TeapotService

```
public class TeapotService : SchedulerService
{
    IStateService _stateService;
    IHardwareService _hardwareService;

    public TeapotService(IStateService stateService,
        IHardwareService hardwareService)
        : base(TimeSpan.FromSeconds(1))
    {
        _stateService = stateService;
        _hardwareService = hardwareService;
    }

    protected override void ExecuteAsync()
    {
        var targetState = _stateService.GetTargetState();
        var currentState = _hardwareService.GetState();

        if (currentState.WaterLevel < 10)
            return;

        if ((currentState.Temperature < targetState.Temperature) &&
            targetState.KeepTemperature)
        {
```

Сервисы

TeapotService

```
public class TeapotService : SchedulerService
{
    IStateService _stateService;
    IHardwareService _hardwareService;

    public TeapotService(IStateService stateService,
        IHardwareService hardwareService)
        : base(TimeSpan.FromSeconds(1))
    {
        _stateService = stateService;
        _hardwareService = hardwareService;
    }

    protected override void ExecuteAsync()
    {
        var targetState = _stateService.GetTargetState();
        var currentState = _hardwareService.GetState();

        if (currentState.WaterLevel < 10)
            return;

        if ((currentState.Temperature < targetState.Temperature) &&
            targetState.KeepTemperature)
        {
```

Сервисы

TeapotService

```
public class TeapotService : SchedulerService
{
    IStateService _stateService;
    IHardwareService _hardwareService;

    public TeapotService(IStateService stateService,
        IHardwareService hardwareService)
        : base(TimeSpan.FromSeconds(1))
    {
        _stateService = stateService;
        _hardwareService = hardwareService;
    }

    protected override void ExecuteAsync()
    {
        var targetState = _stateService.GetTargetState();
        var currentState = _hardwareService.GetState();

        if (currentState.WaterLevel < 10)
            return;

        if ((currentState.Temperature < targetState.Temperature) &&
            targetState.KeepTemperature)
        {
```

Сервисы

TeapotService

```
public class TeapotService : SchedulerService
{
    IStateService _stateService;
    IHardwareService _hardwareService;

    public TeapotService(IStateService stateService,
        IHardwareService hardwareService)
        : base(TimeSpan.FromSeconds(1))
    {
        _stateService = stateService;
        _hardwareService = hardwareService;
    }

    protected override void ExecuteAsync()
    {
        var targetState = _stateService.GetTargetState();
        var currentState = _hardwareService.GetState();

        if (currentState.WaterLevel < 10)
            return;

        if ((currentState.Temperature < targetState.Temperature) &&
            targetState.KeepTemperature)
        {
```

Сервисы

```
public static IHostBuilder CreateHostBuilder() =>
    Host.CreateDefaultBuilder()
      .ConfigureServices(services =>
        {
            services.AddSingleton(typeof(IStateService),
                typeof(StateService));
            services.AddSingleton(typeof(IHardwareService),
                typeof(HardwareService));
            services.AddHostedService(typeof(TeapotService));
        });
```

Сервисы

```
public static IHostBuilder CreateHostBuilder() =>
    Host.CreateDefaultBuilder()
       .ConfigureServices(services =>
         {
             services.AddSingleton(typeof(IStateService),
                 typeof(StateService));
             services.AddSingleton(typeof(IHardwareService),
                 typeof(HardwareService));
             services.AddHostedService(typeof(TeapotService));
         });


IHost host = CreateHostBuilder().Build();
host.Run();
```

WIFI

```
Debug.WriteLine("Waiting for network up and IP address...");
bool success;
CancellationTokenSource cs = new(60000);
success = WifiNetworkHelper.ConnectDhcp(MySsid, MyPassword,
    token: cs.Token);
if (!success)
{
    Debug.WriteLine($"Can't connect, error: {WifiNetworkHelper.Status}.");
    if (WifiNetworkHelper.HelperException != null)
    {
        Debug.WriteLine($"Exception: {WifiNetworkHelper.HelperException}");
    }
    return;
}
Console.WriteLine($"Connected to network {MySsid}");
```

WIFI

```
Debug.WriteLine("Waiting for network up and IP address...");
bool success;
CancellationTokenSource cs = new(60000);
success = WifiNetworkHelper.ConnectDhcp(MySsid, MyPassword,
    token: cs.Token);
if (!success)
{
    Debug.WriteLine($"Can't connect, error: {WifiNetworkHelper.Status}.");
    if (WifiNetworkHelper.HelperException != null)
    {
        Debug.WriteLine($"Exception: {WifiNetworkHelper.HelperException}");
    }
    return;
}
Console.WriteLine($"Connected to network {MySsid}");
```

Контроллер

```
class TeapotController
{
    private readonly IStateService _stateService;

    public TeapotController(IStateService stateService)
    {
        _stateService = stateService;
    }

    [Route("coffee")]
    [Method("GET")]
    public void Coffee(WebServerEventArgs e)
    {
       e.Context.Response.StatusCode = 418;
    }
```

Контроллер

```
class TeapotController
{
    private readonly IStateService _stateService;

    public TeapotController(IStateService stateService)
    {
        _stateService = stateService;
    }

    [Route("coffee")]
    [Method("GET")]
    public void Coffee(WebServerEventArgs e)
    {
       e.Context.Response.StatusCode = 418;
    }
```

Контроллер

```
class TeapotController
{
    private readonly IStateService _stateService;

    public TeapotController(IStateService stateService)
    {
        _stateService = stateService;
    }

    [Route("coffee")]
    [Method("GET")]
    public void Coffee(WebServerEventArgs e)
    {
       e.Context.Response.StatusCode = 418;
    }
```

Контроллер

```
class TeapotController
{
    private readonly IStateS

    public TeapotController(
    {
        _stateService = stat
    }
```

```
    [Route("coffee")]
    [Method("GET")]
    public void Coffee(WebServerEventArgs e)
    {
        e.Context.Response.StatusCode = 418;
    }
```

418 I'm a teapot

The HTTP `418 I'm a teapot` client error response code indicates that the server refuses to brew coffee because it is, permanently, a teapot. A combined coffee/tea pot that is temporarily out of coffee should instead return 503. This error is a reference to Hyper Text Coffee Pot Control Protocol defined in April Fools' jokes in 1998 and 2014.

Контроллер

```
[Route("state")]
[Method("GET")]
public void GetState(WebServerEventArgs e)
{
    try
    {
        var state = _stateService.GetState();
        var content = JsonConvert.SerializeObject(state);
        e.Context.Response.ContentType = "application/json";
        WebServer.OutPutStream(e.Context.Response, content);

    }
    catch (Exception)
    {
        WebServer.OutputHttpCode(e.Context.Response, HttpStatusCode.BadRequest);
    }
}
```

Контроллер

```
[Route("state")]
[Method("POST")]
public void SetState(WebServerEventArgs e)
{
    try
    {
        byte[] buff = new byte[e.Context.Request.ContentLength64];
        e.Context.Request.InputStream.Read(buff, 0, buff.Length);
        string rawData = new string(Encoding.UTF8.GetChars(buff));
        TeapotTargetState state =
            (TeapotTargetState)JsonConvert.DeserializeObject(rawData, typeof(TeapotTargetState));

        _stateService.SetTargetState(state);
    }
    catch (Exception)
    {
        WebServer.OutputHttpCode(e.Context.Response, HttpStatusCode.BadRequest);
    }
}
```

Веб сервер

```
// Instantiate a new web server on port 80.
using (var webServer = new WebServerDi(80,
    HttpProtocol.Http,
    new Type[] { typeof(TeapotController) },
    serviceProvider))
{
    webServer.Start();
    Thread.Sleep(Timeout.Infinite);
}
```

Веб сервер

```
// Instantiate a new web server on port 80.
using (var webServer = new WebServerDi(80,
    HttpProtocol.Http,
    new Type[] { typeof(TeapotController) },
    serviceProvider))
{
    webServer.Start();
    Thread.Sleep(Timeout.Infinite);
}
```

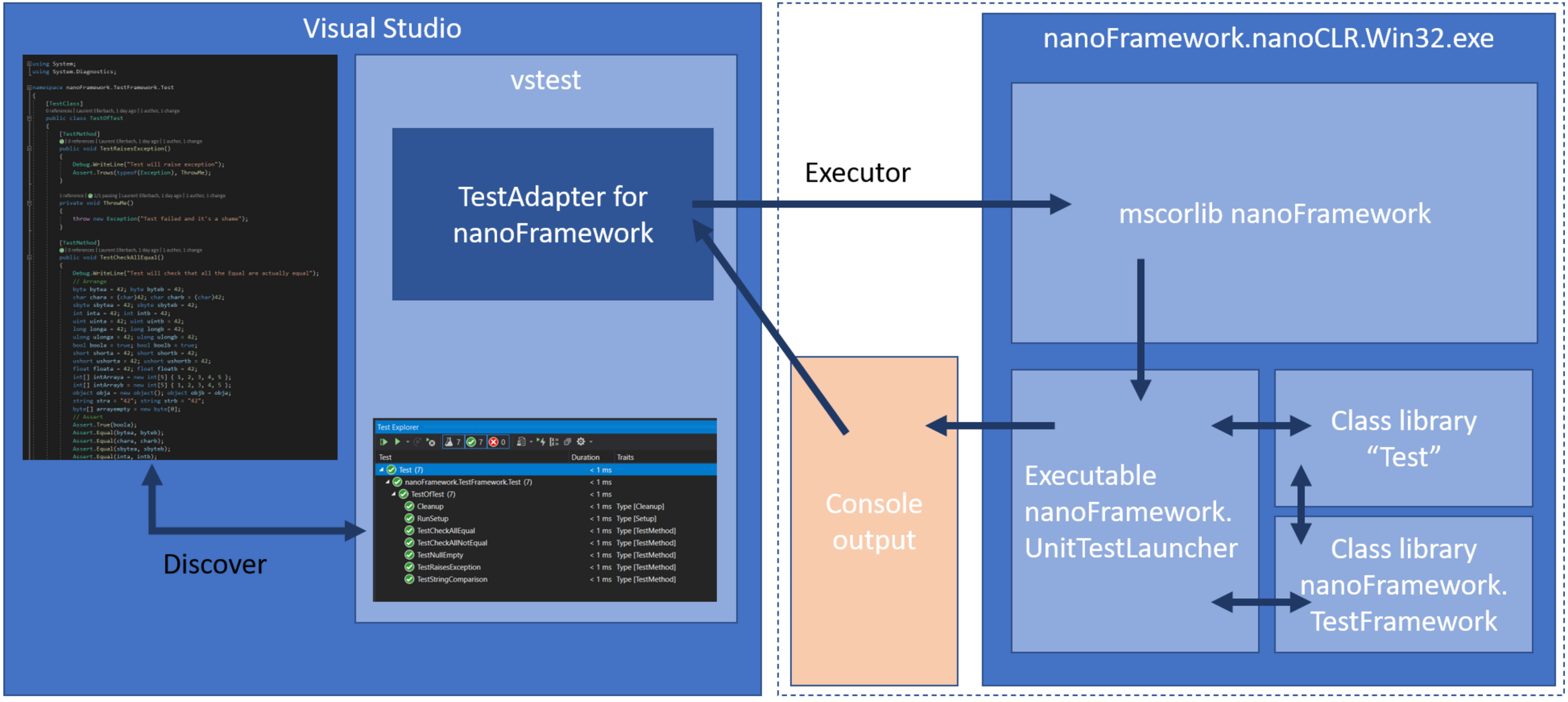
Веб сервер

```
// Instantiate a new web server on port 80.
using (var webServer = new WebServerDi(80,
    HttpProtocol.Http,
    new Type[] { typeof(TeapotController) },
    serviceProvider))
{
    webServer.Start();
    Thread.Sleep(Timeout.Infinite);
}
```

Веб сервер

```
// Instantiate a new web server on port 80.
using (var webServer = new WebServerDi(80,
    HttpProtocol.Http,
    new Type[] { typeof(TeapotController) },
    serviceProvider))
{
    webServer.Start();
    Thread.Sleep(Timeout.Infinite);
}
```

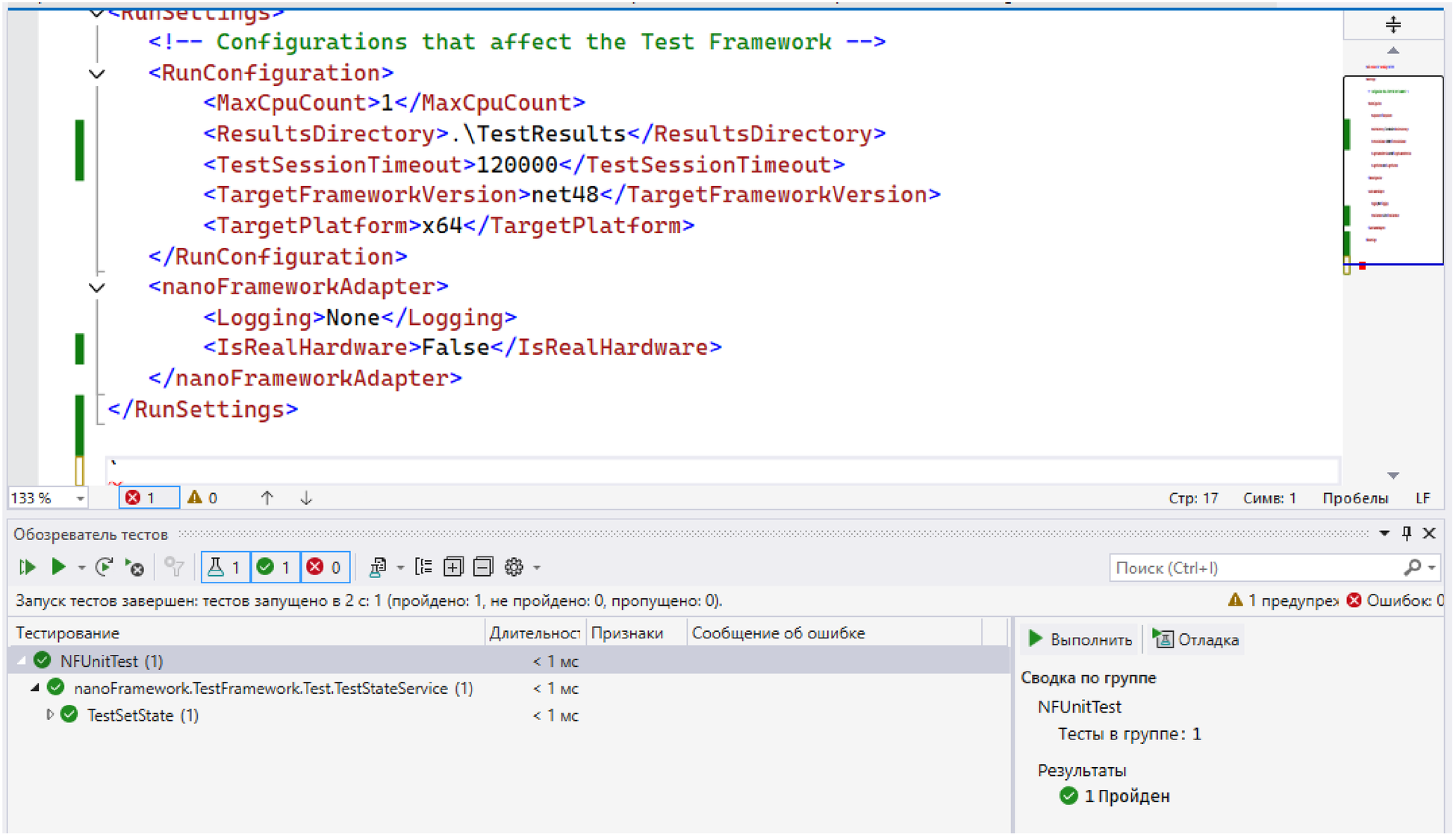
Тесты


Visual Studio
vstest
TestAdapter for nanoFramework
Discover
Test Explorer
Test
Duration
Traits
Test (7)
nanoFramework.TestFramework.Test (7)
TestOfTest (7)
Cleanup
RunSetup
TestCheckAllEqual
TestCheckAllNotEqual
TestNullEmpty
TestRaisesException
TestStringComparison
Executor
Console output
nanoFramework.nanoCLR.Win32.exe
mscorlib nanoFramework
Executable nanoFramework. UnitTestLauncher
Class library “Test”
Class library nanoFramework. TestFramework

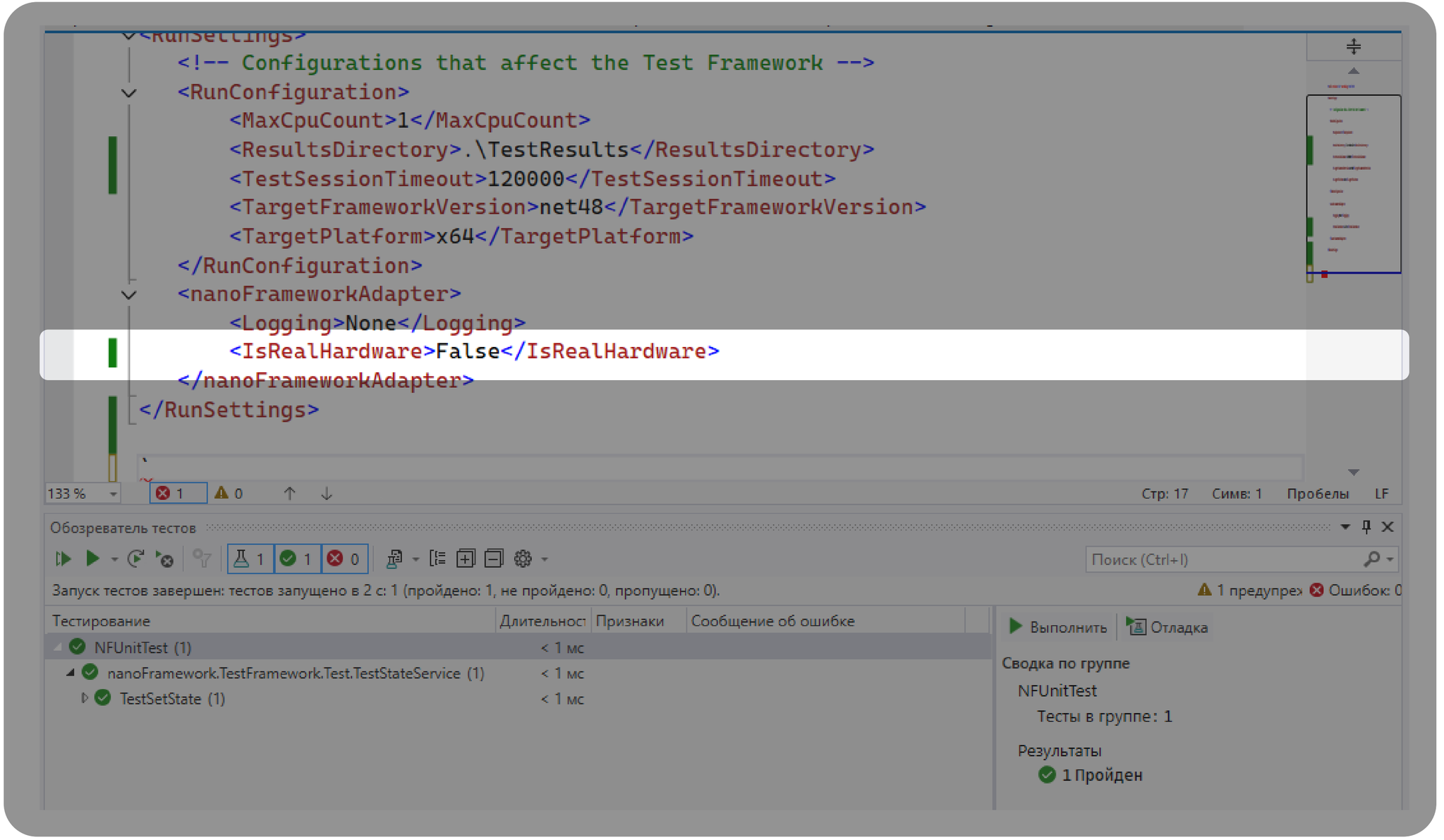
Тесты

```
using NFUnitTest;

namespace nanoFramework.TestFramework.Test
{
  [TestClass]
  public class TestStateService
  {
    [TestMethod]
    public void TestSetState()
    {
      IStateService service = new StateService();
      TeapotState state = new TeapotState()
      {
        IsOn = true,
        Temperature = 99,
        WaterLevel = 33
      };
      service.SetState(state);
      var newState = service.GetState();
      Assert.AreEqual(state, newState);
    }
  }
}
```

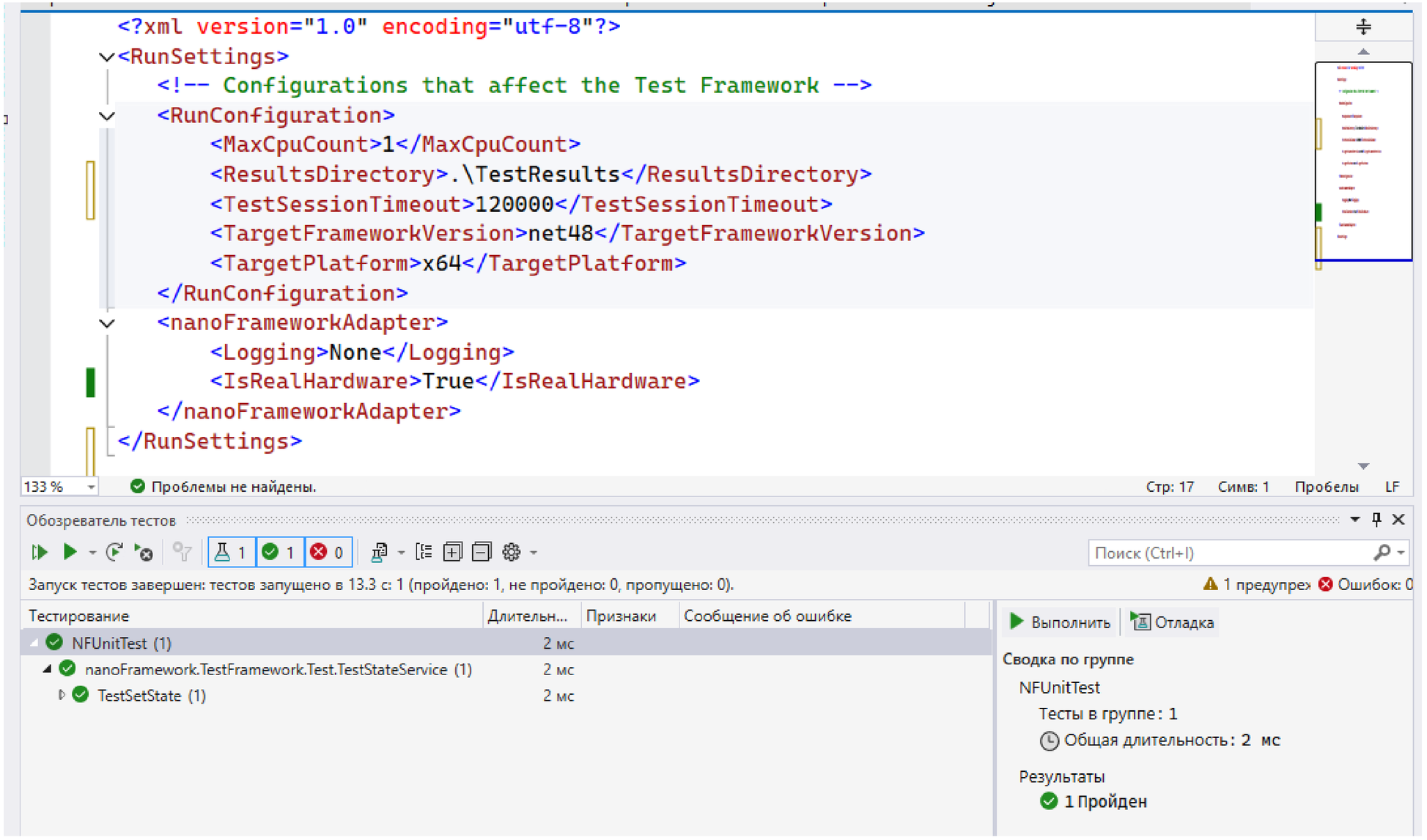
Тесты


<RunSettings>
<!-- Configurations that affect the Test Framework -->
<RunConfiguration>
<MaxCpuCount>1</MaxCpuCount>
<ResultsDirectory>.\TestResults</ResultsDirectory>
<TestSessionTimeout>120000</TestSessionTimeout>
<TargetFrameworkVersion>net48</TargetFrameworkVersion>
<TargetPlatform>x64</TargetPlatform>
</RunConfiguration>
<nanoFrameworkAdapter>
<Logging>None</Logging>
<IsRealHardware>False</IsRealHardware>
</nanoFrameworkAdapter>
</RunSettings>
133 %
1
0
Стр: 17
Симв: 1
Пробелы
LF
Обозреватель тестов
1
1
0
Поиск (Ctrl+I)
Запуск тестов завершен: тестов запущено в 2 с: 1 (пройдено: 1, не пройдено: 0, пропущено: 0).
1 предупрежд
Ошибок: 0
Тестирование
Длительност
Признаки
Сообщение об ошибке
NFUnitTest (1)
< 1 мс
nanoFramework.TestFramework.Test.TestStateService (1)
< 1 мс
TestSetState (1)
< 1 мс
Выполнить
Отладка
Сводка по группе
NFUnitTest
Тесты в группе: 1
Результаты
1 Пройден

Тесты

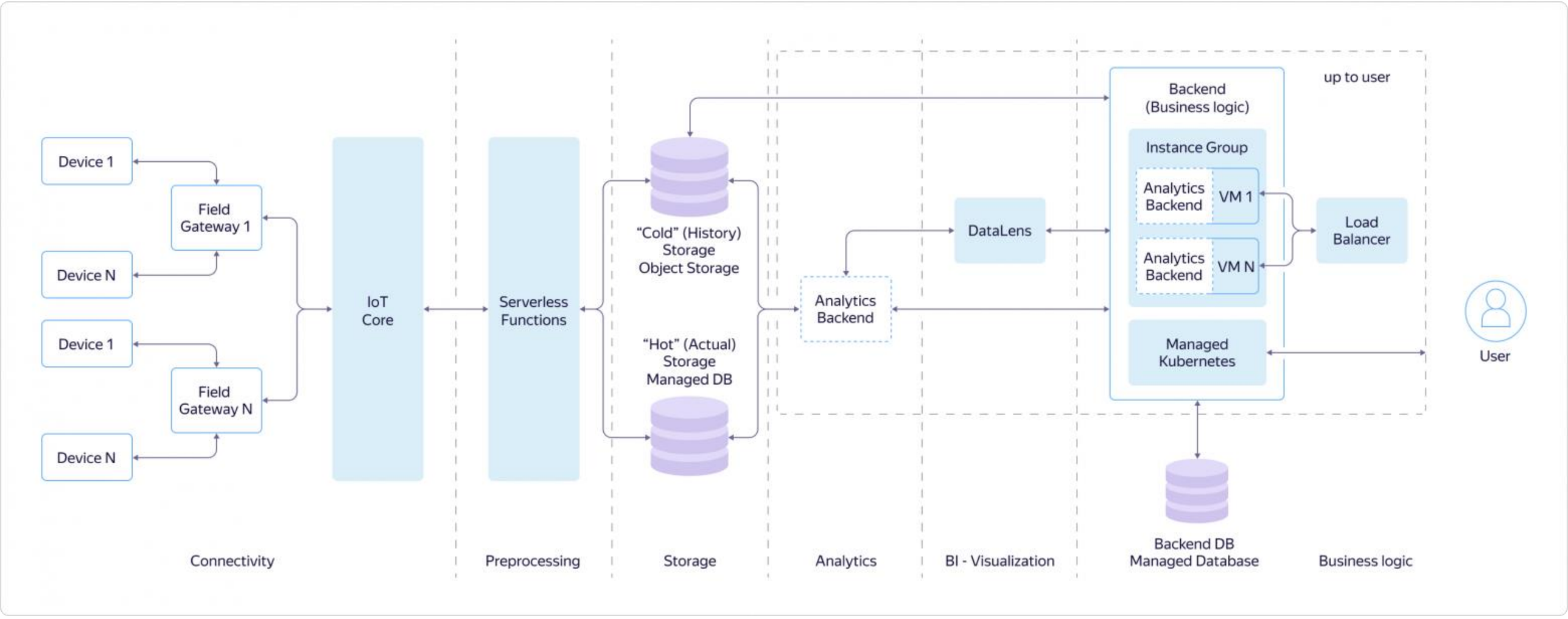

<RunSettings>
<!-- Configurations that affect the Test Framework -->
<RunConfiguration>
<MaxCpuCount>1</MaxCpuCount>
<ResultsDirectory>.\TestResults</ResultsDirectory>
<TestSessionTimeout>120000</TestSessionTimeout>
<TargetFrameworkVersion>net48</TargetFrameworkVersion>
<TargetPlatform>x64</TargetPlatform>
</RunConfiguration>
<nanoFrameworkAdapter>
<Logging>None</Logging>
<IsRealHardware>False</IsRealHardware>
</nanoFrameworkAdapter>
</RunSettings>
133 %
1
0
Стр: 17
Симв: 1
Пробелы
LF
Обозреватель тестов
1
1
0
Поиск (Ctrl+I)
Запуск тестов завершен: тестов запущено в 2 с: 1 (пройдено: 1, не пройдено: 0, пропущено: 0).
1 предупре
Ошибок: 0
Тестирование
Длительнос
Признаки
Сообщение об ошибке
NFUnitTest (1)
< 1 мс
nanoFramework.TestFramework.Test.TestStateService (1)
< 1 мс
TestSetState (1)
< 1 мс
Выполнить
Отладка
Сводка по группе
NFUnitTest
Тесты в группе: 1
Результаты
1 Пройден

Тесты

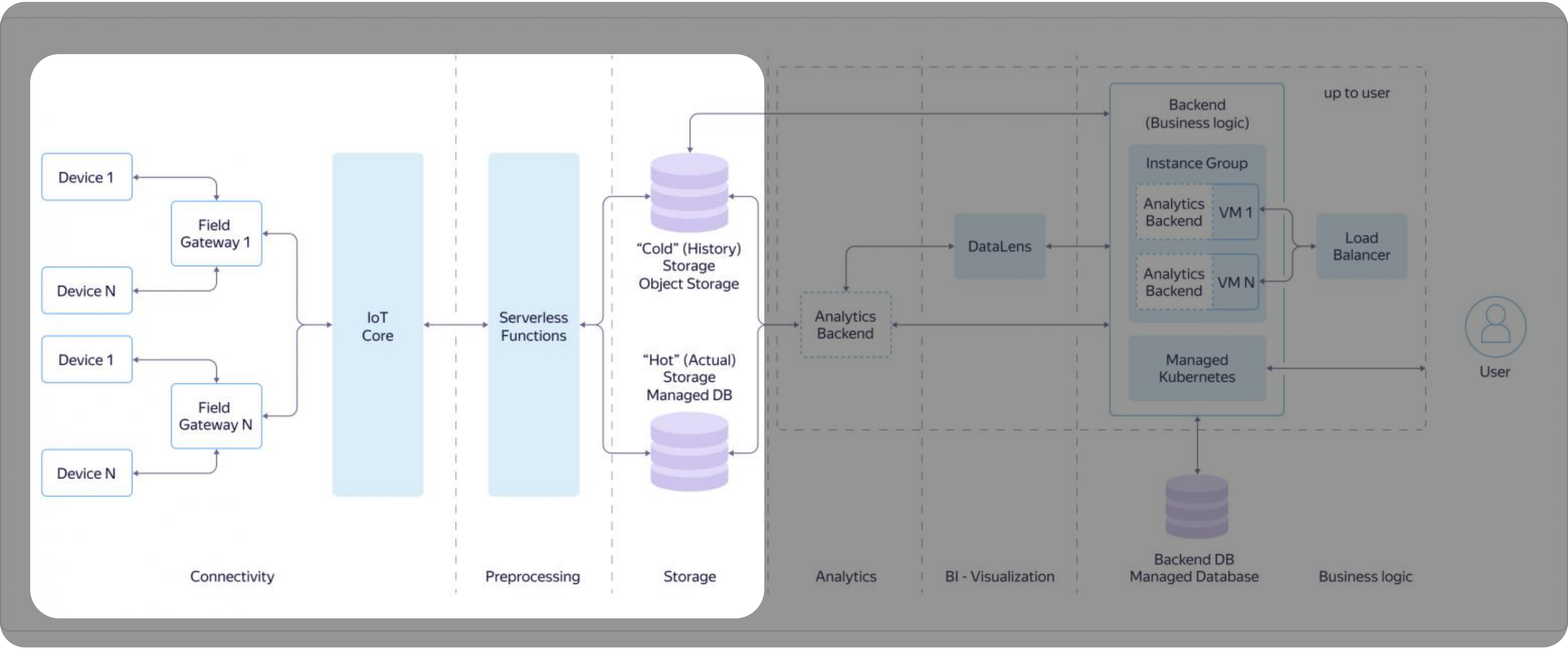

<?xml version="1.0" encoding="utf-8"?>
<RunSettings>
<!-- Configurations that affect the Test Framework -->
<RunConfiguration>
<MaxCpuCount>1</MaxCpuCount>
<ResultsDirectory>.\TestResults</ResultsDirectory>
<TestSessionTimeout>120000</TestSessionTimeout>
<TargetFrameworkVersion>net48</TargetFrameworkVersion>
<TargetPlatform>x64</TargetPlatform>
</RunConfiguration>
<nanoFrameworkAdapter>
<Logging>None</Logging>
<IsRealHardware>True</IsRealHardware>
</nanoFrameworkAdapter>
</RunSettings>
133 %
Проблемы не найдены.
Стр: 17
Симв: 1
Пробелы
LF
Обозреватель тестов
Поиск (Ctrl+I)
Запуск тестов завершен: тестов запущено в 13.3 с: 1 (пройдено: 1, не пройдено: 0, пропущено: 0).
Ошибок: 0
Тестирование
Длительн...
Признаки
Сообщение об ошибке
NFUnitTest (1)
2 мс
nanoFramework.TestFramework.Test.TestStateService (1)
2 мс
TestSetState (1)
2 мс
Выполнить
Отладка
Сводка по группе
NFUnitTest
Тесты в группе: 1
Общая длительность: 2 мс
Результаты
1 Пройден

IoT*

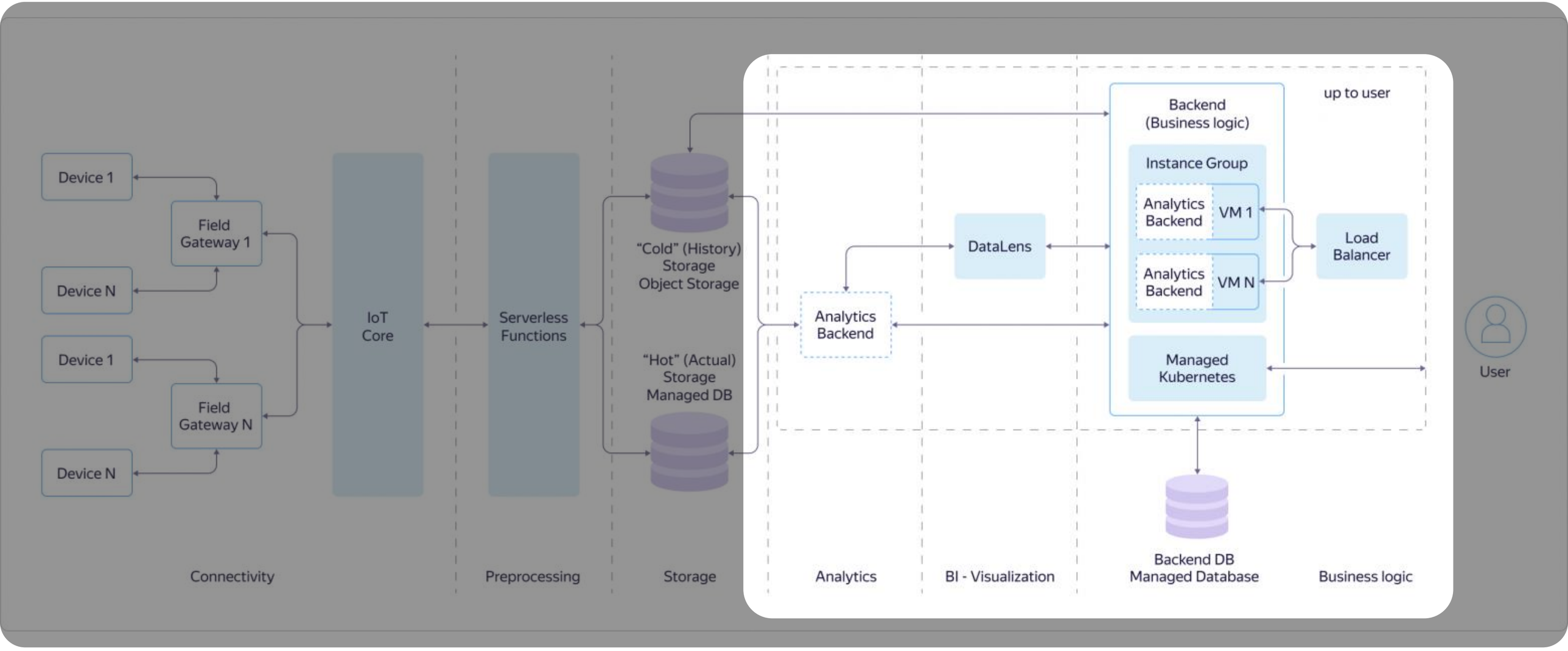
IoT


Device 1
Field Gateway 1
Device N
Device 1
Field Gateway N
Device N
IoT Core
Serverless Functions
"Cold" (History) Storage Object Storage
"Hot" (Actual) Storage Managed DB
Analytics Backend
DataLens
Backend (Business logic)
up to user
Instance Group
Analytics Backend
VM 1
Analytics Backend
VM N
Load Balancer
Managed Kubernetes
User
Backend DB Managed Database
Connectivity
Preprocessing
Storage
Analytics
BI - Visualization
Business logic

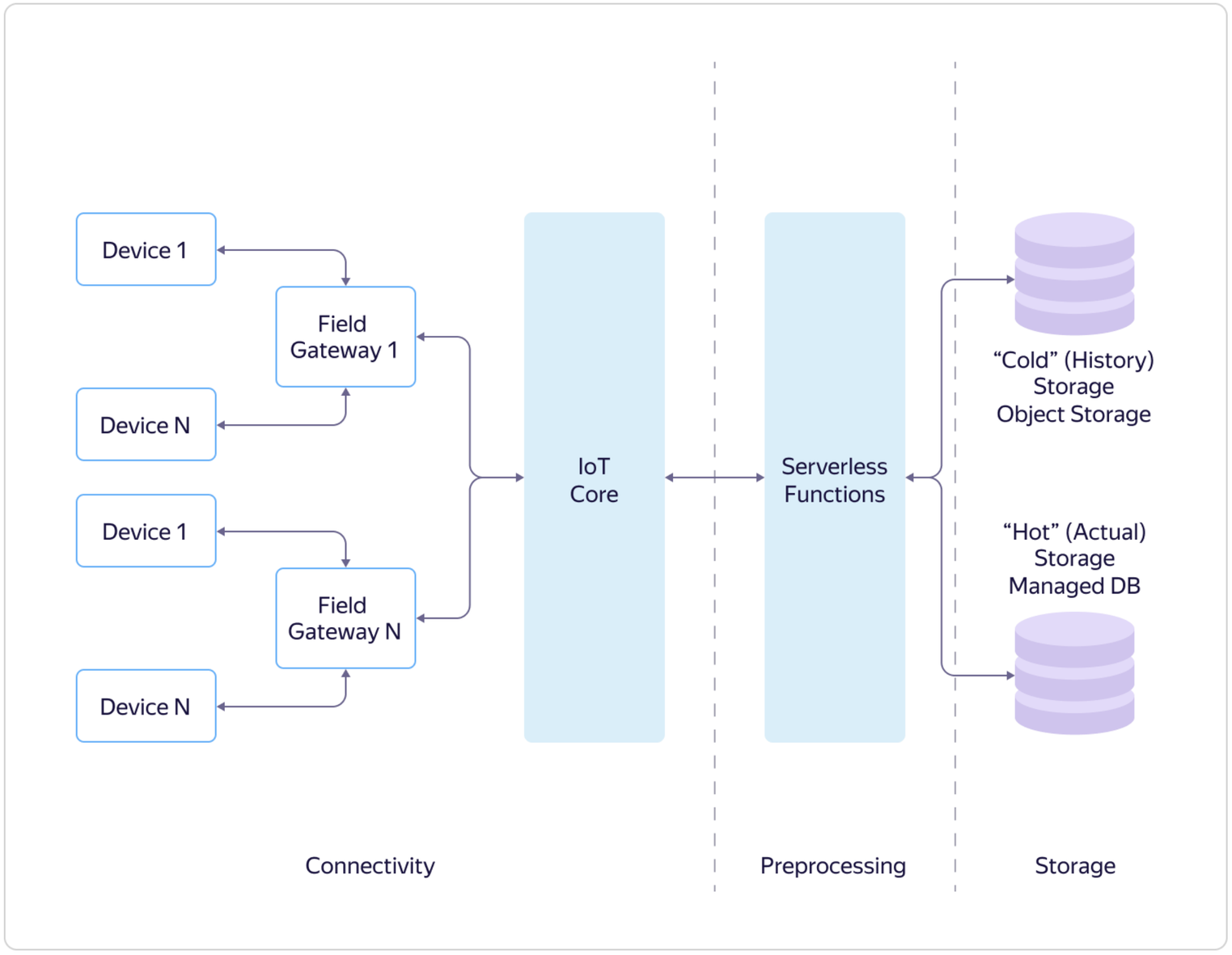
IoT


Device 1
Field Gateway 1
Device N
Device 1
Field Gateway N
Device N
IoT Core
Serverless Functions
"Cold" (History) Storage Object Storage
"Hot" (Actual) Storage Managed DB
Analytics Backend
DataLens
Backend (Business logic)
Instance Group
Analytics Backend
VM 1
Analytics Backend
VM N
Load Balancer
Managed Kubernetes
up to user
User
Backend DB Managed Database
Connectivity
Preprocessing
Storage
Analytics
BI - Visualization
Business logic

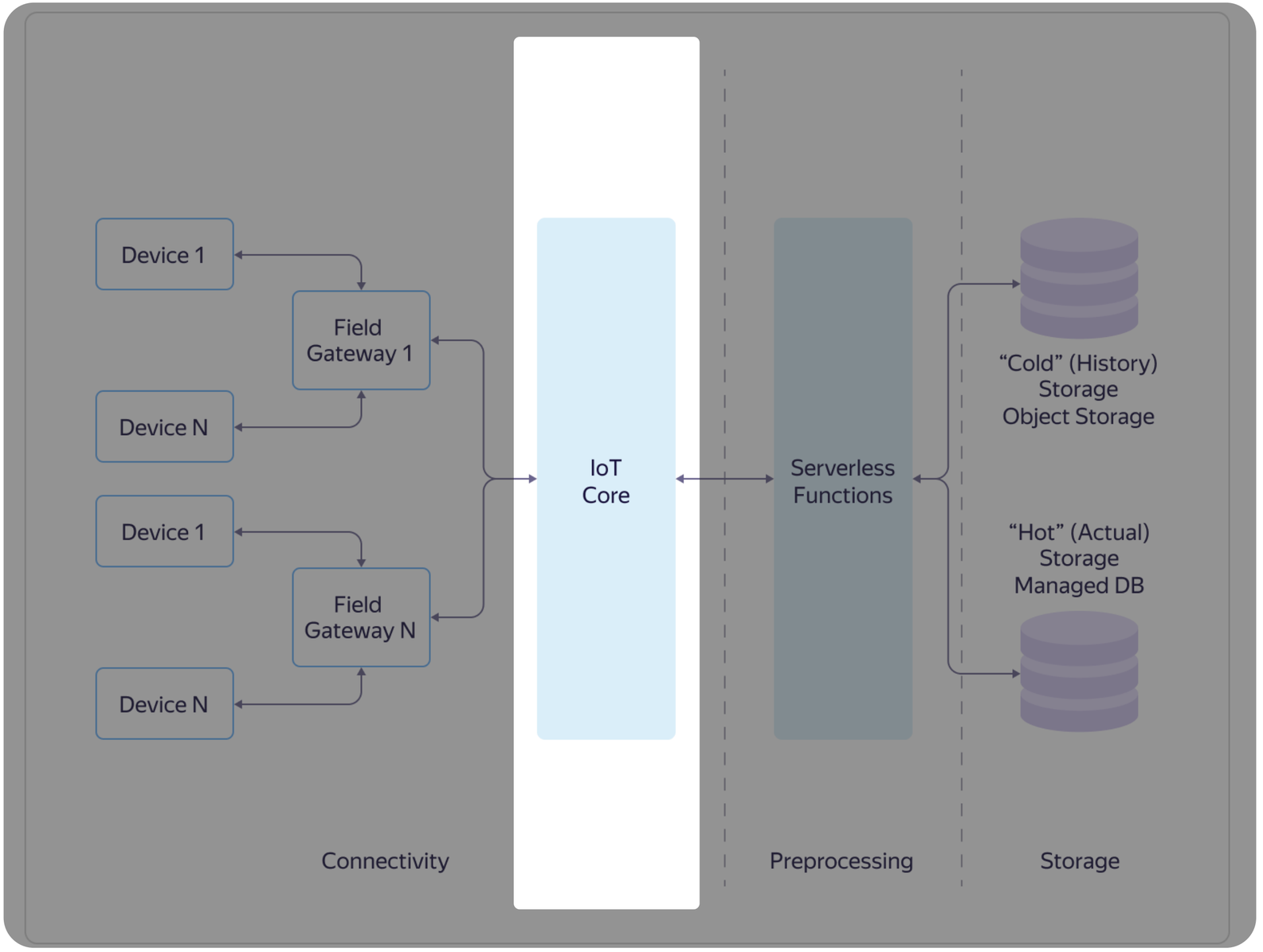
IoT


Device 1
Field Gateway 1
Device N
Device 1
Field Gateway N
Device N
IoT Core
Serverless Functions
"Cold" (History) Storage Object Storage
"Hot" (Actual) Storage Managed DB
Analytics Backend
DataLens
Backend (Business logic)
Instance Group
Analytics Backend
VM 1
Analytics Backend
VM N
Load Balancer
Managed Kubernetes
up to user
User
Backend DB Managed Database
Connectivity
Preprocessing
Storage
Analytics
BI - Visualization
Business logic

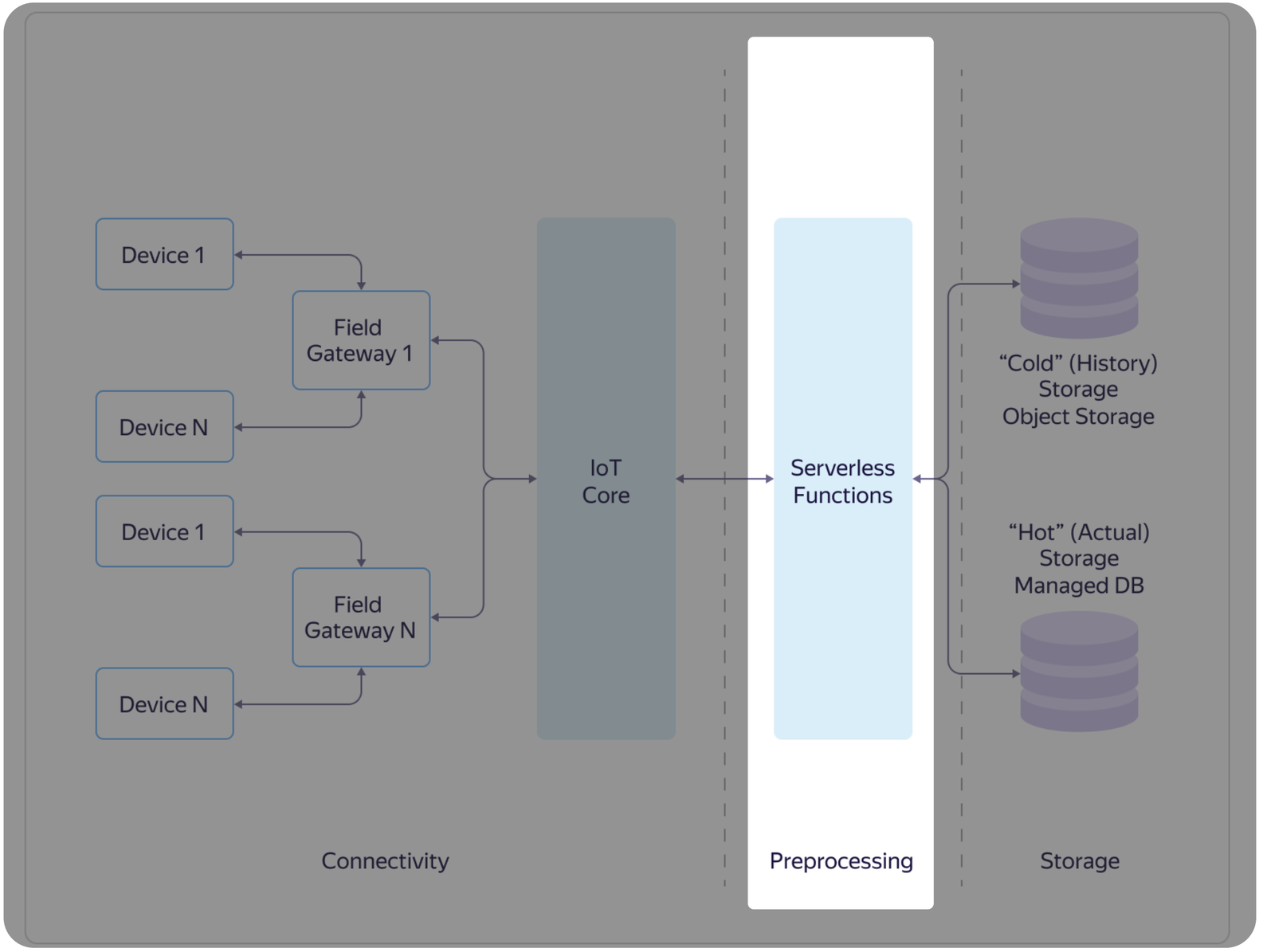
IoT


Device 1
Field Gateway 1
Device N
Device 1
Field Gateway N
Device N
IoT Core
Serverless Functions
“Cold” (History) Storage Object Storage
“Hot” (Actual) Storage Managed DB
Connectivity
Preprocessing
Storage

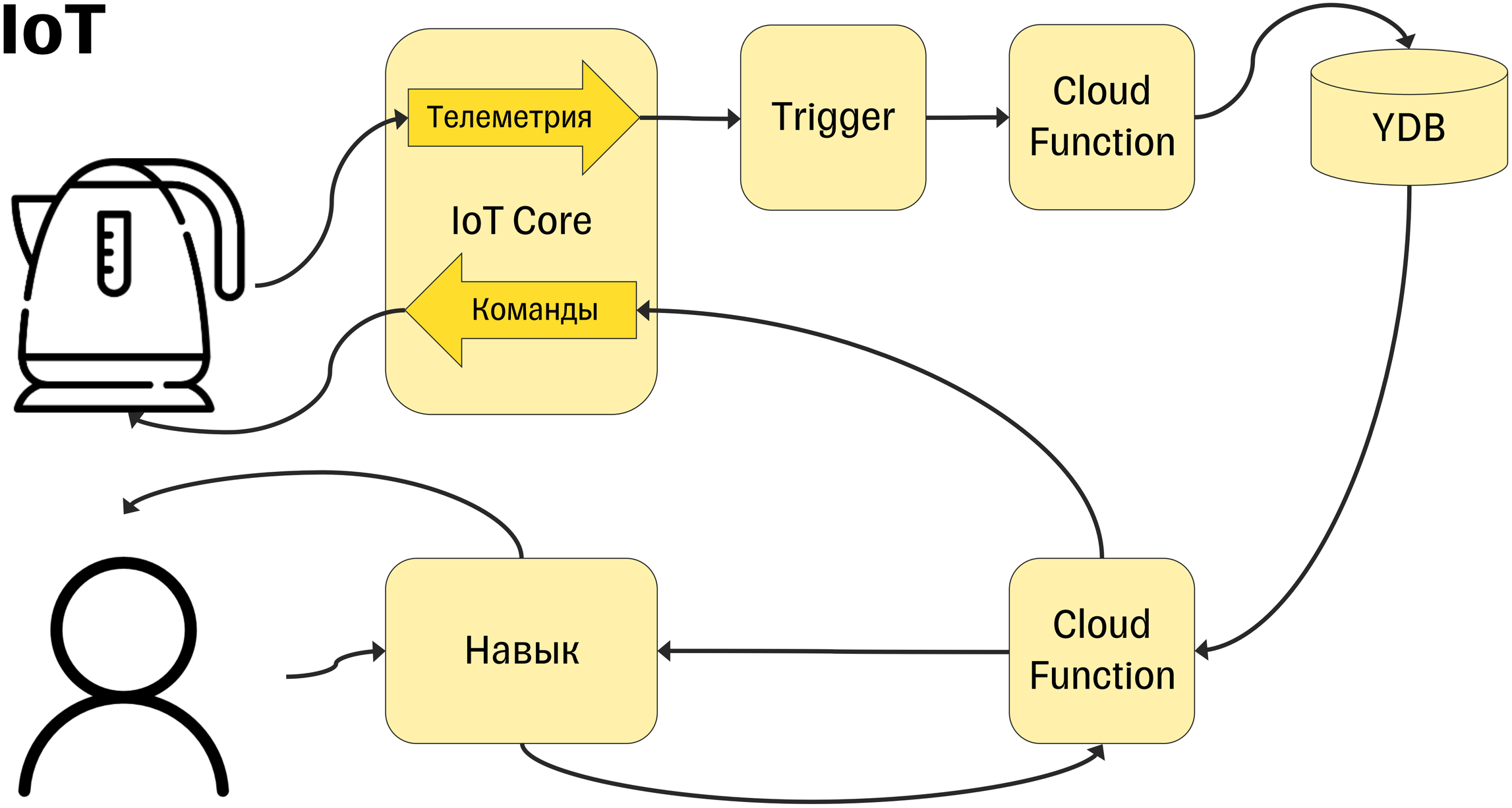
IoT


Device 1
Field Gateway 1
Device N
Device 1
Field Gateway N
Device N
IoT Core
Serverless Functions
“Cold” (History) Storage Object Storage
“Hot” (Actual) Storage Managed DB
Connectivity
Preprocessing
Storage

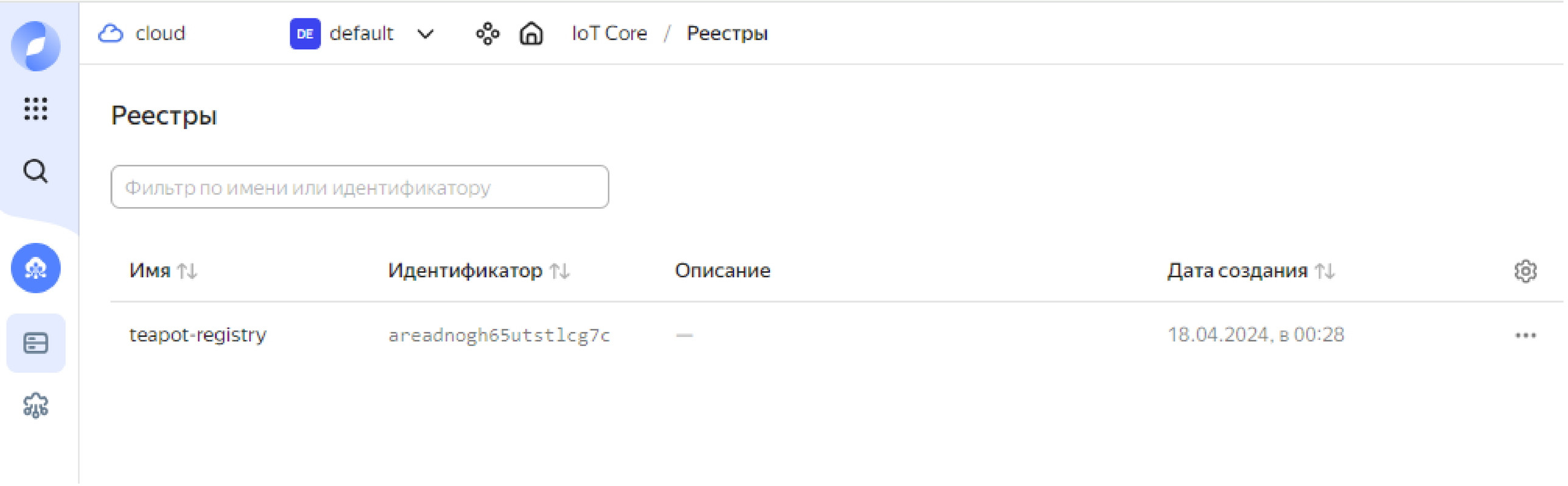
IoT


Device 1
Field Gateway 1
Device N
Device 1
Field Gateway N
Device N
IoT Core
Serverless Functions
“Cold” (History) Storage Object Storage
“Hot” (Actual) Storage Managed DB
Connectivity
Preprocessing
Storage

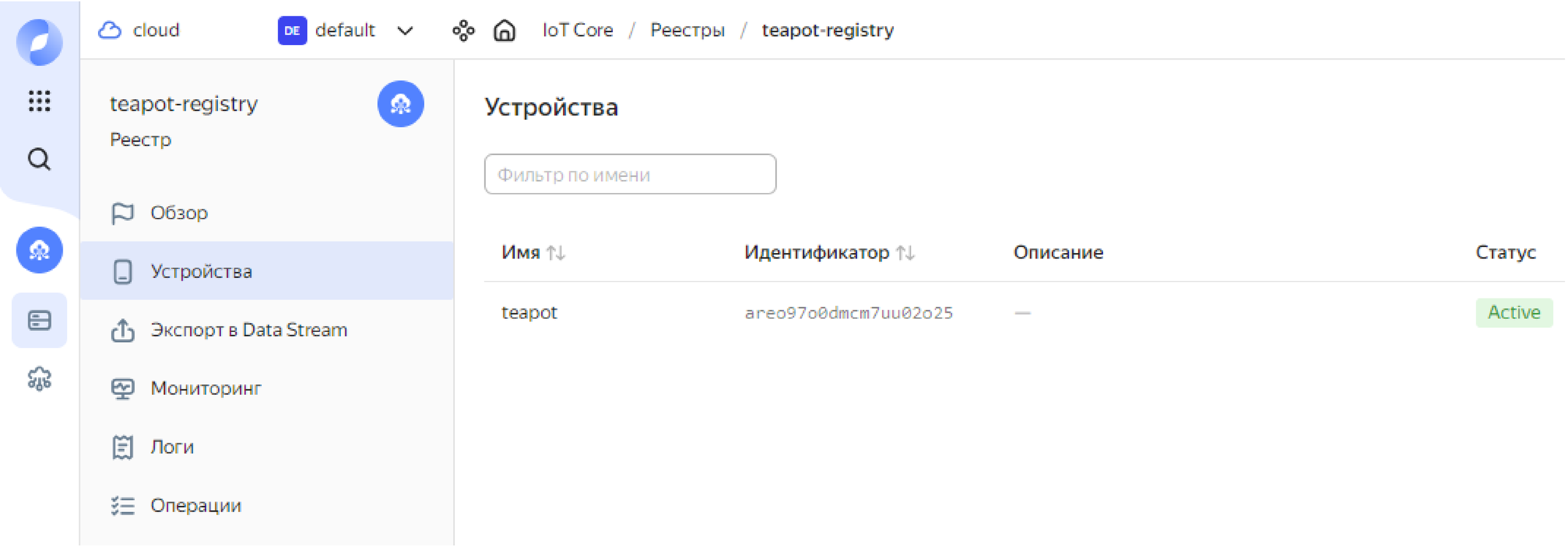
IoT


Телеметрия
Trigger
Cloud Function
YDB
IoT Core
Команды
Навык
Cloud Function

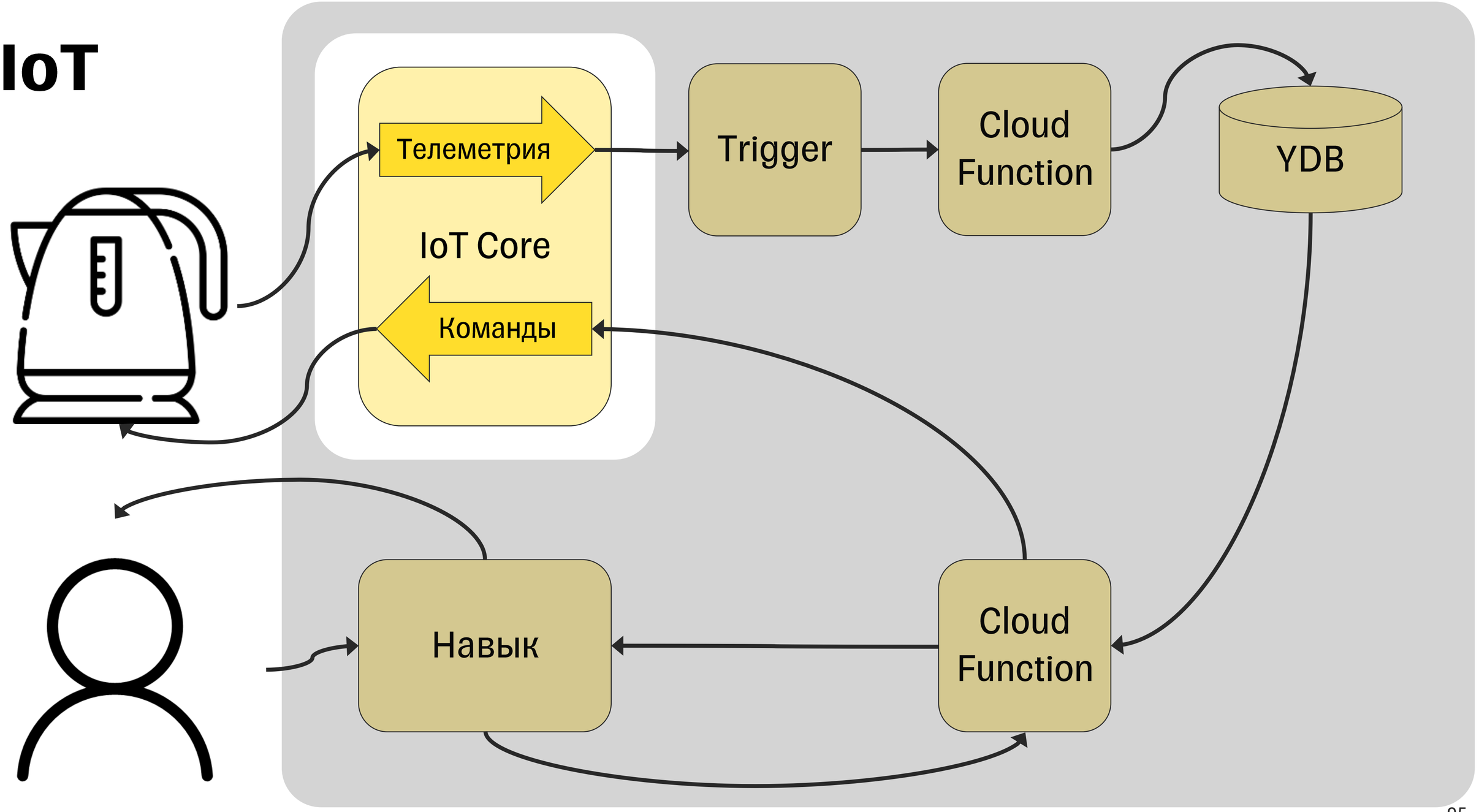
IoT


cloud
DE default
IoT Core / Реестры
Реестры
Фильтр по имени или идентификатору
Имя
Идентификатор
Описание
Дата создания
teapot-registry
areadnogh65utstlcg7c
—
18.04.2024, в 00:28

IoT


cloud
DE default
IoT Core / Реестры / teapot-registry
teapot-registry
Реестр
Обзор
Устройства
Экспорт в Data Stream
Мониторинг
Логи
Операции
Устройства
Фильтр по имени
Имя
Идентификатор
Описание
Статус
teapot
areo97o0dmcm7uu02o25
—
Active

IoT


Телеметрия
Trigger
Cloud Function
YDB
IoT Core
Команды
Навык
Cloud Function

IoT

Топик	Перманентный
$devices/<идентификатор_устройства>/events	
$devices/<идентификатор_устройства>/state	✔

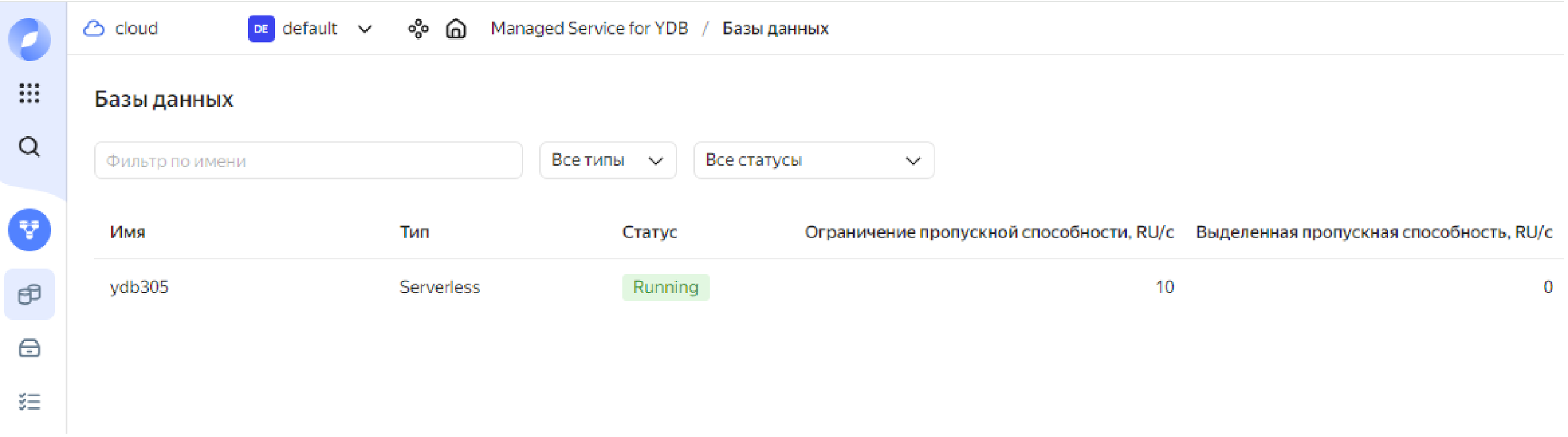
IoT

Топик	Перманентный
$devices/<идентификатор_устройства>/events	
$devices/<идентификатор_устройства>/state	✔
$devices/<идентификатор_устройства>/commands	
$devices/<идентификатор_устройства>/config	✔

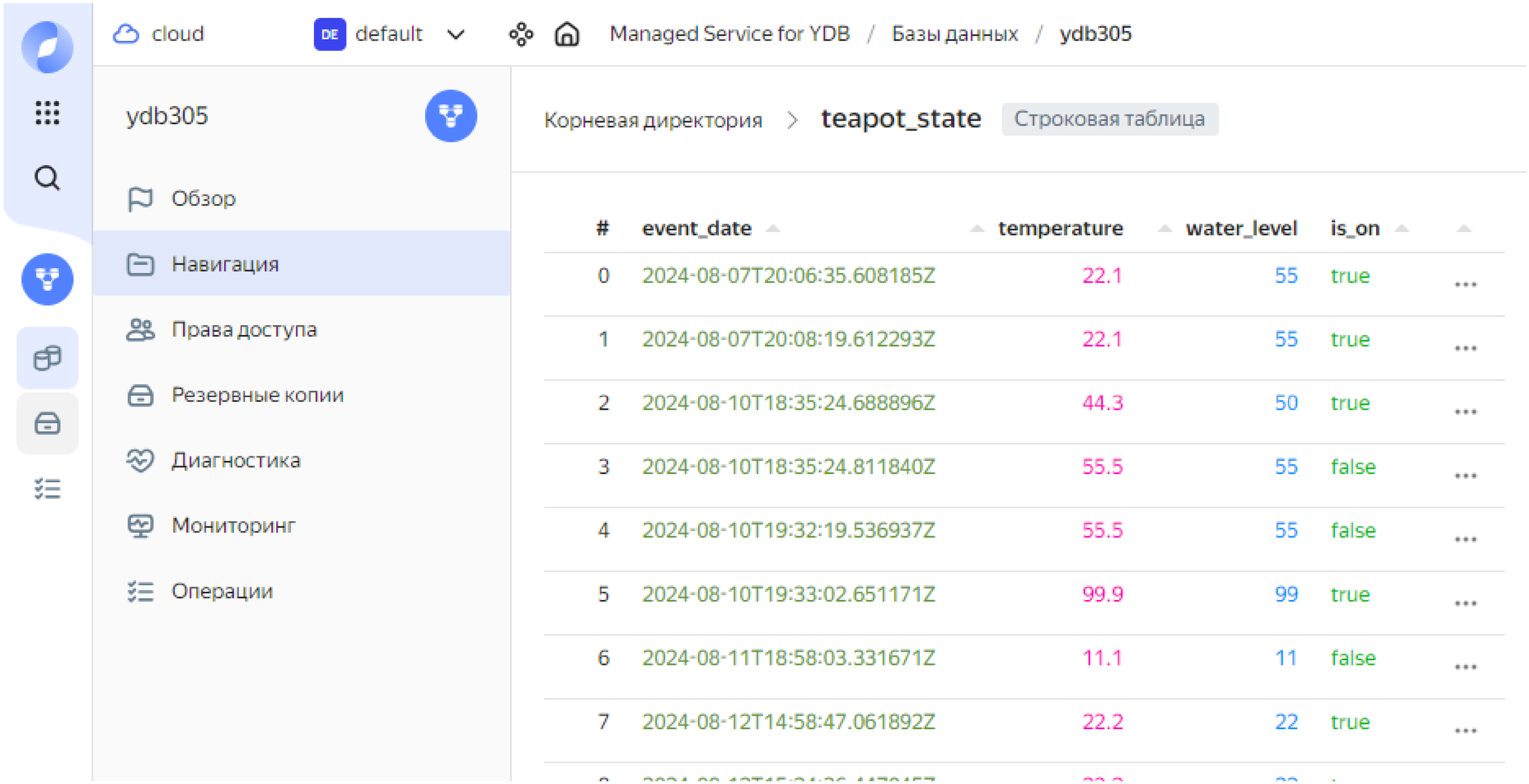
IoT

Топик	Перманентный
$devices/<идентификатор_устройства>/events	
$devices/<идентификатор_устройства>/state	✔
$devices/<идентификатор_устройства>/commands	
$devices/<идентификатор_устройства>/config	✔
$monitoring/<идентификатор_устройства>/json	

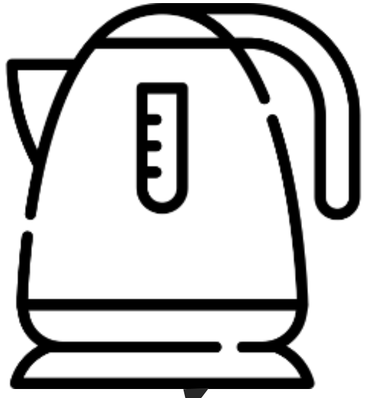
IoT


cloud
DE
default
Managed Service for YDB / Базы данных
Базы данных
Фильтр по имени
Все типы
Все статусы
Имя
Тип
Статус
Ограничение пропускной способности, RU/с
Выделенная пропускная способность, RU/с
ydb305
Serverless
Running
10
0

IoT


cloud
DE default
Managed Service for YDB / Базы данных / ydb305
ydb305
Обзор
Навигация
Права доступа
Резервные копии
Диагностика
Мониторинг
Операции
Корневая директория > teapot_state
Строковая таблица
event_date temperature water_level is_on
0 2024-08-07T20:06:35.608185Z 22.1 55 true
1 2024-08-07T20:08:19.612293Z 22.1 55 true
2 2024-08-10T18:35:24.688896Z 44.3 50 true
3 2024-08-10T18:35:24.811840Z 55.5 55 false
4 2024-08-10T19:32:19.536937Z 55.5 55 false
5 2024-08-10T19:33:02.651171Z 99.9 99 true
6 2024-08-11T18:58:03.331671Z 11.1 11 false
7 2024-08-12T14:58:47.061892Z 22.2 22 true

IoT



Телеметрия

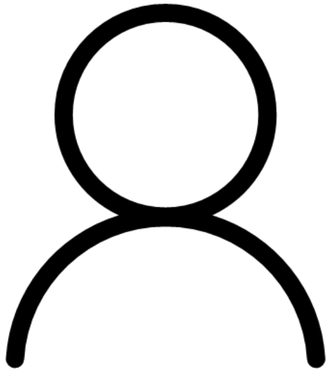
IoT Core

Команды

Trigger

Cloud Function

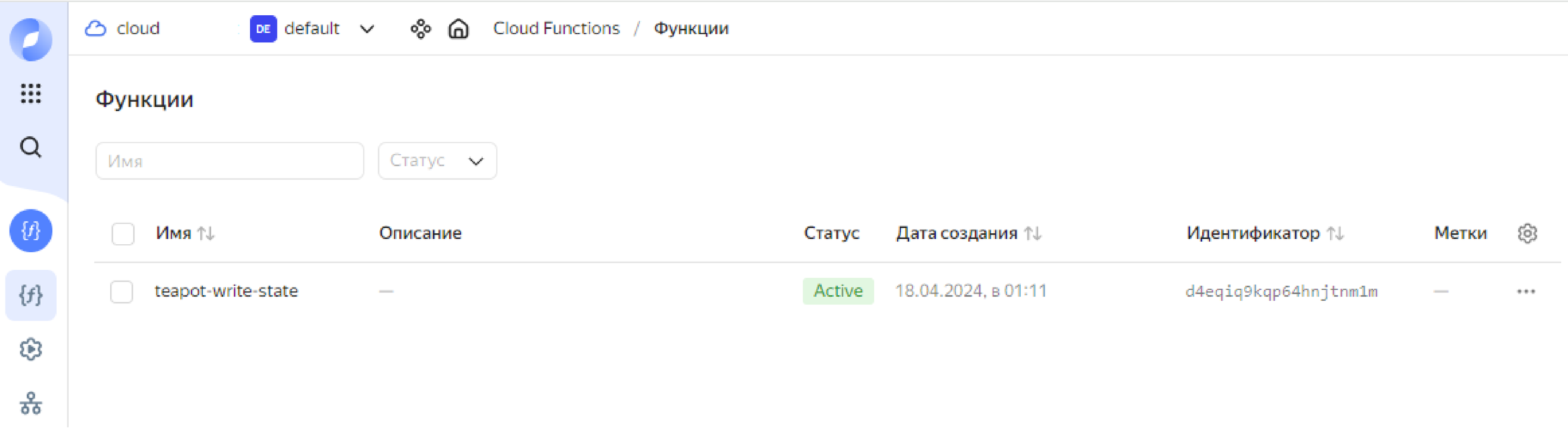
YDB



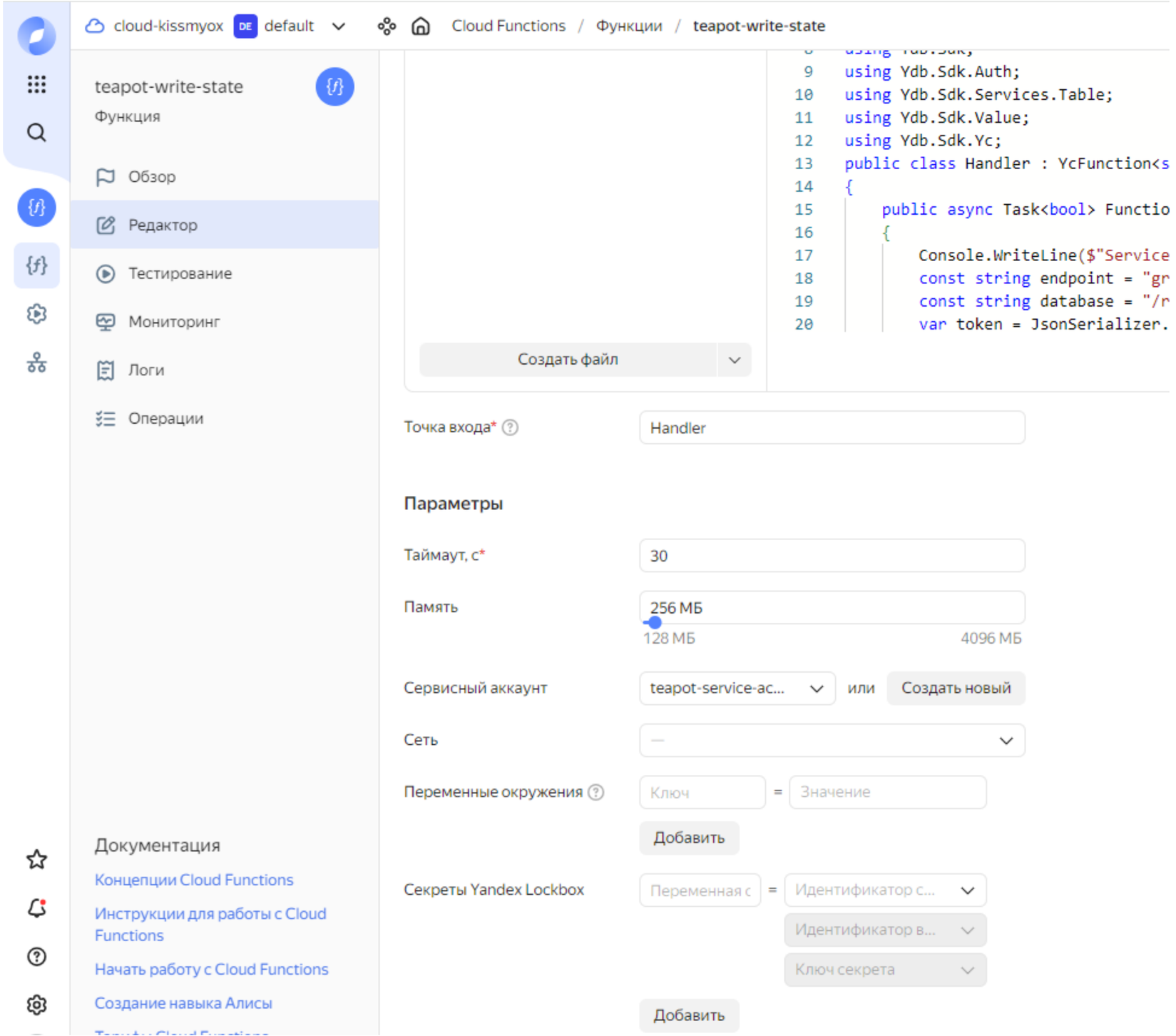
Навык

Cloud Function

IoT


cloud
DE default
Cloud Functions / Функции
Функции
Имя
Статус
Имя
Описание
Статус
Дата создания
Идентификатор
Метки
teapot-write-state
—
Active
18.04.2024, в 01:11
d4eqiq9kqp64hnjtnm1m
—

IoT


cloud-kissmyox
default
Cloud Functions / Функции / teapot-write-state
teapot-write-state
Функция
Обзор
Редактор
Тестирование
Мониторинг
Логи
Операции
Документация
Концепции Cloud Functions
Инструкции для работы с Cloud Functions
Начать работу с Cloud Functions
Создание навыка Алисы
using Ydb.Sdk.Auth;
using Ydb.Sdk.Services.Table;
using Ydb.Sdk.Value;
using Ydb.Sdk.Yc;
public class Handler : YcFunction<s
{
public async Task<bool> Functio
{
Console.WriteLine($"Service
const string endpoint = "gr
const string database = "/r
var token = JsonSerializer.
Создать файл
Точка входа*
Handler
Параметры
Таймаут, с*
30
Память
256 МБ
128 МБ
4096 МБ
Сервисный аккаунт
teapot-service-ac...
или
Создать новый
Сеть
Переменные окружения
Ключ
Значение
Добавить
Секреты Yandex Lockbox
Переменная с
Идентификатор с...
Идентификатор в...
Ключ секрета
Добавить

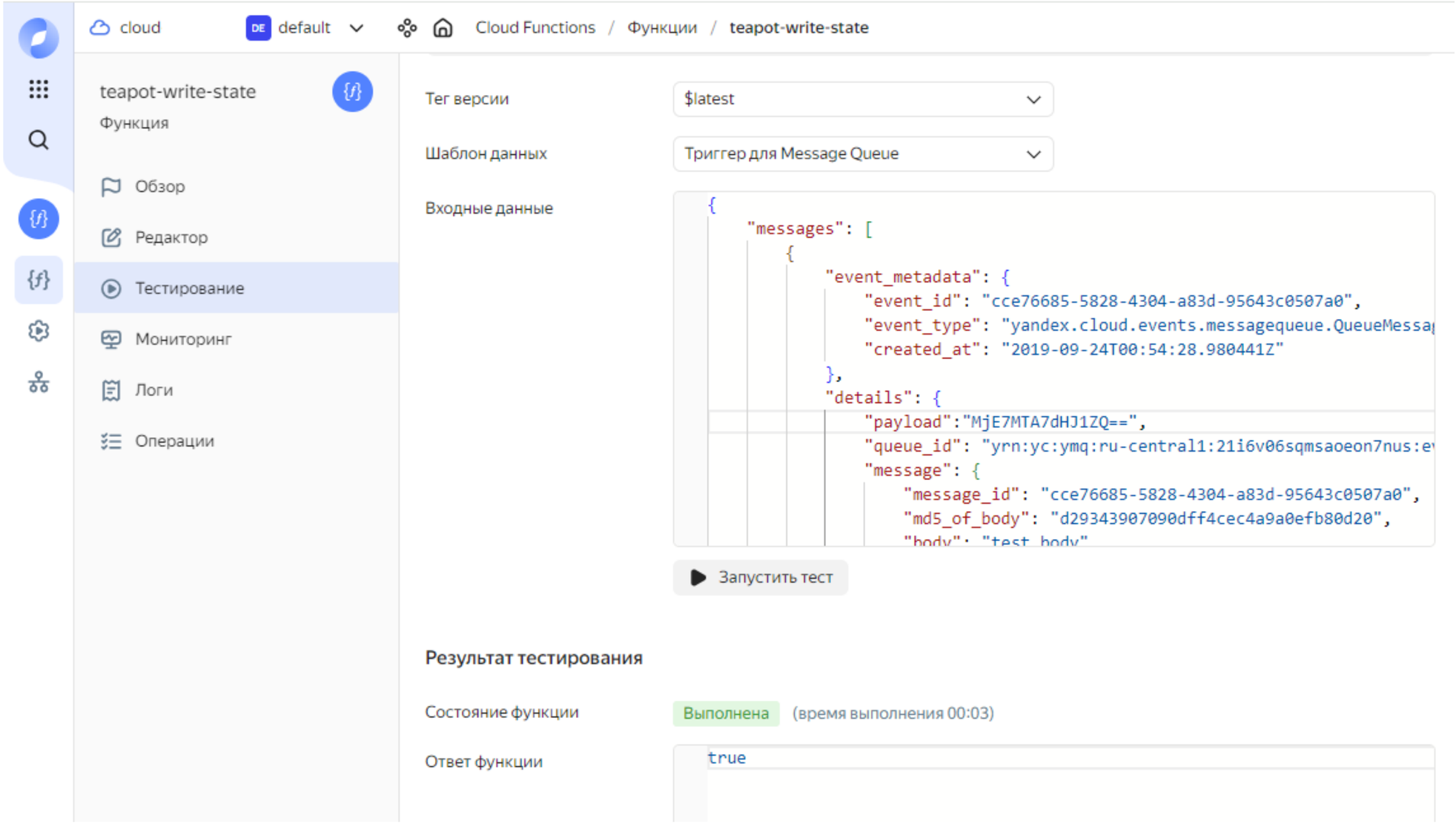
IoT

```
var config = new DriverConfig(
    endpoint: endpoint,
    database: database,
    credentials: new TokenProvider(token)
);

await using var driver = await Driver.CreateInitialized(config);
using var tableClient = new TableClient(driver, new TableClientConfig());

var query = @$"
    DECLARE $event_date AS Timestamp;
    DECLARE $temperature AS Float;
    DECLARE $water_level AS Int32;
    DECLARE $is_on AS Bool;

    UPSERT INTO teapot_state (event_date, temperature, water_level, is_on)
           VALUES ($event_date, $temperature, $water_level, $is_on);";
```

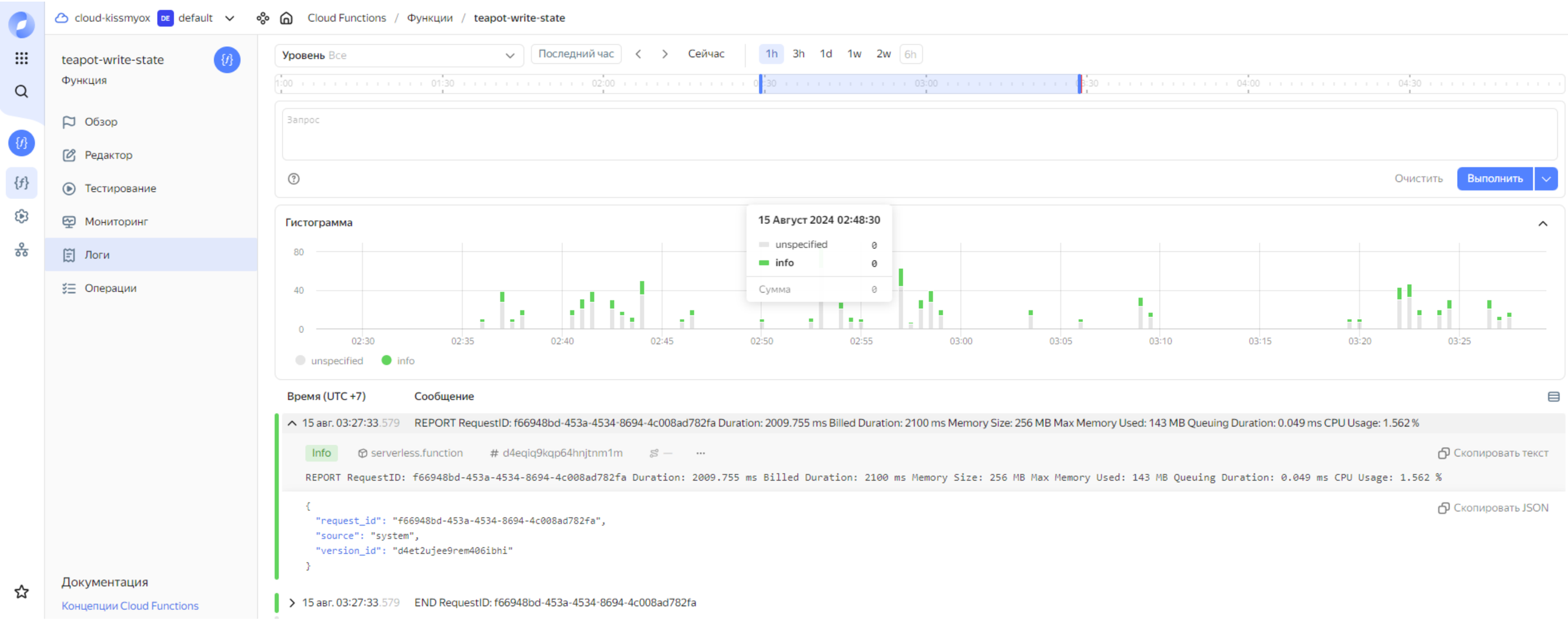
IoT

```
var response = await tableClient.SessionExec(async session =>
{
    return await session.ExecuteDataQuery(
        query: query,
        txControl: TxControl.BeginSerializableRW().Commit(),
        parameters: new Dictionary<string, YdbValue>
        {
            {"$event_date", YdbValue.MakeTimestamp(DateTime.Now)},
            {"$temperature", YdbValue.MakeFloat(temperature)},
            {"$water_level", YdbValue.MakeInt32(waterLevel)},
            {"$is_on", YdbValue.MakeBool(isOn)}
        },
        settings: new ExecuteDataQuerySettings()
        {
            OperationTimeout = TimeSpan.FromSeconds(1),
            TransportTimeout = TimeSpan.FromSeconds(5)
        }
    );
});
response.Status.EnsureSuccess();
```

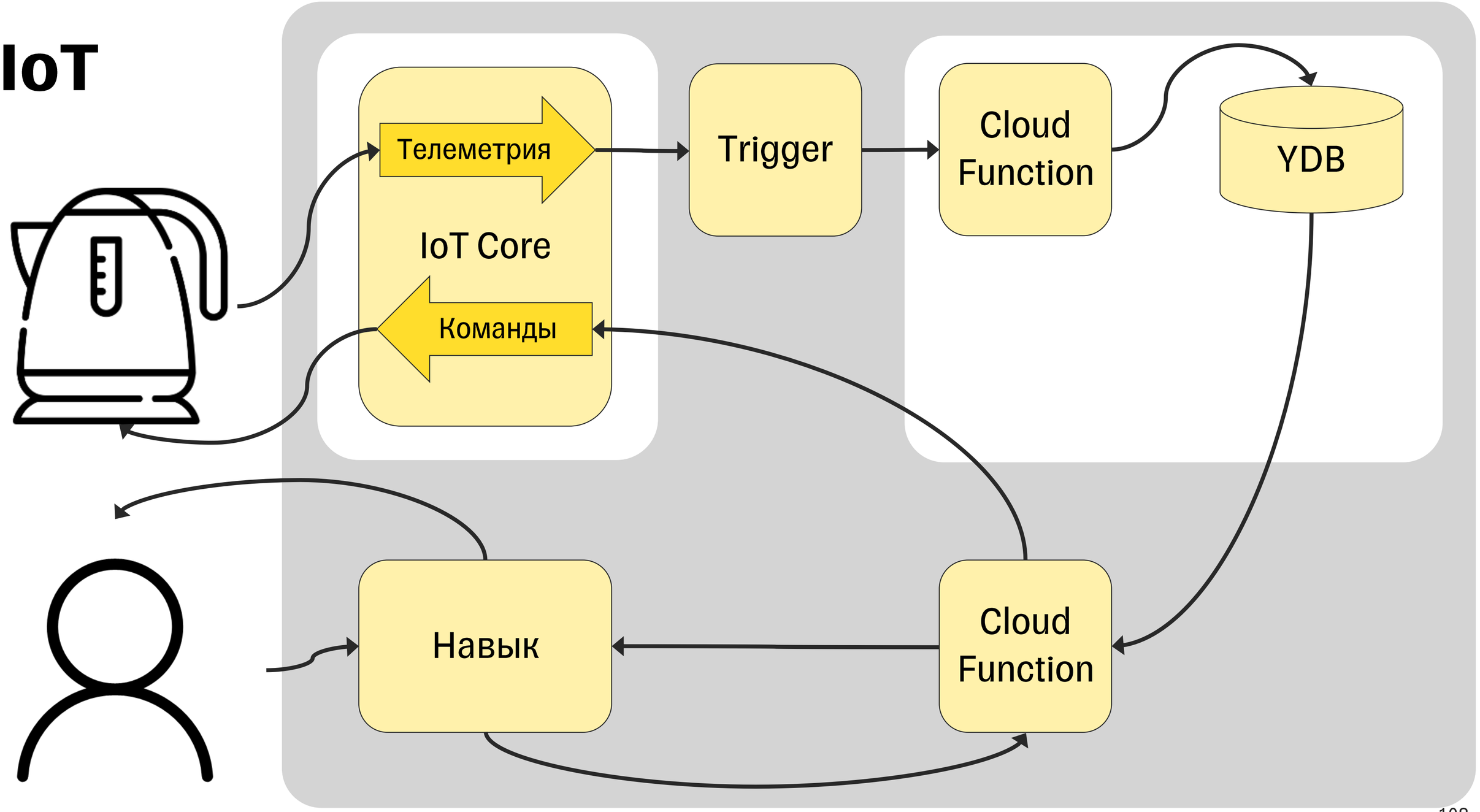
IoT


cloud
default
Cloud Functions / Функции / teapot-write-state
teapot-write-state
Функция
Обзор
Редактор
Тестирование
Мониторинг
Логи
Операции
Тег версии
$latest
Шаблон данных
Триггер для Message Queue
Входные данные
{
"messages": [
{
"event_metadata": {
"event_id": "cce76685-5828-4304-a83d-95643c0507a0",
"event_type": "yandex.cloud.events.messagequeue.QueueMessa
"created_at": "2019-09-24T00:54:28.980441Z"
},
"details": {
"payload":"MjE7MTA7dHJ1ZQ==",
"queue_id": "yrn:yc:ymq:ru-central1:21i6v06sqmsaoeon7nus:e
"message": {
"message_id": "cce76685-5828-4304-a83d-95643c0507a0",
"md5_of_body": "d29343907090dff4cec4a9a0efb80d20",
"body": "test body"
Запустить тест
Результат тестирования
Состояние функции
Выполнена
(время выполнения 00:03)
Ответ функции
true

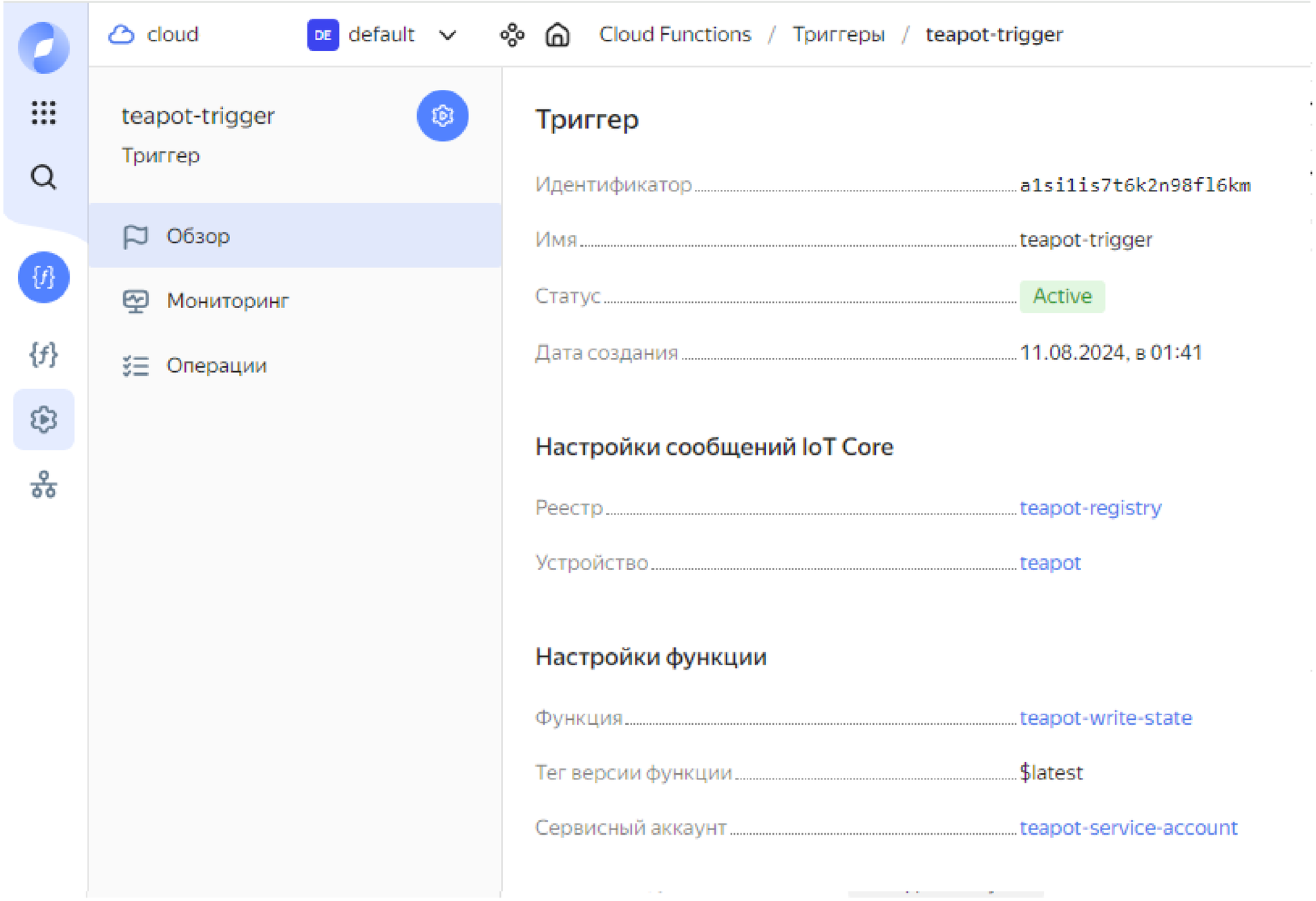
IoT


cloud-kissmyox
default
Cloud Functions / Функции / teapot-write-state
teapot-write-state
Функция
Обзор
Редактор
Тестирование
Мониторинг
Логи
Операции
Документация
Концепции Cloud Functions
Уровень Все
Последний час
Сейчас
1h 3h 1d 1w 2w 6h
Запрос
Очистить
Выполнить
Гистограмма
15 Август 2024 02:48:30
unspecified 0
info 0
Сумма 0
80
40
0
02:30 02:35 02:40 02:45 02:50 02:55 03:00 03:05 03:10 03:15 03:20 03:25
unspecified
info
Время (UTC +7)
Сообщение
15 авг. 03:27:33.579 REPORT RequestID: f66948bd-453a-4534-8694-4c008ad782fa Duration: 2009.755 ms Billed Duration: 2100 ms Memory Size: 256 MB Max Memory Used: 143 MB Queuing Duration: 0.049 ms CPU Usage: 1.562 %
Info
serverless.function
d4eqiq9kqp64hnjtnm1m
Скопировать текст
REPORT RequestID: f66948bd-453a-4534-8694-4c008ad782fa Duration: 2009.755 ms Billed Duration: 2100 ms Memory Size: 256 MB Max Memory Used: 143 MB Queuing Duration: 0.049 ms CPU Usage: 1.562 %
{
"request_id": "f66948bd-453a-4534-8694-4c008ad782fa",
"source": "system",
"version_id": "d4et2ujee9rem406ibhi"
}
Скопировать JSON
15 авг. 03:27:33.579 END RequestID: f66948bd-453a-4534-8694-4c008ad782fa

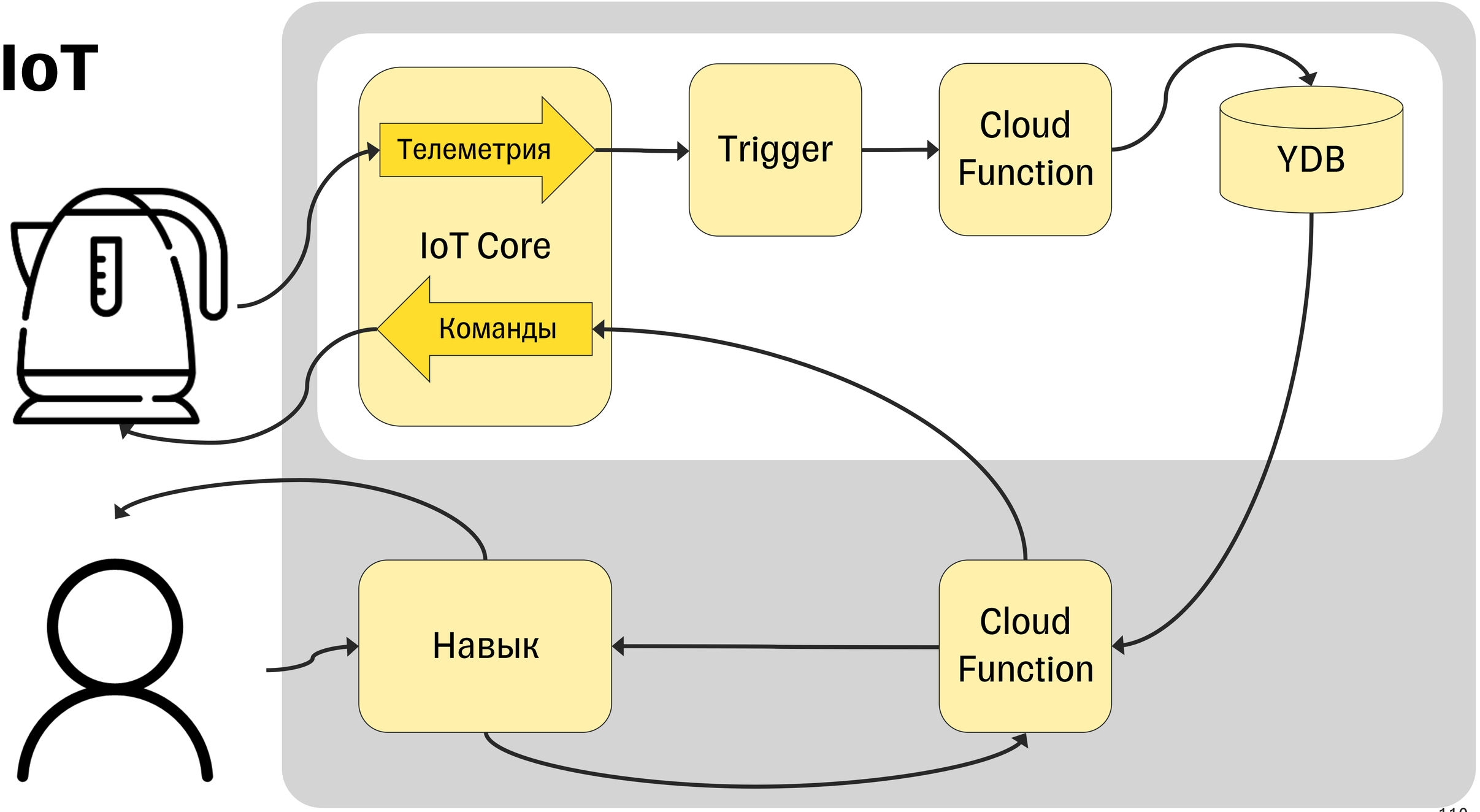
IoT


Телеметрия
Trigger
Cloud Function
YDB
IoT Core
Команды
Навык
Cloud Function

IoT


cloud
DE default
Cloud Functions / Триггеры / teapot-trigger
teapot-trigger
Триггер
Обзор
Мониторинг
Операции
Триггер
Идентификатор a1si1is7t6k2n98fl6km
Имя teapot-trigger
Статус Active
Дата создания 11.08.2024, в 01:41
Настройки сообщений IoT Core
Реестр teapot-registry
Устройство teapot
Настройки функции
Функция teapot-write-state
Тег версии функции $latest
Сервисный аккаунт teapot-service-account

IoT


Телеметрия
Trigger
Cloud Function
YDB
IoT Core
Команды
Навык
Cloud Function

IoT

Яндекс Диалоги

KissMyAxe

Премия Алисы: победители весеннего этапа Подробнее

Чайник дотнет

Диалог опубликован | Черновик в разработке

Общие сведения | Настройки | Связка аккаунтов | Ресурсы | Тестирование | Оценки пользователей | Продвижение | Мониторинг | Донаты | Доступ

Настройки

Черновик | Опубликованная версия

Главные настройки | Интенты

Основные настройки

Имя навыка *: Чайник дотнет

Название, которое будет отображаться в каталоге Алисы. С его помощью пользователь сможет активировать навык, например «Запусти навык "Имя навыка"».

Активационные имена: Чайник дотнет

мой чайник –

+

Здесь можно указать разные словоформы имени навыка или уточнить его произношение. Например, чтобы Алиса корректно распознавала навык «Изучаем C++», добавьте фразу «Изучаем си плюс плюс».

Backend *:

- Webhook URL
- Функция в Яндекс Облаке

cloud-kissmyox/default/teapot-get-state

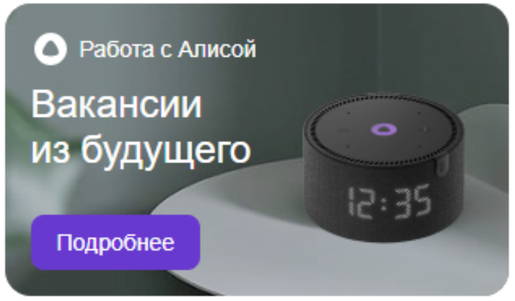
Функции Cloud Functions, которые вы используете для навыков Алисы

1 **Черновик в разработке**

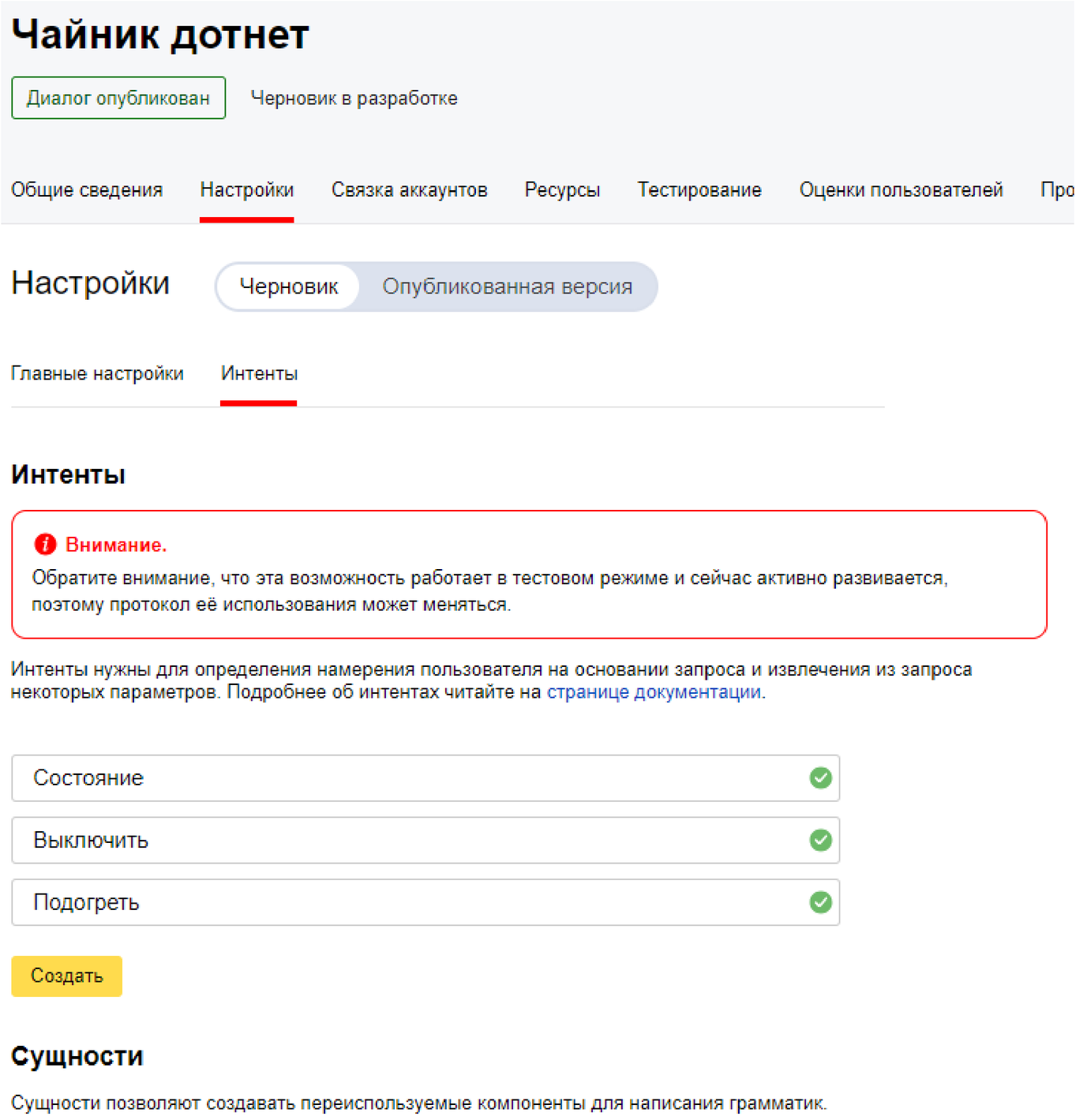
Можно менять его настройки.

Опубликовать

2 **Черновик публикуется**



IoT


Чайник дотнет
Диалог опубликован
Черновик в разработке
Общие сведения
Настройки
Связка аккаунтов
Ресурсы
Тестирование
Оценки пользователей
Про
Настройки
Черновик
Опубликованная версия
Главные настройки
Интенты
Интенты
Внимание.
Обратите внимание, что эта возможность работает в тестовом режиме и сейчас активно развивается, поэтому протокол её использования может меняться.
Интенты нужны для определения намерения пользователя на основании запроса и извлечения из запроса некоторых параметров. Подробнее об интентах читайте на странице документации.
Состояние
Выключить
Подогреть
Создать
Сущности
Сущности позволяют создавать переиспользуемые компоненты для написания грамматик.

IoT

Настройки

Название *

Подогреть

ID *

TurnOn

Грамматика

```
slots:
    temp:
        source: $Temp
        type: YANDEX.NUMBER
root:
    %lemma
    нагрей воду до $Temp градусов
$Temp:
    $YANDEX.NUMBER
```

Положительные тесты

нагрей воду до 43 градусов
нагрей воду до 41 градуса

Отрицательные тесты

подогрей воду до 33 градусов
согрей воду до 33 градусов

Результаты тестирования

Точность: 100%
Полнота: 100%

Положительные тесты:

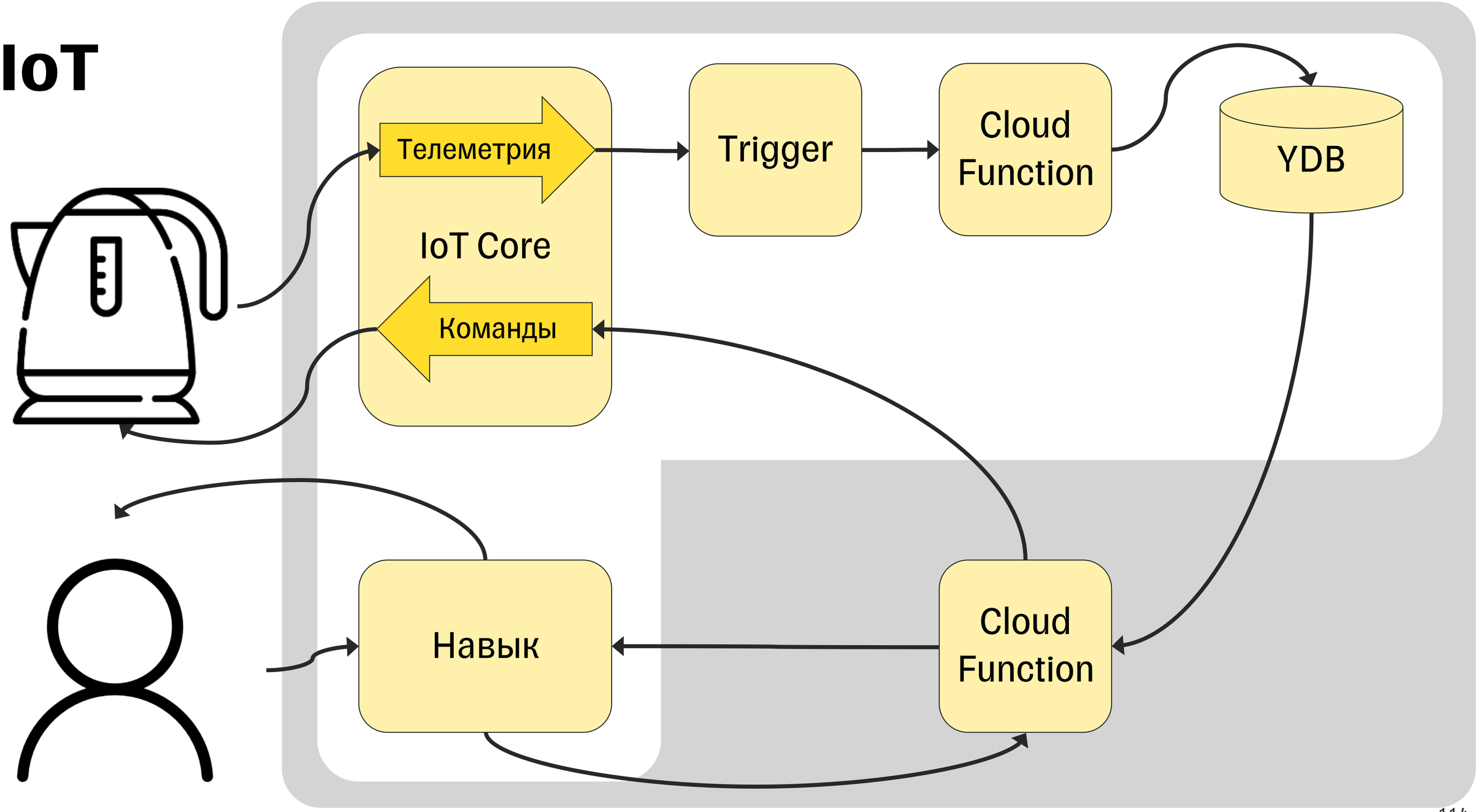
- нагрей воду до 43 градусов
- нагрей воду до 41 градуса

Отрицательные тесты:

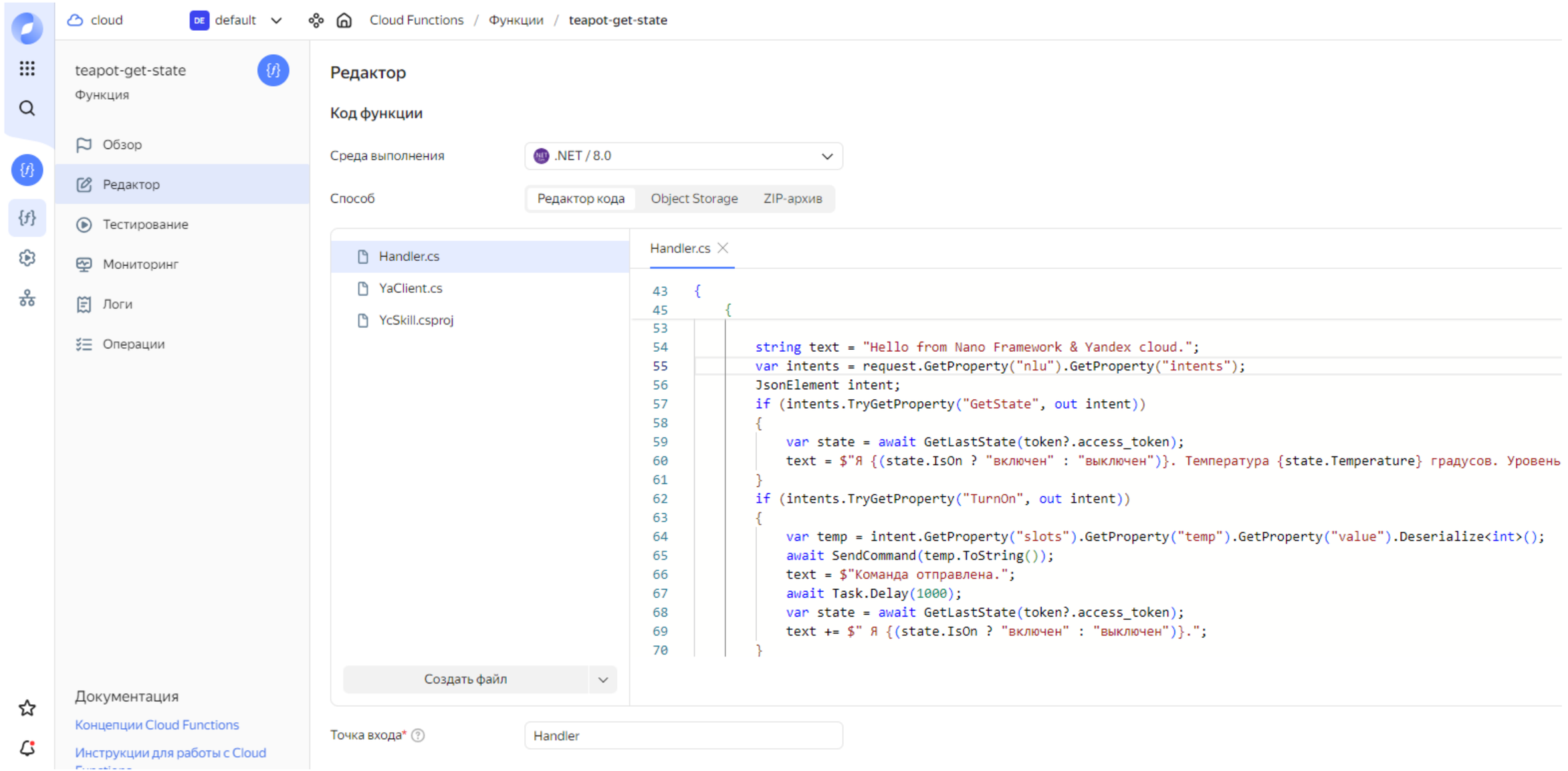
- подогрей чайник до 33 градусов
- согрей чайник до 33 градусов

Сохранить | Протестировать | Собрать запросы β | Удалить

IoT


Телеметрия
Trigger
Cloud Function
YDB
IoT Core
Команды
Навык
Cloud Function

IoT


cloud
default
Cloud Functions / Функции / teapot-get-state
teapot-get-state
Функция
Обзор
Редактор
Тестирование
Мониторинг
Логи
Операции
Документация
Концепции Cloud Functions
Инструкции для работы с Cloud
Редактор
Код функции
Среда выполнения
.NET / 8.0
Способ
Редактор кода
Object Storage
ZIP-архив
Handler.cs
YaClient.cs
YcSkill.csproj
Создать файл
Handler.cs
43 {
45 {
53
54 string text = "Hello from Nano Framework & Yandex cloud.";
55 var intents = request.GetProperty("nlu").GetProperty("intents");
56 JsonElement intent;
57 if (intents.TryGetProperty("GetState", out intent))
58 {
59 var state = await GetLastState(token?.access_token);
60 text = $"Я {(state.IsOn ? "включен" : "выключен")}. Температура {state.Temperature} градусов. Уровень
61 }
62 if (intents.TryGetProperty("TurnOn", out intent))
63 {
64 var temp = intent.GetProperty("slots").GetProperty("temp").GetProperty("value").Deserialize<int>();
65 await SendCommand(temp.ToString());
66 text = $"Команда отправлена.";
67 await Task.Delay(1000);
68 var state = await GetLastState(token?.access_token);
69 text += $" Я {(state.IsOn ? "включен" : "выключен")}.";
70 }
Точка входа*
Handler

IoT

```
var response = await tableClient.SessionExec(async session =>
{
    var query = @"SELECT event_date, temperature, water_level, is_on
                  FROM teapot_state
                  ORDER BY event_date DESC
                  LIMIT 1;";

    return await session.ExecuteDataQuery(
        query: query,
        txControl: TxControl.BeginSerializableRW().Commit());
});
response.Status.EnsureSuccess();
var queryResponse = (ExecuteDataQueryResponse) response;
var resultSet = queryResponse.Result.ResultSets[0];

var temp = (float?)resultSet.Rows[0][1];

Console.WriteLine($"Temperature: {temp}");
```

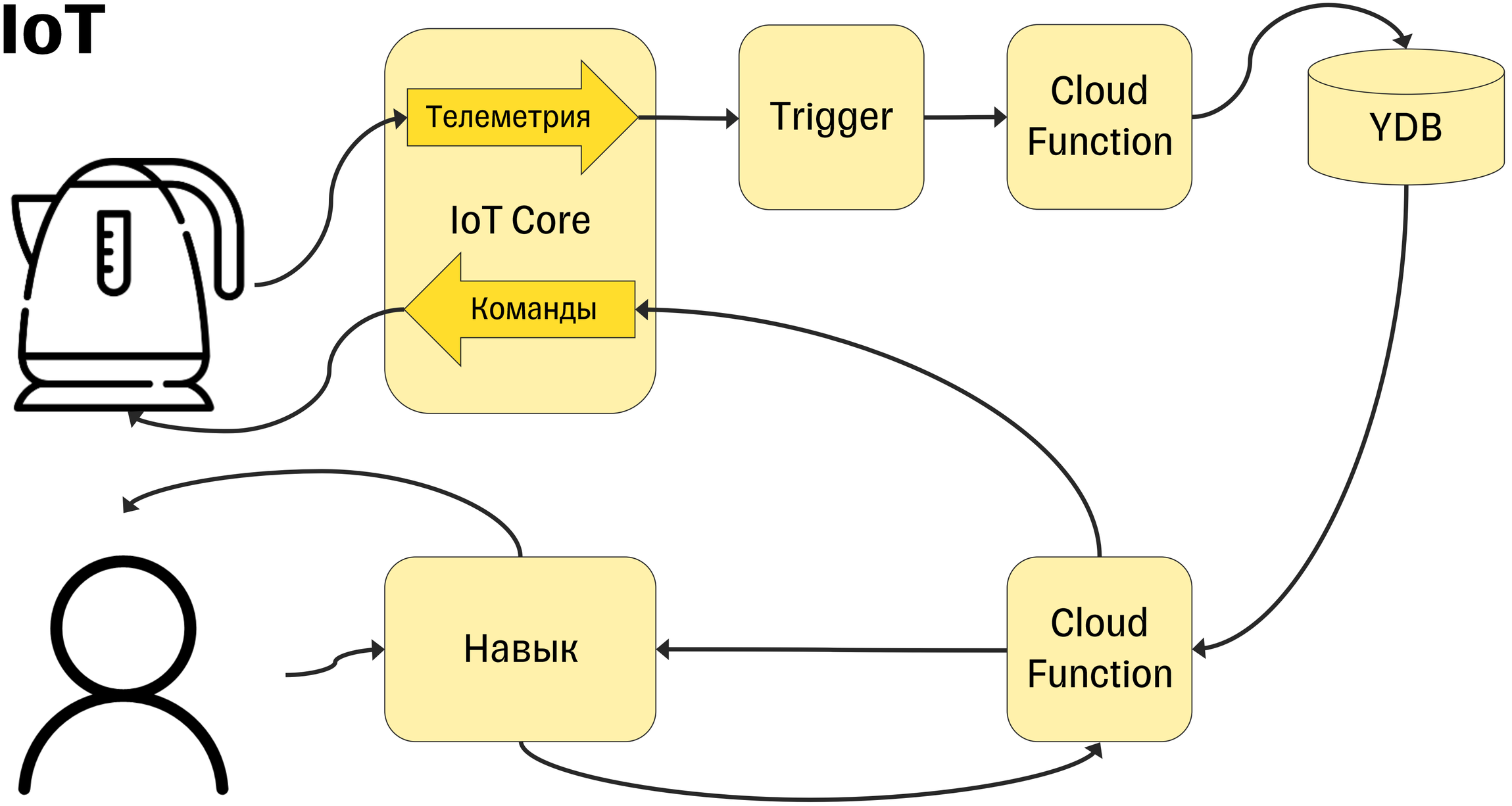
IoT

```
var response = await tableClient.SessionExec(async session =>
{
    var query = @"SELECT event_date, temperature, water_level, is_on
                  FROM teapot_state
                  ORDER BY event_date DESC
                  LIMIT 1;";

    return await session.ExecuteDataQuery(
        query: query,
        txControl: TxControl.BeginSerializableRW().Commit());
});
response.Status.EnsureSuccess();
var queryResponse = (ExecuteDataQueryResponse) response;
var resultSet = queryResponse.Result.ResultSets[0];

var temp = (float?)resultSet.Rows[0][1];

Console.WriteLine($"Temperature: {temp}");
```

IoT

```
var tlsOptions = new MqttClientOptionsBuilderTlsParameters
{
    SslProtocol = SslProtocols.Tls12,
    UseTls = true
};
tlsOptions.CertificateValidationCallback += CertificateValidationCallback;

var options = new MqttClientOptionsBuilder()
    .WithClientId($"Test_C#_Client_{Guid.NewGuid()}")
    .WithTcpServer(MqttServer, MqttPort)
    .WithTls(tlsOptions)
    .WithCleanSession()
    .WithCredentials(id, password)
    .Build();

var factory = new MqttFactory();
mqttClient = factory.CreateMqttClient();

await mqttClient.ConnectAsync(options, CancellationToken.None);

await mqttClient.PublishAsync(topic, message, MqttQualityOfServiceLevel.AtLeastOnce);
```

IoT


Телеметрия
Trigger
Cloud Function
YDB
IoT Core
Команды
Навык
Cloud Function

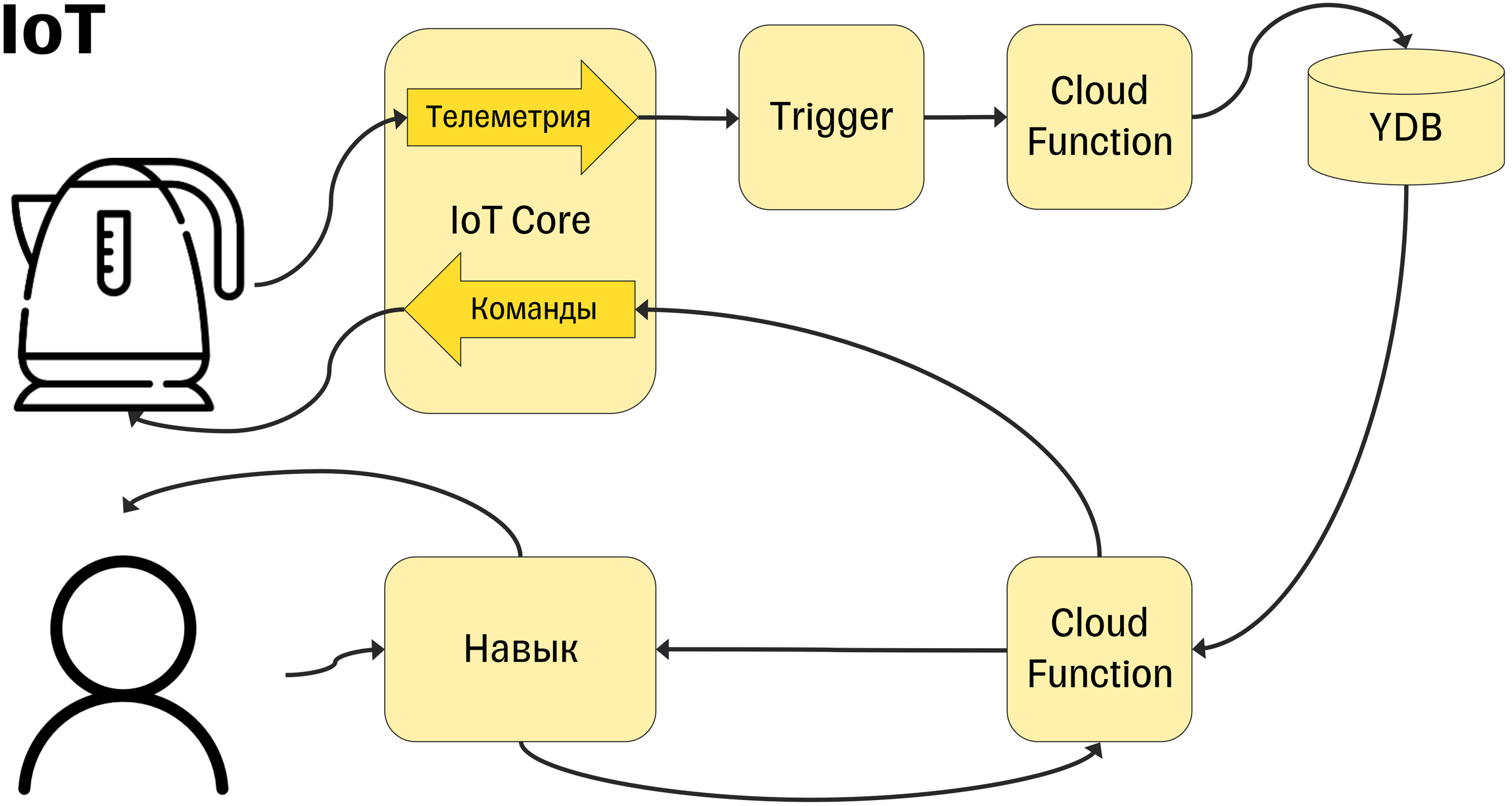
IoT

```
var caCert = new X509Certificate(s_certificate);
_mqttDeviceClient =
    new MqttClient(_mqttServer, _mqttPort, true, caCert, null, MqttSslProtocols.TLSv1_2);
_mqttDeviceClient.Connect(Guid.NewGuid().ToString(), _deviceId, _devicePassword);

_mqttDeviceClient.Subscribe(new[] { topicName }, new[] { MqttQoSLevel.AtLeastOnce });
_mqttDeviceClient.MqttMsgPublishReceived += HandleIncomingMessage;
```

IoT

```
var caCert = new X509Certificate(s_certificate);
_mqttDeviceClient =
    new MqttClient(_mqttServer, _mqttPort, true, caCert, null, MqttSslProtocols.TLSv1_2);
_mqttDeviceClient.Connect(Guid.NewGuid().ToString(), _deviceId, _devicePassword);

_mqttDeviceClient.Subscribe(new[] { topicName }, new[] { MqttQoSLevel.AtLeastOnce });
_mqttDeviceClient.MqttMsgPublishReceived += HandleIncomingMessage;


private void ReportState(string topicName, TeapotState state)
{
    _mqttDeviceClient.Publish(topicName,
        Encoding.UTF8.GetBytes($"{state.Temperature};{state.WaterLevel};{state.IsOn}"));
}

private void HandleIncomingMessage(object sender, MqttMsgPublishEventArgs e)
{
    var messageStr = Encoding.UTF8.GetString(e.Message, 0, e.Message.Length);
    int temp = 0;
    if (int.TryParse(messageStr, out temp))
        SetTargetState(new TeapotTargetState() { Temperature = temp });
}
```

IoT


Телеметрия
Trigger
Cloud Function
YDB
IoT Core
Команды
Навык
Cloud Function

Экономика мероприятия

	AliExpress	Локально
Чайник		699
ESP32	300	500
HX711	150	275
DS1820	50	100
Relay	50	150
ACS712	65	120
Итого	**615**	**1145**

Т БАНК

Спасибо!