РАЗПРЕДЕЛЕНИ ПРИЛОЖЕНИЯ

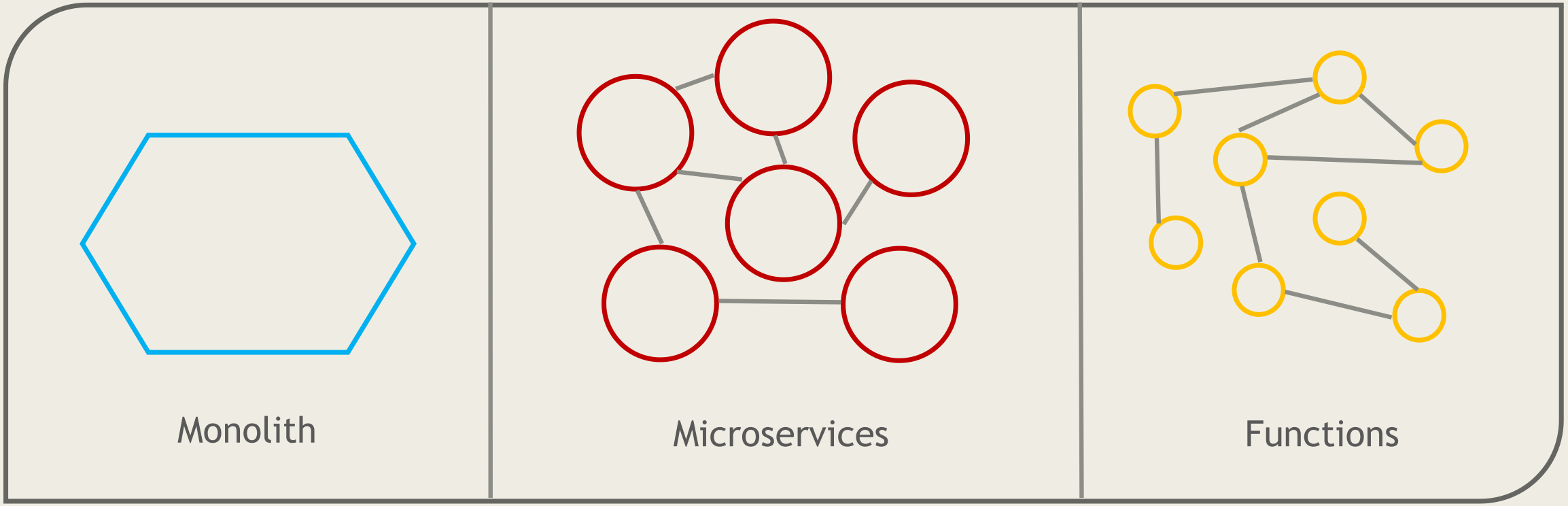
Павел Кюркчиев
Ас. към ПУ „Паисий Хилендарски“
@pkyurkchiev

АРХИТЕКТУРА „БЕЗ СЪРВЪР“

Традиционна архитектура

Традиционна архитектура
3 tier client-oriented

Еволюцията на бизнес логиката



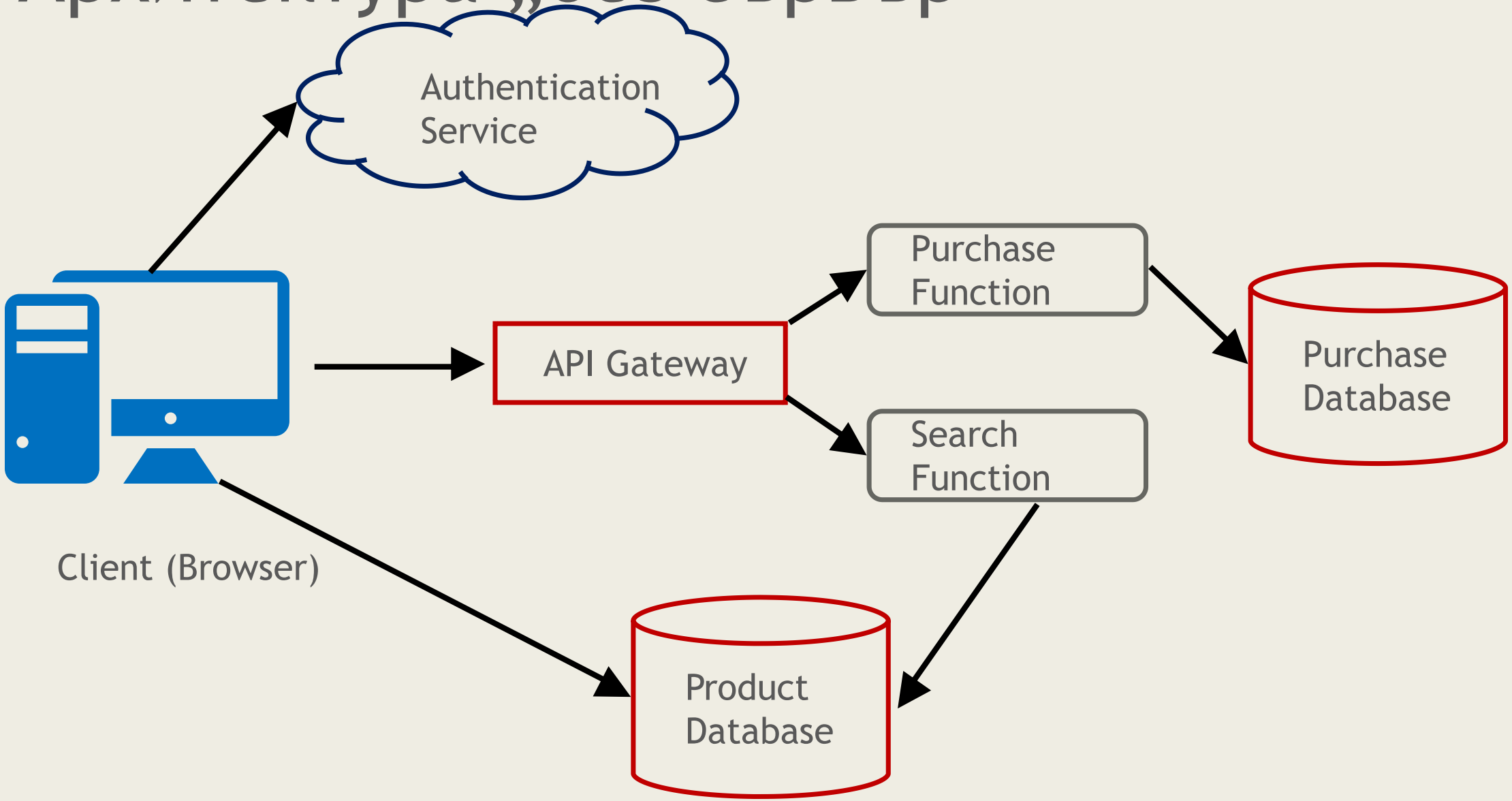
Какво е архитектура „без сървър“?

- „Без сървър“ архитектура наричаме приложенията, които в голямата си част или изцяло зависят от трета страна за обработката и обстлужването на техните нужни. Този тип услуги е описван като BaaS ((Mobile) Backend as a Service).

• • •

- „Без сървър“ архитектура може да означава и приложения, на които голяма част от логиката е написа от програмисти, но е качена на отдалечени контейнери, които се извикват и изпълняват при определени събития. Тези контейнери са напълно подържана от трета страна. За тях можем да мислим като за (Functions as a service / FaaS).

Архитектура „без сървър“



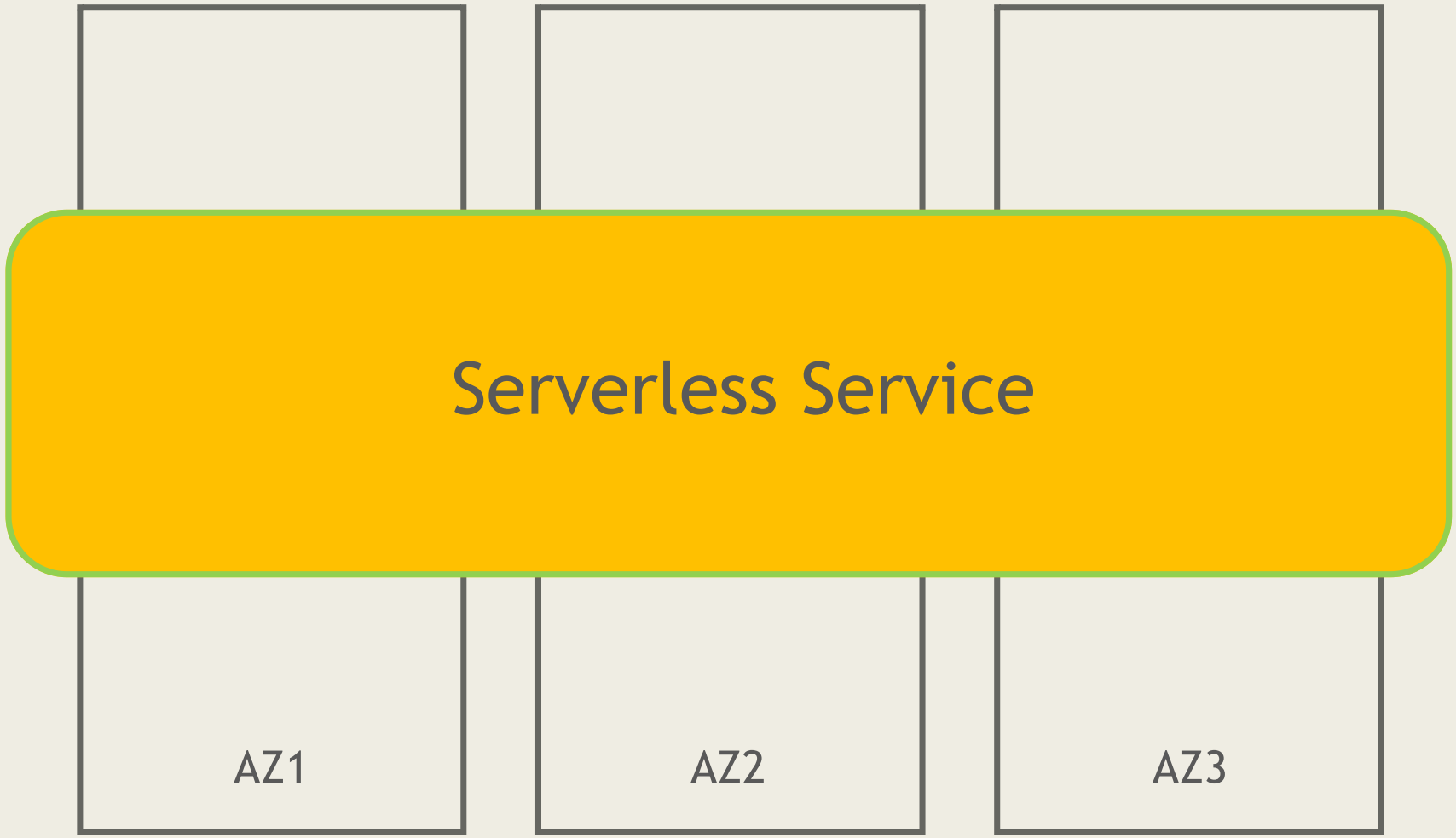
Примери за Functions

- Azure
  - *Azure Functions - C#, F#, Node.js, Java, PHP …*
- Amazon AWS
  - *Lambda Functions – Node.js, Java, C#, Go, Python ..*
- Google Firebase
  - *Firebase Functions – Node.js*

„Без сървър“ означава

- Без нужда от собствен сървър или от управлението му
- Плащаме само това, което използваме
- Разширение само според нашите потребности
- Достъпност и толерантност към средата

Регионална услуга



Добри практики Functions

- Да се минимизира големината на пакетите
- Да се разделя handler от основната логика на функцията
- Да се използват Environment Variables за модифициране поведението
- Да се възползваме от “Max Memory Used” за да определим правилно големината на function
- Да се премахнат големите неизползвани функции

AWS Lambda functions

Анатомия на Lambda Functions

- Handler() function
  - *Функцията да бъде извикана при възникването на някакъв обект*
- Event object
  - *Изпращаната информация по време на извикване на функцията*
- Context object
  - *Достъпен метод отговарящ при изпълнението на функцията*

```
exports.myHandler = function(event, context, callback) {
      console.log("value1 = " + event.key1);
      console.log("value2 = " + event.key2);
      callback(null, "some success message");
}
```

AWS Lambda function

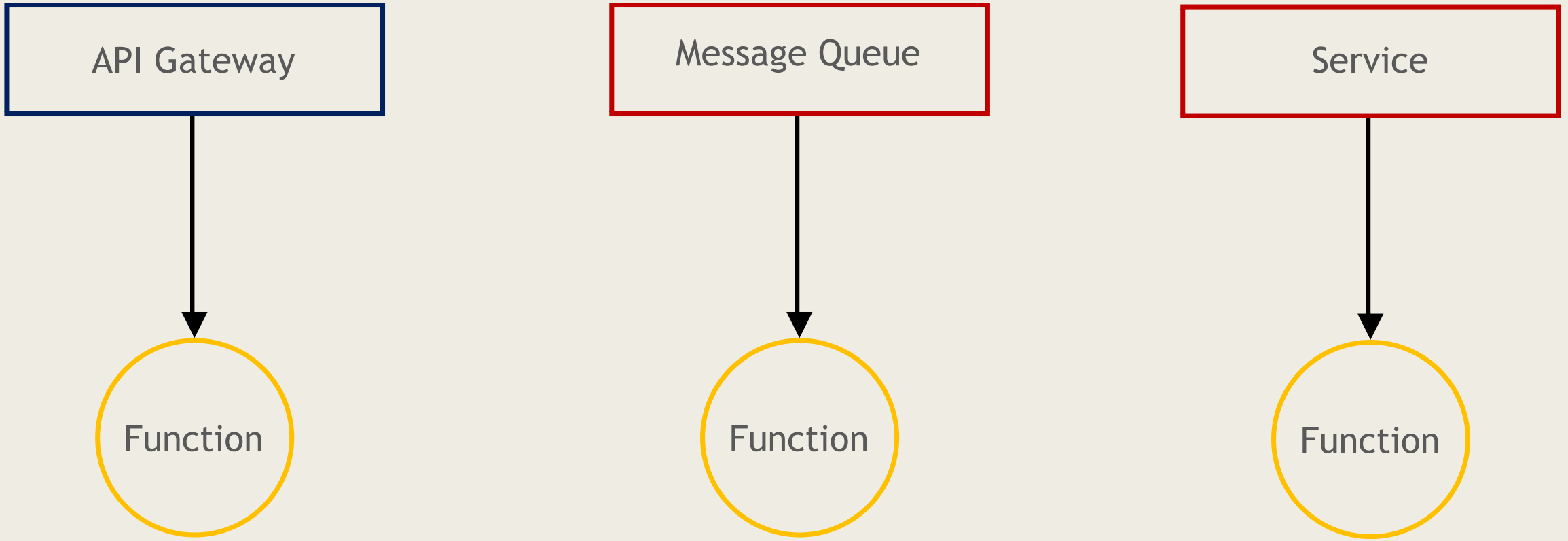
JavaScript

Functions модел на изпълнение

Synchronous (push)

Asynchronous (event)

Stream-based

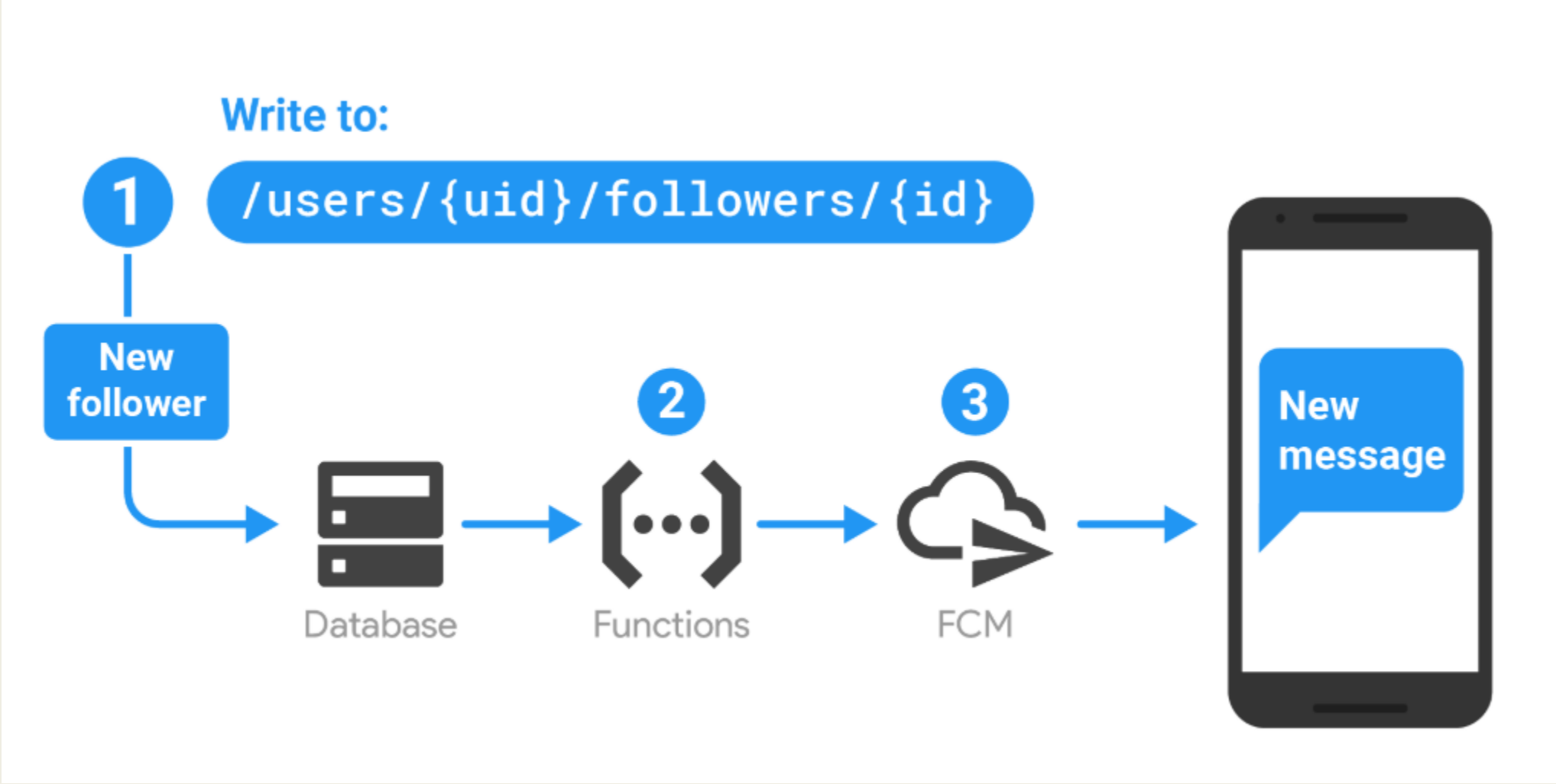


Google Firebase Functions

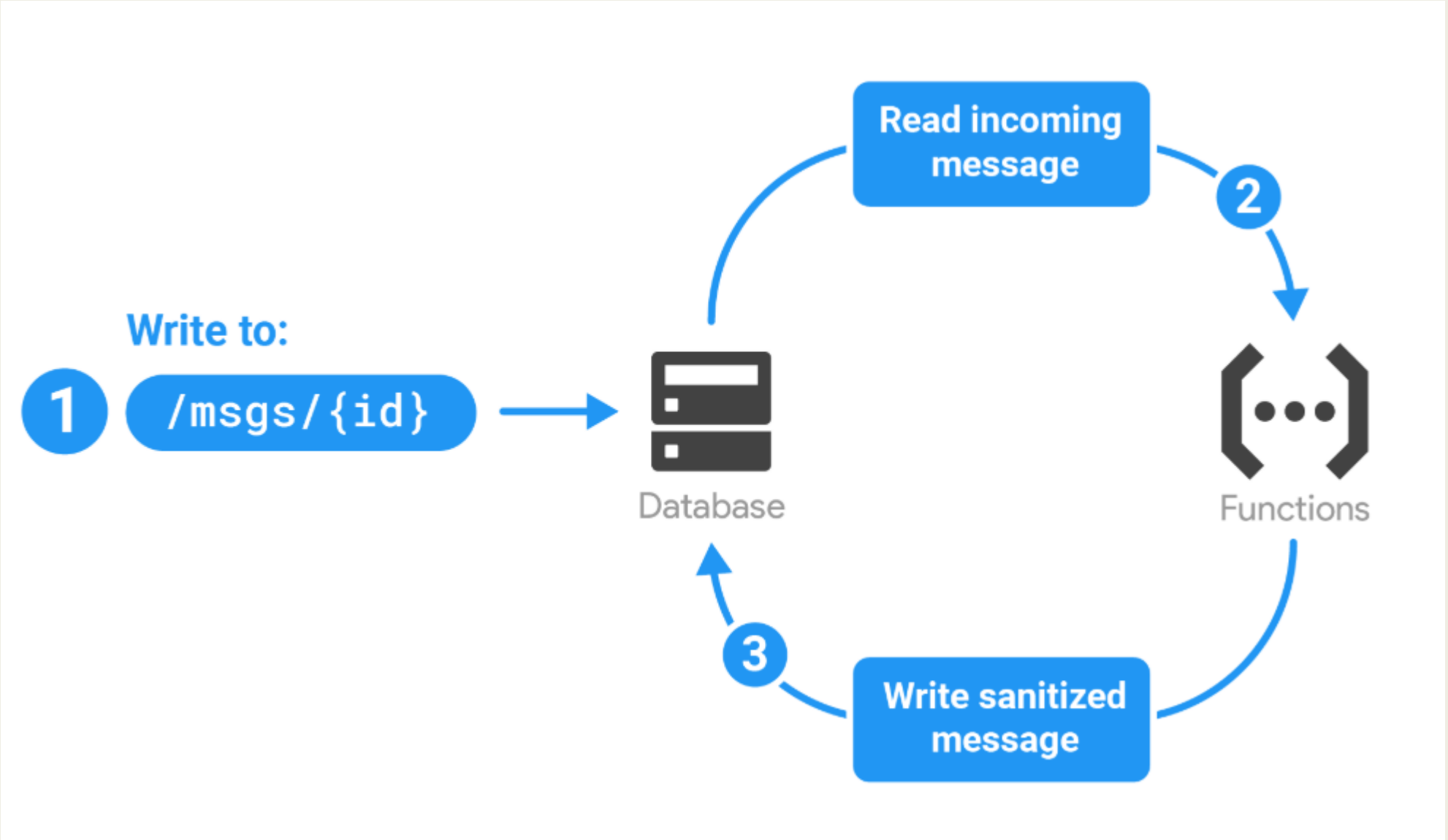
Интеграция на Firebase platform

- Cloud Firestore Triggers
- Realtime Database Triggers
- Remote Config Triggers
- Firebase Authentication Triggers
- Google Analytics for Firebase Triggers
- Crashlytics Triggers
- Cloud Storage Triggers
- Cloud Pub/Sub Triggers
- HTTP Triggers

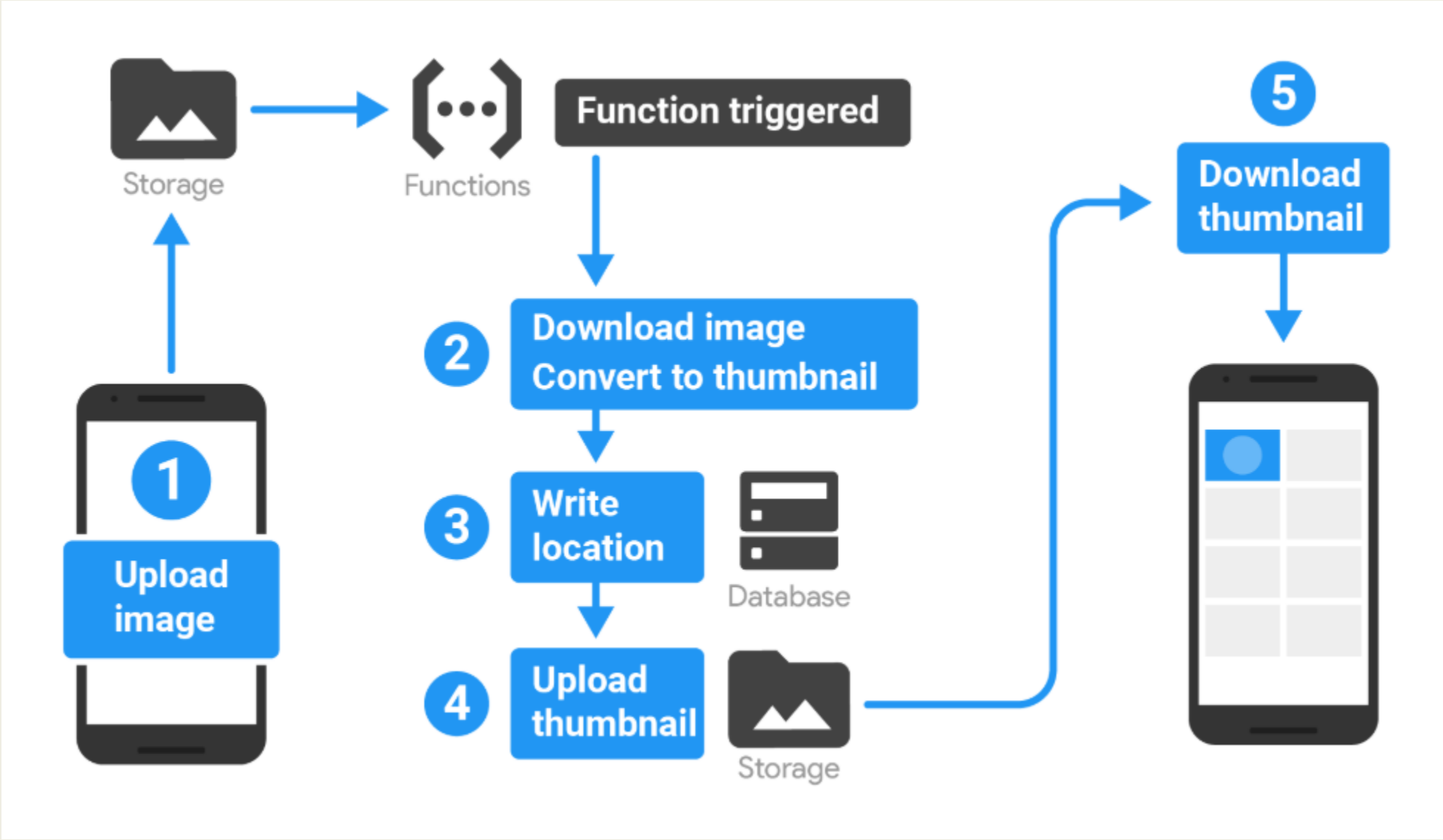
Уведомяване на потребител



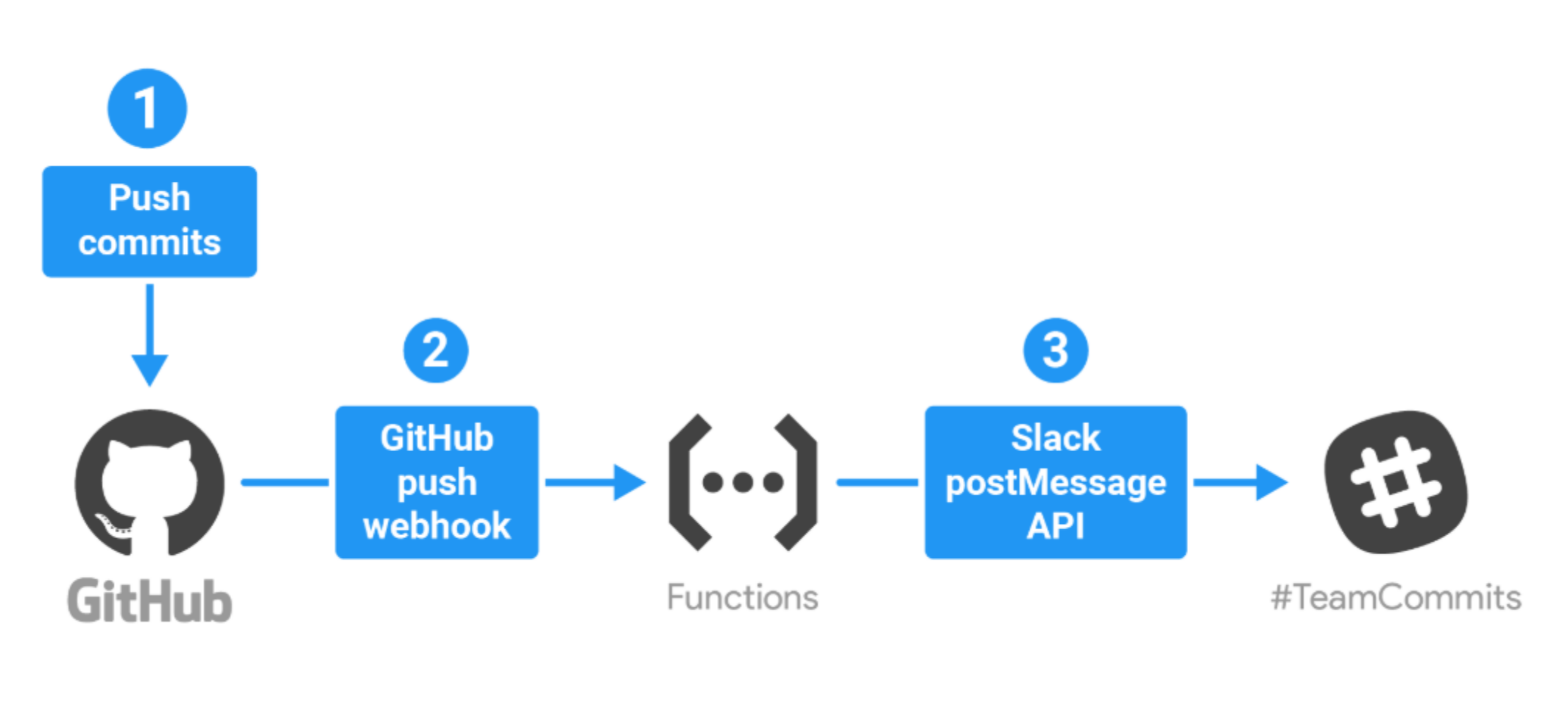
Realtime Database интеграция



Обработка на сложни задачи



Интеграция с трети услуги



DEMO FIREBASE FUNCTION 1

https://github.com/pkyurkchiev/distributed-applications-se/tree/master/examples/helloWorld-firebase-function-ts

DEMO FIREBASE FUNCTION 2

https://github.com/pkyurkchiev/distributed-applications-se/tree/master/examples/translate-firebase-function-js

Azure Functions

Интеграция на Azure function

- Azure Cosmos DB
- Azure Event Hubs
- Azure Event Grid
- Azure Notification Hubs
- Azure Service Bus (queues and topics)
- Azure Storage (blob, queues, and tables)
- On-premises (using Service Bus)
- Twilio (SMS messages)

Azure functions 1 vs 2

Language	1.x	2.x
C#	GA (.NET Framework 4.7)	GA (.NET Core 2.2)
JavaScript	GA (Node 6)	GA (Node 8 & 10)
F#	GA (.NET Framework 4.7)	GA (.NET Core 2.2)
Java	N/A	GA (Java 8)
PowerShell	Experimental	Preview (PowerShell Core 6)
Python	Experimental	Preview (Python 3.6)
TypeScript	Experimental	GA (supported through transpiling to JavaScript)
Bash	Experimental	N/A
Batch (.cmd, .bat)	Experimental	N/A
PHP	Experimental	N/A

DEMO AZURE FUNCTIONS

https://github.com/pkyurkchiev/distributed-applications-se/tree/master/examples/ToDoOperations

ВЪПРОСИ ?