



UNIVERSITY OF TARTU

INSTITUTE OF COMPUTER SCIENCE

# Nanoteenused

Pelle Jakovits

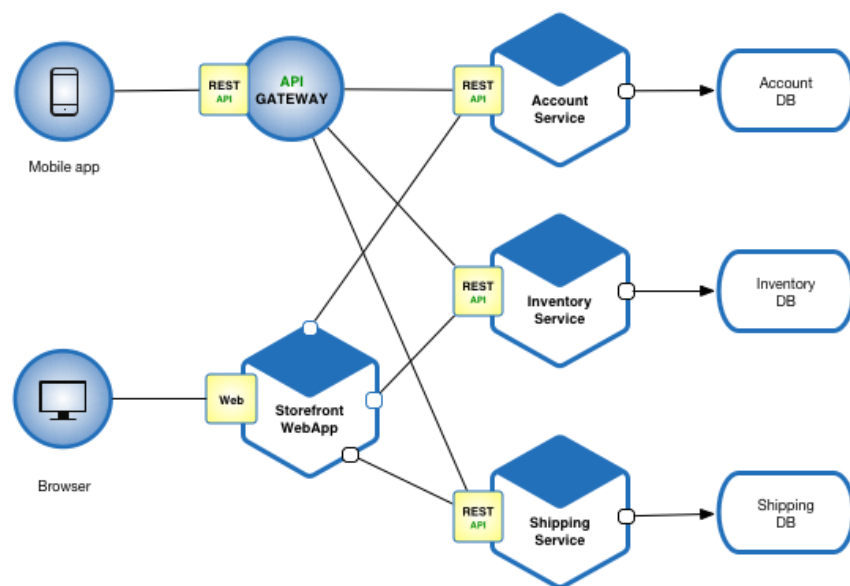
Mai 2025, Tartu

# Sisukord

- Nanoteenused
- Nanoteenuste platvormid ja implementatsioonid
- Eelised
- Puudused

# Mikroteenused

- Hajussüsteemide arhitektuuri muster
- Tarkvarasüsteem koosneb väikestest autonoomsetest ja spetsialiseerunud teenustest, mis omavahel suhtlevad standardsete API'de ja protokollide kaudu
  - Ehitame suurema rakenduse iseseisvate aga koostööd tegevate teenustena
- **Autonoomsus** - iga mikroteenus jookseb eraldi rakendusena, eraldi keskkonnas.
- **Spetsialiseerimine** – iga mikroteenus haldab mingit kindlat süsteemi funktsionaalust (nt sisse logimist)
- **Standardite kasutamine**
  - HTTP, REST, SOAP, RPC, AMQP



# Mikroteenuste puudused

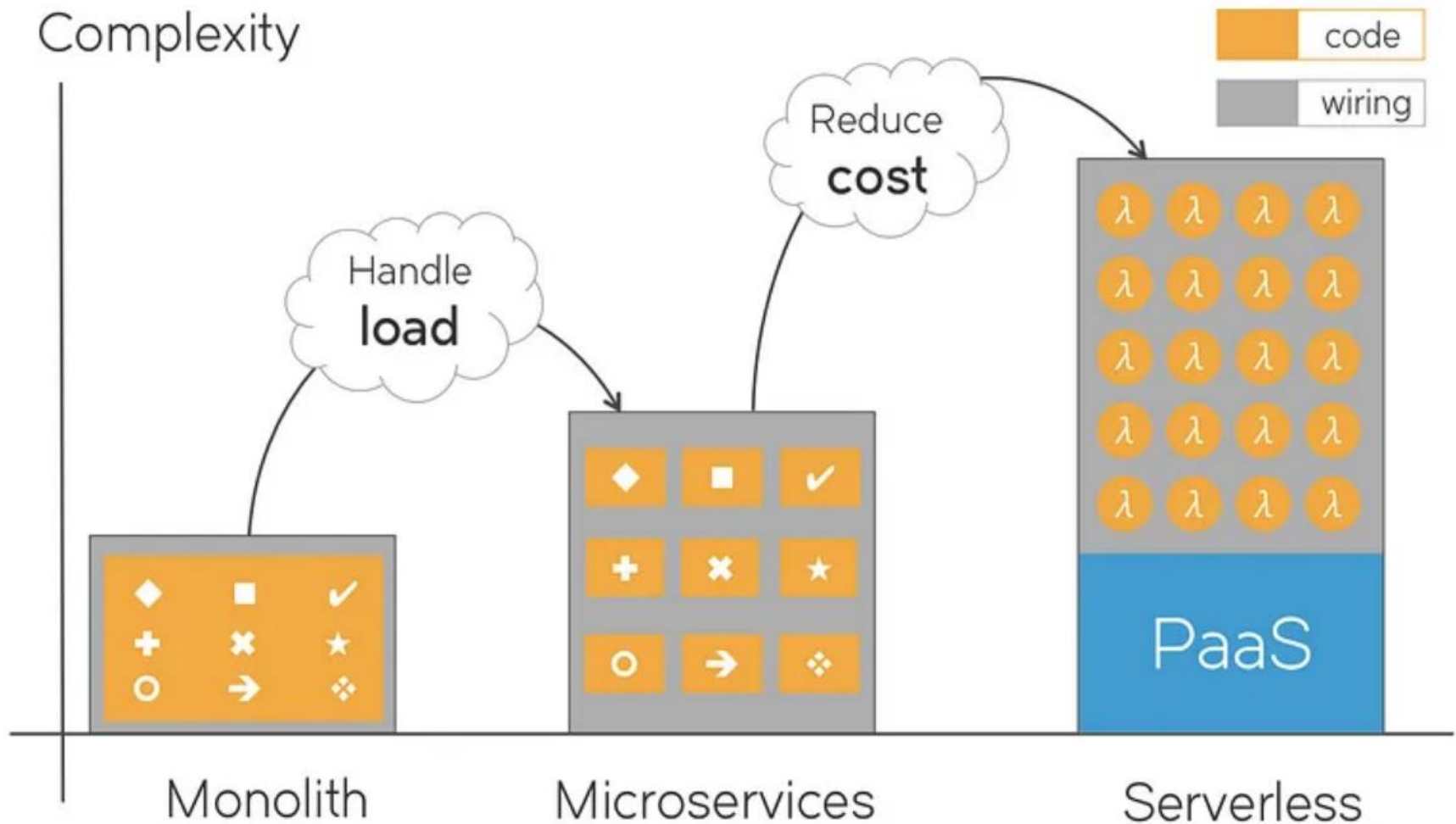
- Raskem aru saada kogu süsteemi hajutatud arhitektuurist
  - Vaja tegeleda teenuste vahelise suhtlusega
  - Päringud, mis vajavad mitme teenuse välja kutsumist, või andmeid, on keerulisemad arendada ja testida
  - Teenuste vaheliste interaktsioonide testimine on keerulisem
- Kogu süsteemi korraga juurutada, üles seada on keerulisem
- Suurem resursside (eriti mälu) kasutus, kui igal teenusel on oma (mitte jagatud ) keskkond
- **Palju väikseid kotneinereid, mis taustal jooksevad**
  - Kas on vaja, et nad koguaeg jooksevad?
  - Aga kui sisendeid ei ole?

**NANOTEENUSED**

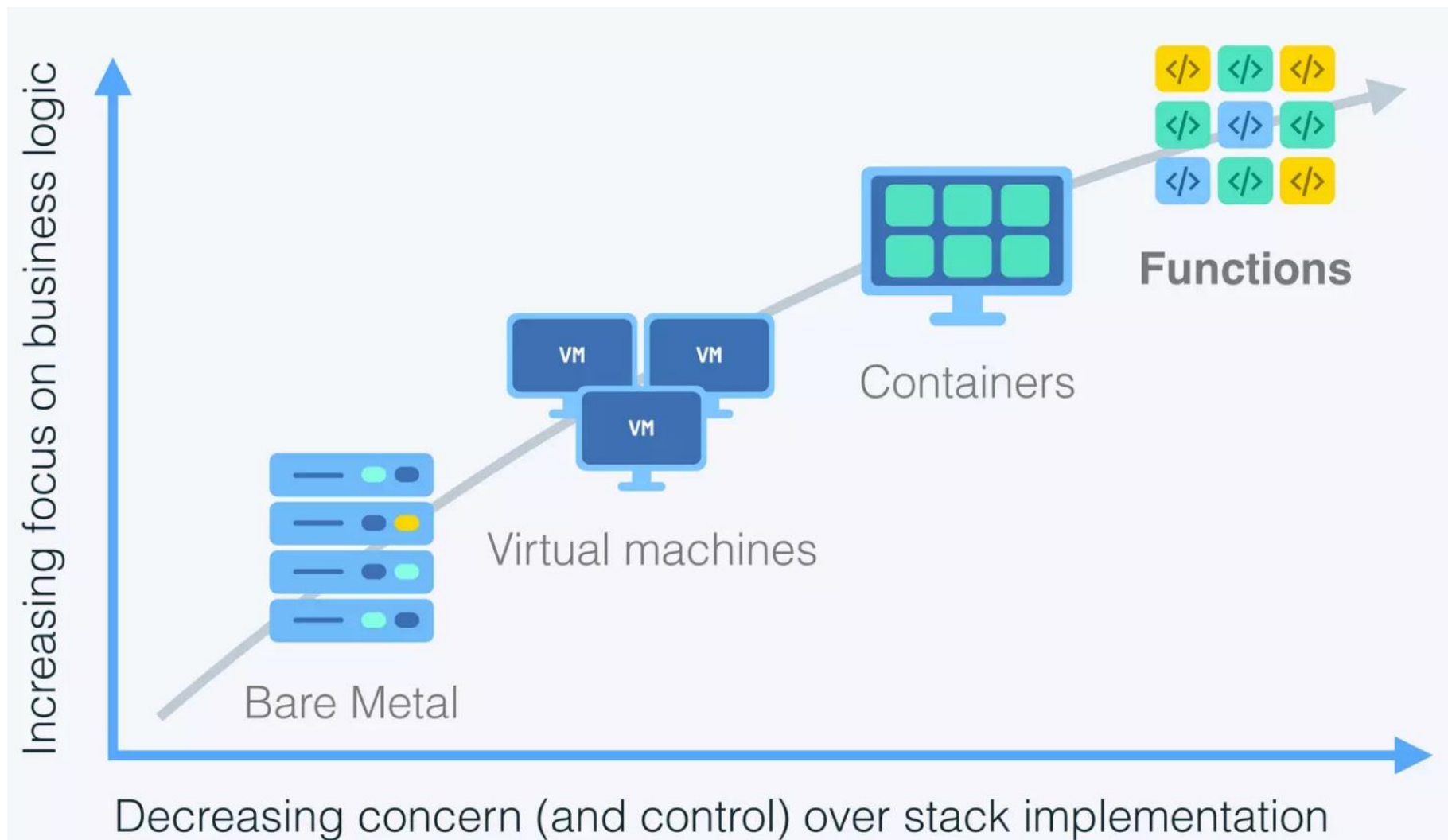
# Function as a Service (FaaS)

- Tuntud ka kui: Serverless
  - Serverivabad rakendused/teenused
- Funktsioonid on üksteisest sõltumatud
  - Eraldi skaleeritavad, hallatavad ja hinnastatavad
  - Iga funktsioon võib olla kirjutatud erinevas keeles
- Loogiline jätk mikroteenustest väiksemaks minemisel
- Sündmustepõhine käivitus:
  - Funktsioonid ei "jookse" taustal
  - Ei makse "idle" eest
- Staatuseeta rakendused
- Tavaliselt jooksevad väga lühikest aega
- Skaleeritakse automaatselt

# Monoliitsetest rakendustest nanoteenusteni



# Infrastruktuurist eemaldumine





# Sündmustepõhine käivitus

- Funktsioonid ei "jookse" taustal
  - Käivitatakse ainult siis kui tekib mingi sündmus (Event)
- Põhineb eeltingimustel ja sündmuse päästikutel
  - **Sündmuse päästik (Trigger):** Uus pilt üles laetud S3'e
  - **Eeltingimus (Precondition):** Faili suurus on suurem kui 10MB
  - **Tegevus:** `Resize_image(filePath)`
- FaaS funktsiooni defineerib Päästik, Eeltingimused ja Tegevus



# Sündmused, päästikud

- Näited sündmustest - päästikutest, mis võivad nanofunktsioone välja kutsuda:
  - Uus pilt on pilve andmehoidlasse üles laaditud
  - Kliendi andmebaasi lisati uus kirje
  - Kirje muutus andmebaasis
  - Temperatuuri väärtus on kõrgem kui 100C!
  - Uus kiri saabus sõnumite järjekorda
  - Uus HTTP päring saabus

# Eeltingimused

- Iga sündmuse puhul ei pruugi olla vaja funktsiooni välja kutsuda
  - Filtri tüüpi päästikud
  - Temperatuur > 80
- Võib vaja minna teha otsus, mis tüüpi tegevus on vaja teha
  - Andmete sisu, või metandmete põhjal marsruutimine
  - Näiteks Content-type põhjal
  - Näiteks kui S3'e laetakse üles pilt või video

# Tegevused

- Funktsiooni sisu, kood, mida käivitatakse
- Kihti saab panna konteineri sisse nii, et igal funktsioonil on oma keskkond
  - Võib olla programmeeritud erinevates keeltes
- Ei ole disainitud jooksmas püsivalt
  - Ei jookse taustal!
  - FaaS platvorm käivitab siis kui "tuleb sisse uus sündmus" ja eeltingimus on täidetud (kui see eksisteerib)

# Näide: IBM Funktsioon HTML vormi näitamiseks

```
1 import sys
2
3 def main(dict):
4     form = '''
5     <html>
6     <body>
7         <form id="form_1" method="post" action="https://eu-gb.functions.appdomain.cloud/api/v1/web/
8             jakovits%40ut.ee_dev/default/pelleform">
9             User: <input id="user" name="user" size="30" type="text" value="" /> <br />
10            Message: <textarea rows="10" cols="30" id="message" name="message" type="text" value="">
11            <input id="saveForm" type="submit" name="submit" value="Send message" />
12        </form>
13    </body>
14    </html>
15    '''
16
17    return {
18        "headers": {
19            'Content-Type': 'text/html'
20        },
21        "statusCode": 201,
22        "body": form
23    }
```


User:

Message:

Send message

# Funktsioon vormist pärit andmete panemiseks andmebaasi

pelleform

Web Action 

Namespace: jakovits@ut.ee\_dev(London)



Code

Parameters

Runtime

Endpoints

Connected Triggers

Enclosing Sequences

Logs



Code ⓘ Python 3.6.6

Reset


Save



```
10 import sys
11 from cloudant.client import Cloudant
12
13 def addDocToDB(new_doc, username, apikey):
14     dbName = "labdb1"
15     client = Cloudant(username, apikey, connect=True)
16     myDatabase = client[dbName]
17     return myDatabase.create_document(new_doc)
18
19 def main(request):
20     new_doc = {'message': message, 'user': user}
21
22     modified_doc = addDocToDB(new_doc, request['username'], request['apikey'])
23
24     return {
25         "headers": {
26             'Content-Type': 'text/html'
27         },
28         "statusCode": 201,
29         "body": '<html><body><h3>Message added to the database</h3></body></html>'
30     }
31
```

# Funktsioonile REST/HTTP lõppunkti määramine

pelleform

Web Action 

Code


Parameters

Runtime

**Endpoints**

Connected Triggers

Enclosing Sequences



Logs 

Namespace: jakovits@ut.ee\_dev(London)


### Web Action

☒ **Enable as Web Action** Allow your Cloud Functions actions to handle HTTP events. Web Actions allow to control the response data and type by using a set of URL extensions, such as .json or .html. Learn more about [Web Actions](#).  
**Note:** The Web Action URL below requires to return a dict object that contains a body property.

☐ **Raw HTTP handling** When enabled your Action receives requests in plain text instead of a JSON body

HTTP Method	Auth	URL	
ANY 	Public	<a href="https://eu-gb.functions.appdomain.cloud/api/v1/web/jakovits%40ut.ee_dev/default/pelleform">https://eu-gb.functions.appdomain.cloud/api/v1/web/jakovits%40ut.ee_dev/default/pelleform</a>	

### REST API

HTTP Method	Auth	URL	
POST	<a href="#">API-KEY</a>	<a href="https://eu-gb.functions.cloud.ibm.com/api/v1/namespaces/jakovits%40ut.ee_dev/actions/pelleform">https://eu-gb.functions.cloud.ibm.com/api/v1/namespaces/jakovits%40ut.ee_dev/actions/pelleform</a>	

# FaaS platvormide näited



AWS Lambda

OpenLambda



Azure Functions



Red-Hat



Google Functions



Kubernetes



# Nanoteenuste näide

- Me soovime luua raamatute haldamise rakenduse
- Me soovime et kasutajad saaksid raamatute nimekirja vaadata, neid alla laadida, kustutada.
- Me loome iga funktsionaalsuse jaoks eraldi FaaS funktsiooni (nt Azure pilves)
  - Raamatute nimekiri
  - Raamatu kustutamine
  - Raamatu vaatamine
  - Raamatu otsimine
  - Raamatu loomine
- Iga funktsiooni seostame kindla REST aadressi ja operatsiooniga pilves
  - <https://pelleramatud.azurewebsites.net/api/raamatud>
- Saab kasutada eraldi API väravaid (gateway), et teha kõik meie funktsioonid sama API aadressi kaudu kätte saadavaks

# AWS Lambda

- Koodi/funktsioone saab käivitada AWS-is ilma infrastruktuuri või tarkvarakeskkonda haldamata
- Hind kujuneb päringute arvu ning **GB-Sec "Memory-Duration"** põhjal
- Tasuta: 1M **päringuid kuus**. Peale seda: \$0.20 iga 1M päringu kohta
- Tasuta: 400,000 **GB-Sec**. Peale seda: \$0.000017 iga 1 **GB-Sec**

# AWS Lambda veebiliides

The screenshot shows the AWS Lambda console interface for a function named 'hello-world-python'. The breadcrumb navigation indicates the path: Lambda > Functions > hello-world-python. The function name 'hello-world-python' is prominently displayed at the top, with buttons for 'Throttle', 'Copy ARN', and 'Actions' to its right. Below this, a 'Function overview' section is visible. A horizontal tab bar contains 'Code', 'Test', 'Monitor', 'Configuration', 'Aliases', and 'Versions', with 'Code' being the active tab. The 'Code source' section shows a file explorer on the left with 'hello-world-python' and 'lambda\_function.py'. The main editor displays the Python code for 'lambda\_function.py'. A red rectangle highlights the 'Test' button and its dropdown menu, which includes the option 'Configure test event'.

aws Services Search for services, features, blogs, docs, and more [Option+S] N. Virginia

Lambda > Functions > hello-world-python

hello-world-python Throttle Copy ARN Actions

Function overview Info

Code Test Monitor Configuration Aliases Versions

Code source Info Upload from

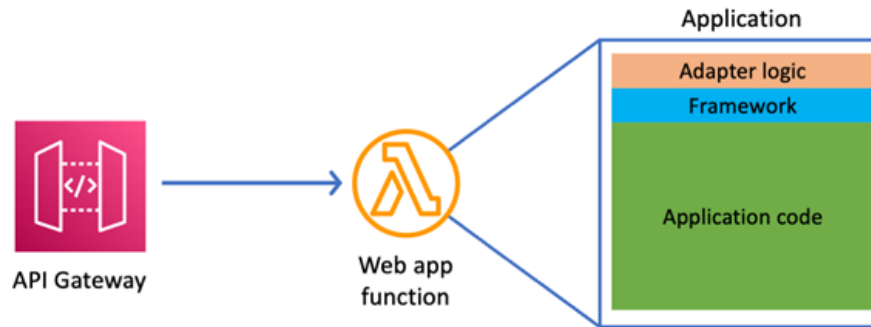
File Edit Find View Go Tools Window Test Deploy

Go to Anything (⌘ P) Configure test event ⌘ ⬆ C

```
1 import json
2
3 print('Loading function')
4
5
6 def lambda_handler(event, context):
7     #print("Received event: " + json.dumps(event, indent=2))
8     print("value1 = " + event['key1'])
9     print("value2 = " + event['key2'])
10    print("value3 = " + event['key3'])
11    return event['key1'] # Echo back the first key value
12    #raise Exception('Something went wrong')
13
```

Feedback Looking for language selection? Find it in the new Unified Settings © 2022, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

# AWS Lambda & API gateway



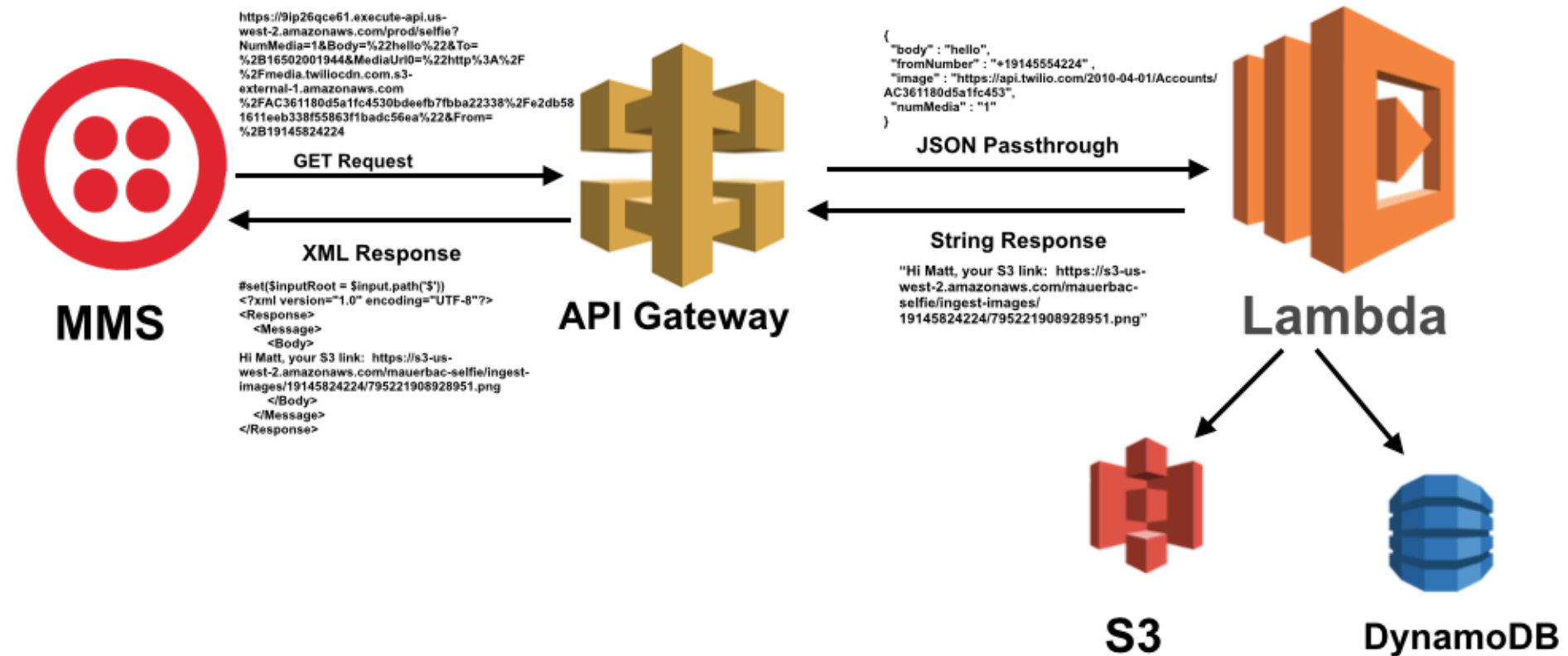
```
POST /v1/pets HTTP/2
Host: x.execute-api.eu-west-1..
User-Agent: curl/7.64.1
Accept: */*
Content-Type: application/json
Content-Length: 39
Body: {"data": "test"}
```



```
{
  "body": "{\"data\": \"test\"} ",
  "resource": "/{proxy+}",
  "path": "/path/to/resource",
  "httpMethod": "POST",
  "isBase64Encoded": true,
  "headers": {
    "Accept-Encoding": "gzip",
    "Accept-Language": "en-US,en;q=0.8"
  },
  "requestContext": {
    "accountId": "123456789012",
    "requestId": "c6af9ac6-7b61-..",
    ...
  }
}
```



# AWS Lambda & API gateway



# AWS Step Functions

- Visuaalsed töövood täiesti pilvepõhiste hajusrakenduste jaoks

# AWS Step Functions



# AWS Step Functions





# AWS Step Functions

The screenshot displays the AWS Step Functions console interface. On the left, a sidebar contains a search bar and a list of actions under the 'Actions' tab. The main workspace shows a workflow diagram with a 'Start' node, a dashed box labeled 'Drag first state here', and an 'End' node. On the right, the 'Form' tab is active, showing configuration options for the workflow.

**Search**

**Actions** | **Flow**

- AWS Lambda Invoke
- Amazon SNS Publish
- Amazon ECS RunTask
- AWS Step Functions StartExecution
- AWS Glue StartJobRun
- AWS Glue DataBrew StartJobRun
- Amazon EventBridge PutEvents
- AWS Batch SubmitJob
- Amazon API Gateway Request
- Amazon Athena StartQueryExecution

**Workflow**

**Comment - optional**  
A human-readable description of the state machine.

**TimeoutSeconds - optional**  
The maximum number of seconds an execution of the state machine can run. If it runs longer than the specified time, the execution fails with a States.Timeout.

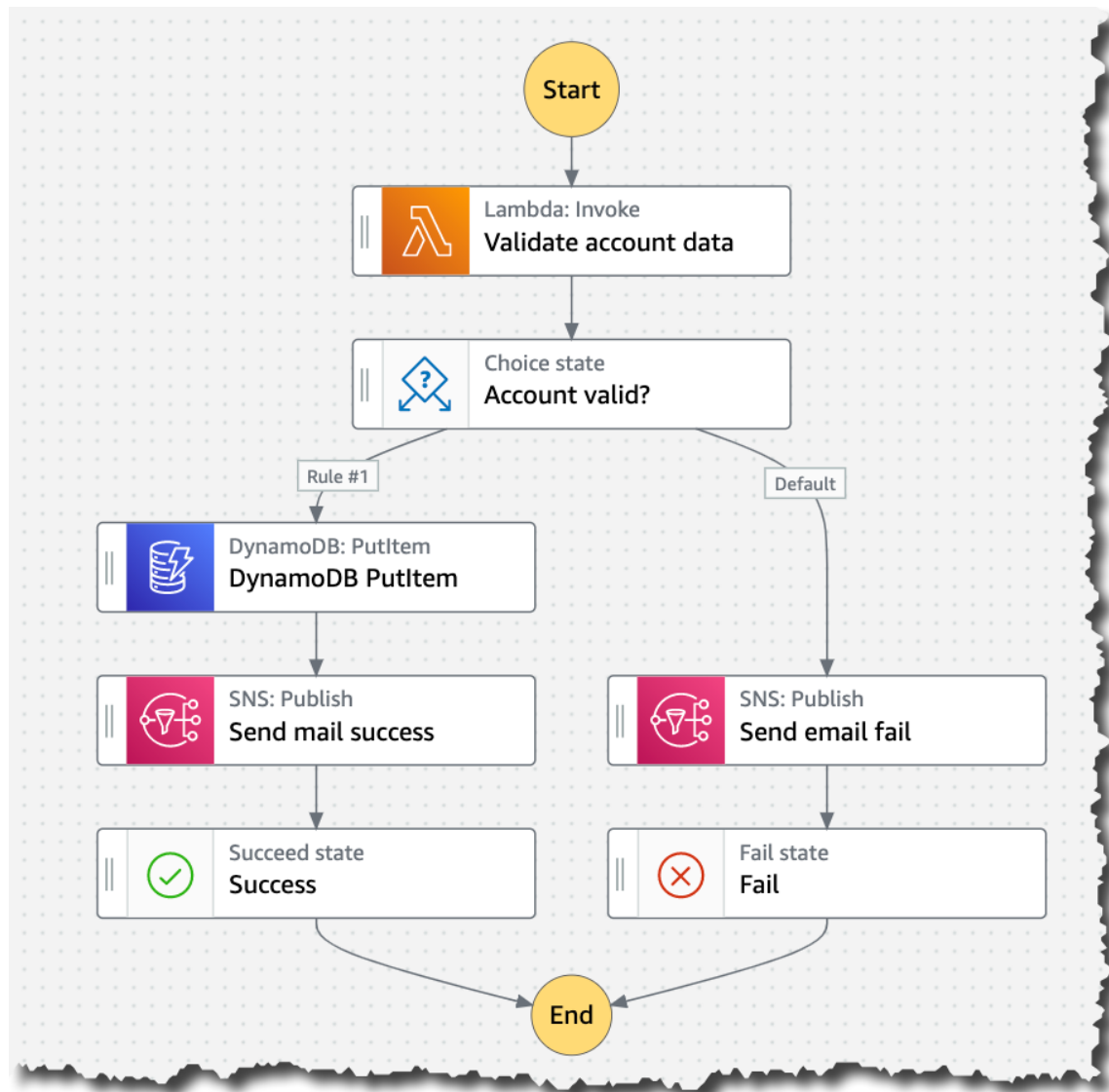
**Form** | **Definition**

Start

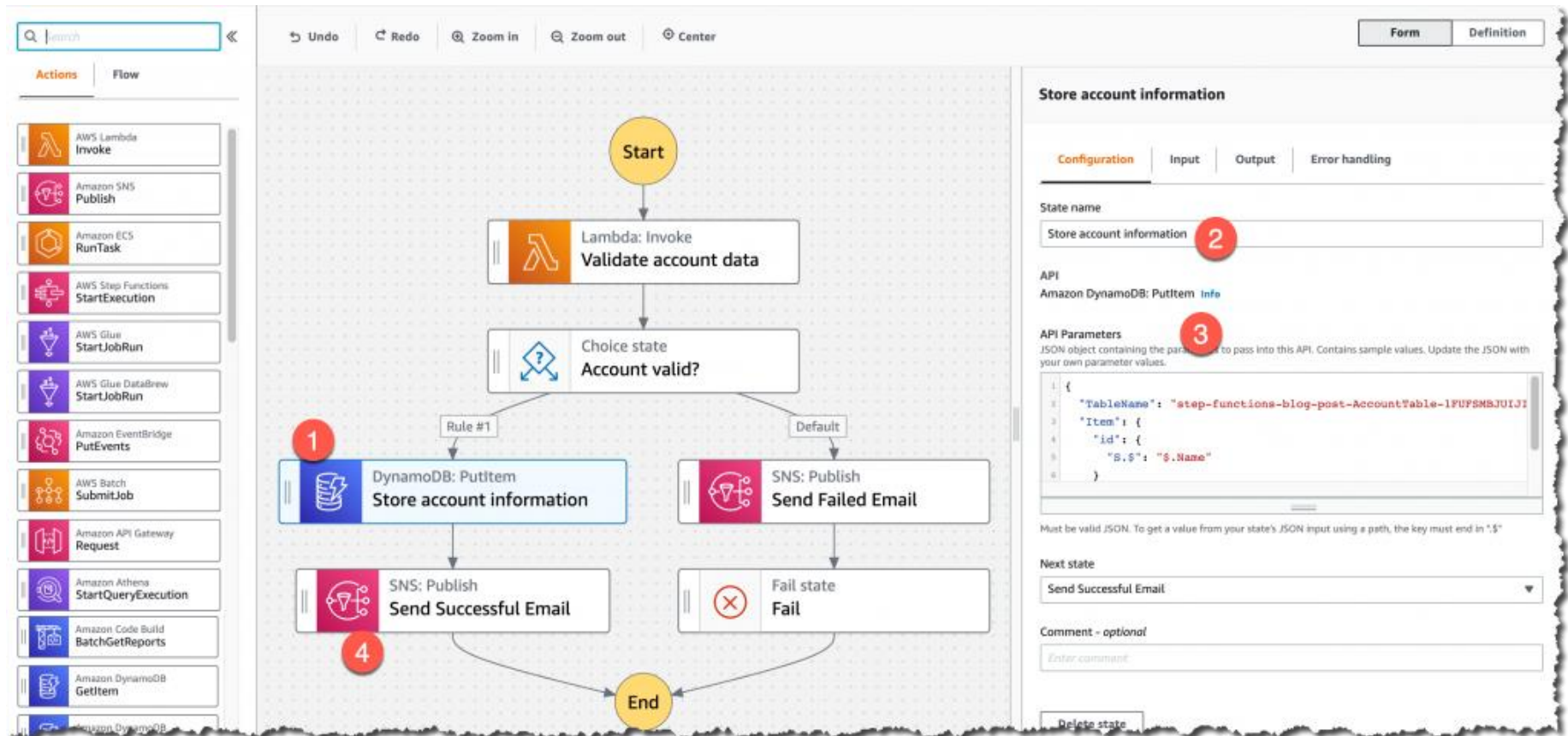
Drag first state here

End

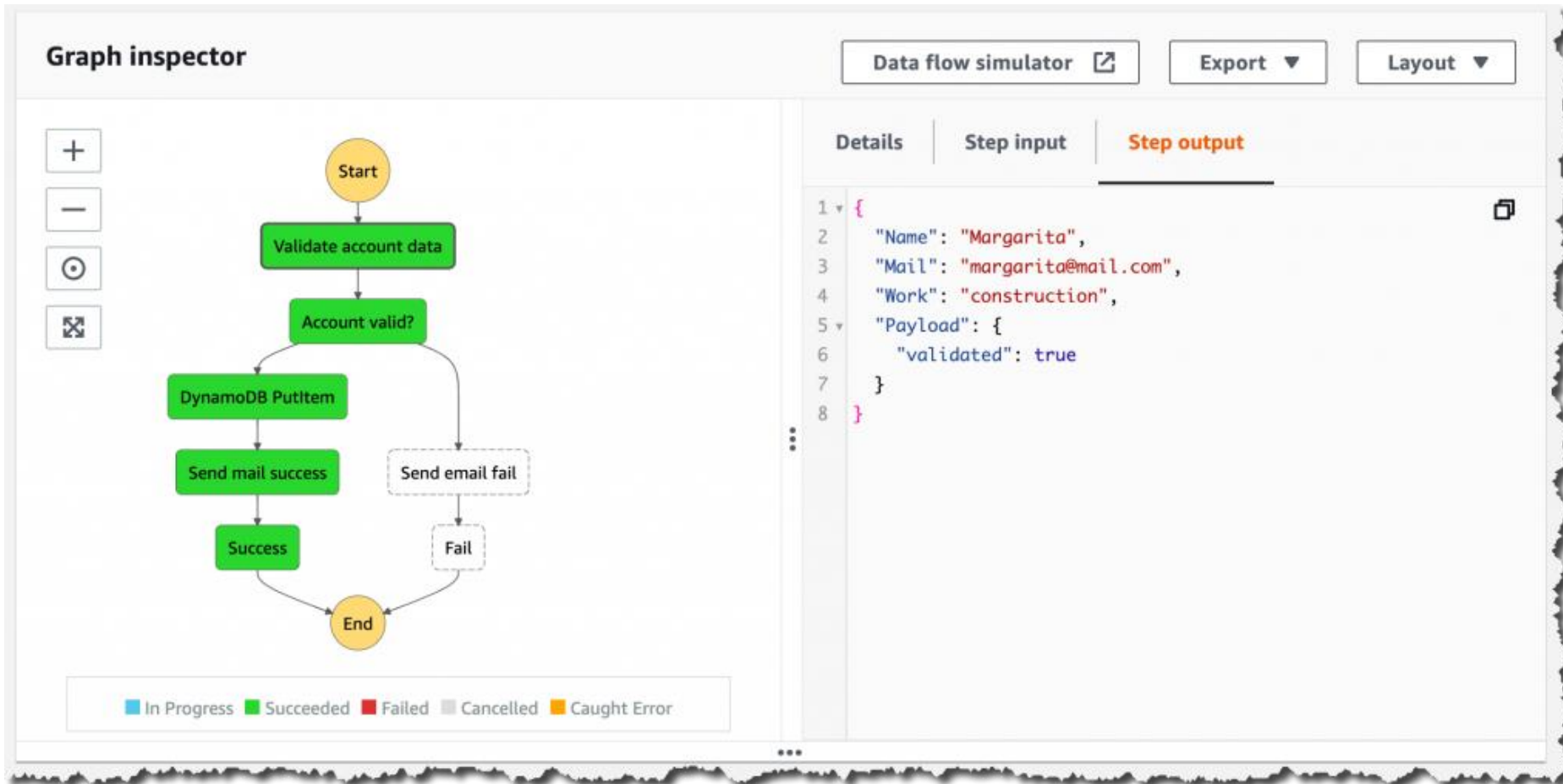
# AWS Step Functions



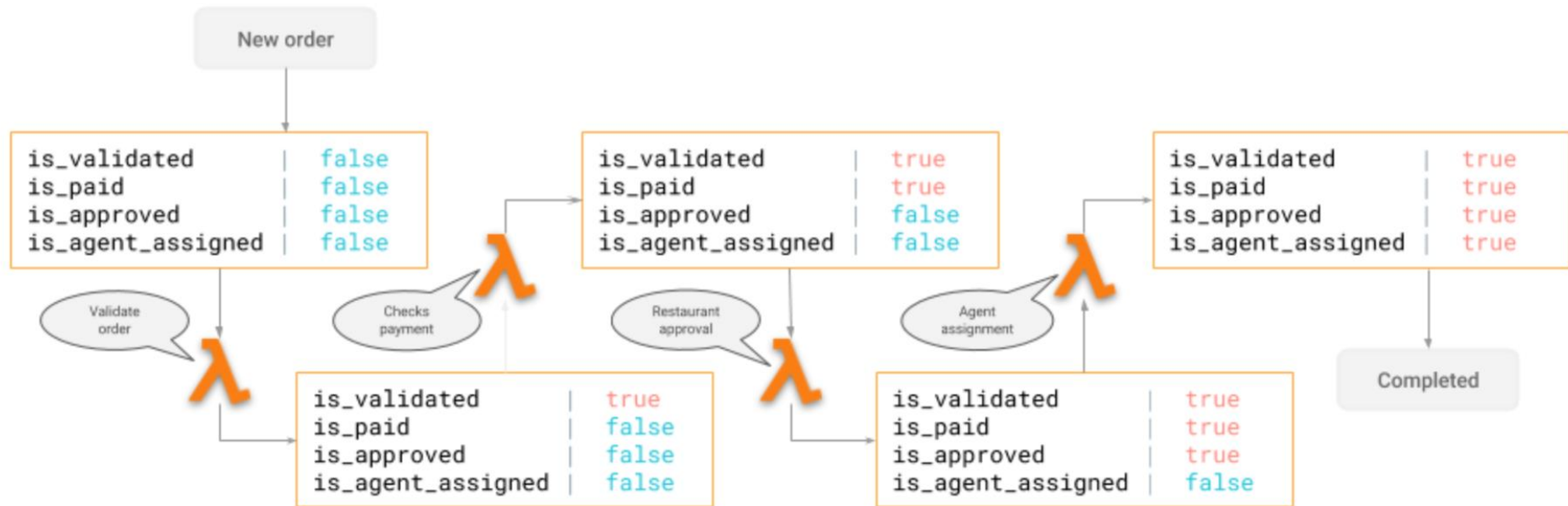
# AWS Step Functions



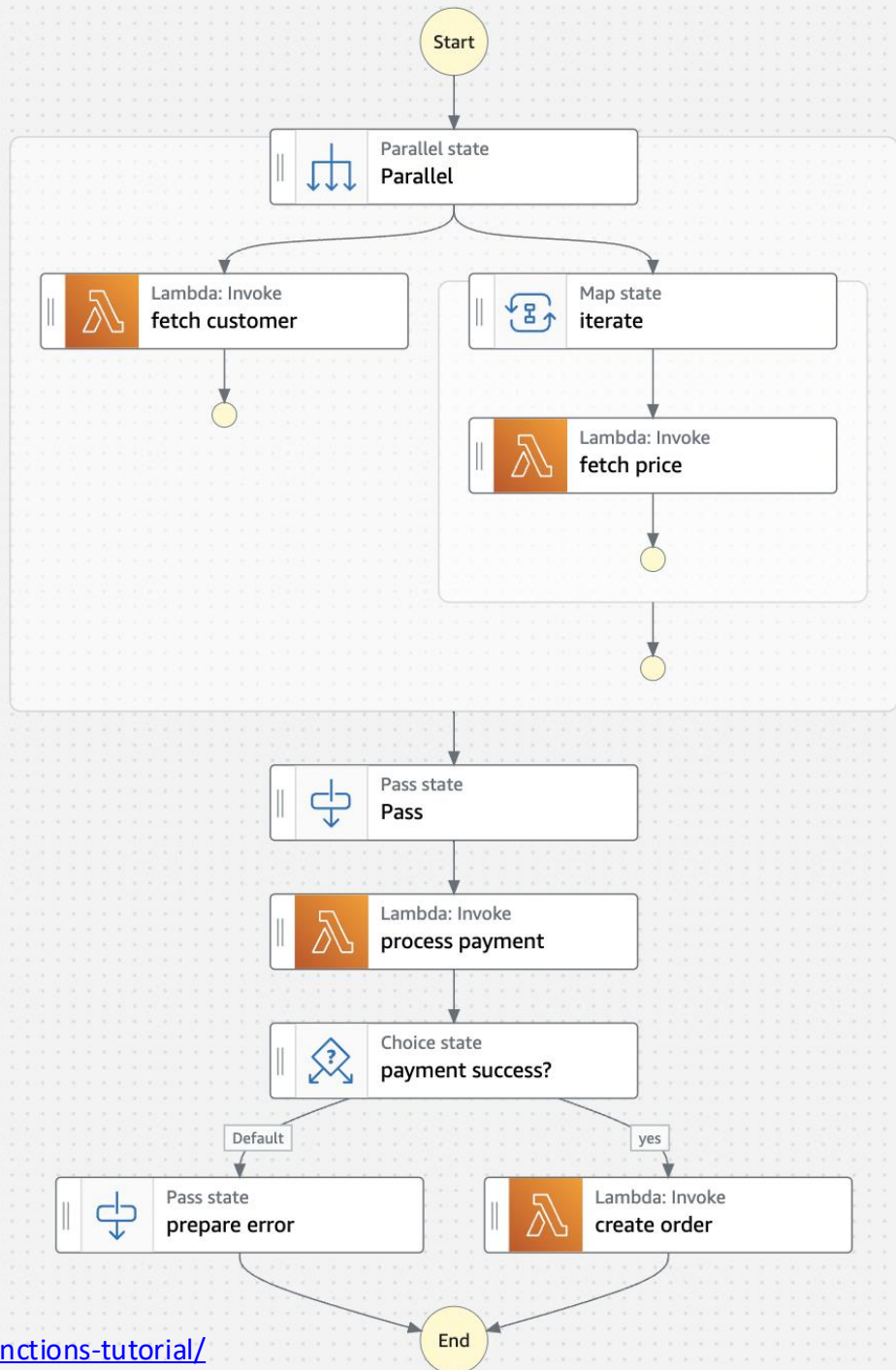
# AWS Step Functions



# Nanoteenuste näide II



# AWS Step functions + Lambda näide

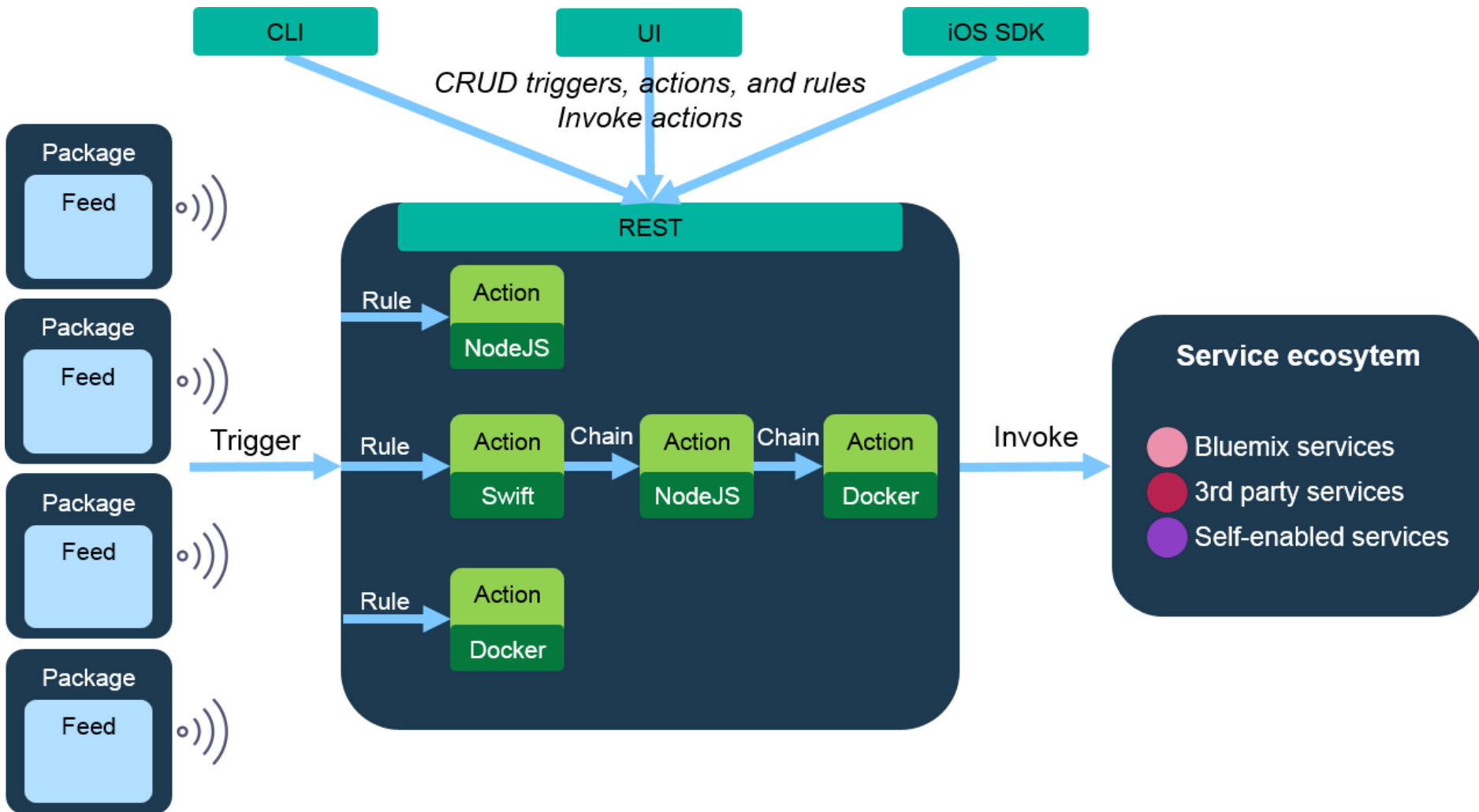




# Apache OpenWhisk

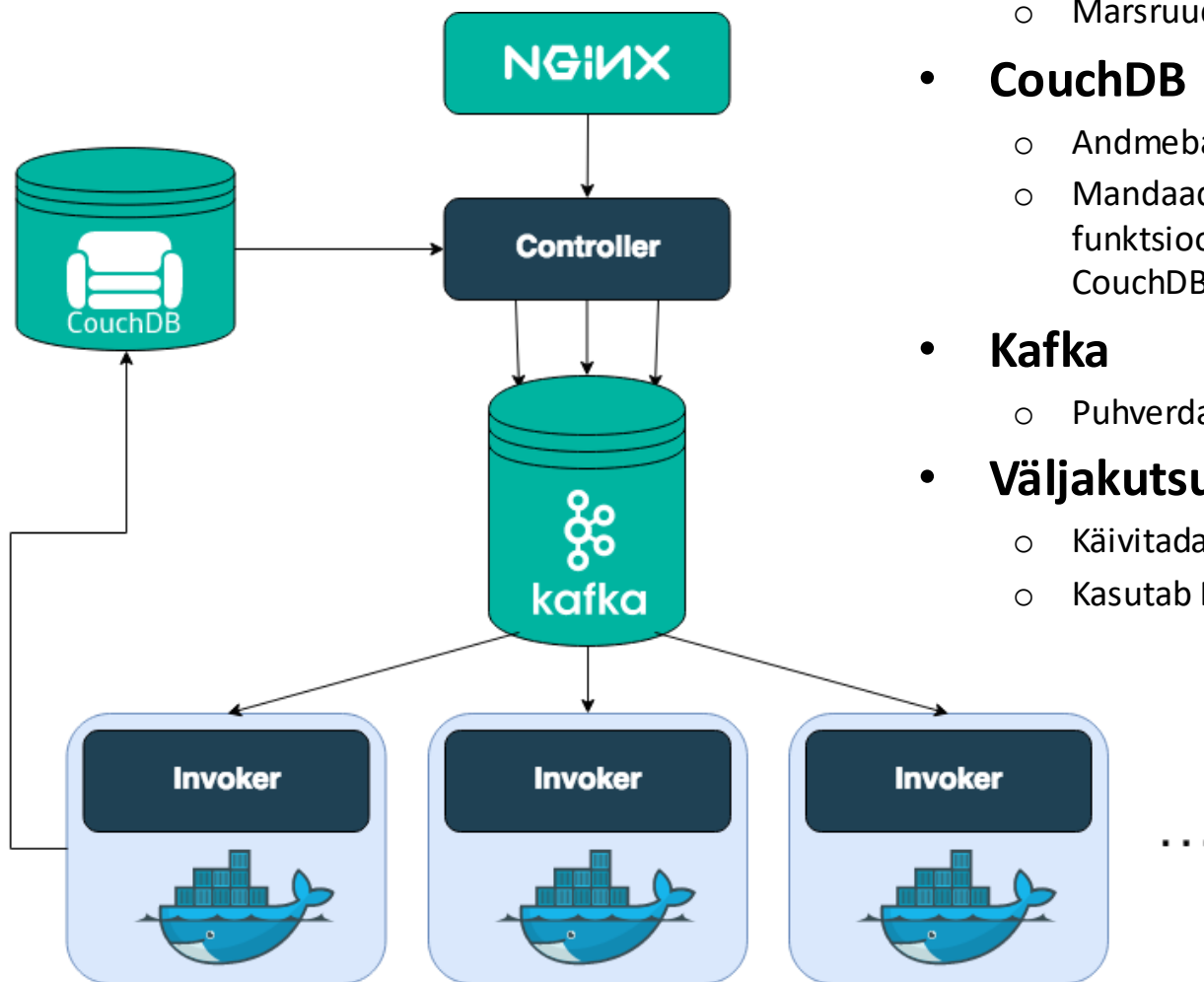
- Avatud lähtekoodiga nanoteenuste platvorm
- IBMi algatatud, kuid nüüdseks Apache projekt
- Seda kasutab sisemiselt IBM Bluemix FaaS teenus
- Põhineb sündmusel, päästikul ja reeglitel
- Toetab kõiki programmeerimiskeeli\*
  - JavaScript, Swift, Python, PHP, Java, Docker\*, binaarsed käsud\*

# Apache OpenWhisk



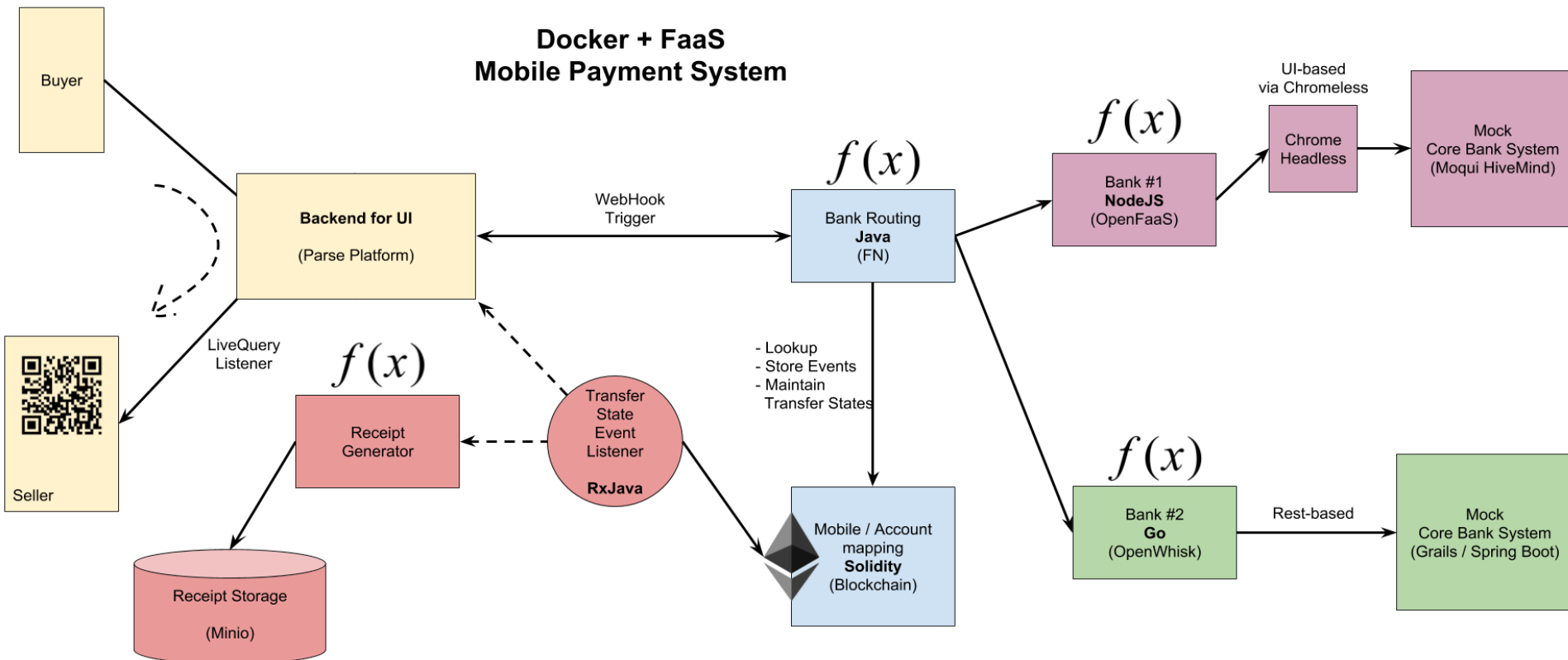


# Apache OpenWhisk



- **Nginx**
  - Teeb kentidele kättesaadavaks nii OpenWhisk kui ka funktsioonide HTTP(S) API otspunktid
  - Iga päring läbib selle kihi
- **Kontroller**
  - Teostab iga päringu autentimist ja autoriseerimist
  - Implementeerib OpenWhisk API
  - Marsruudib päringuid
- **CouchDB**
  - Andmebaas süsteemi seisukorra säilitamiseks.
  - Mandaadid, metaandmed, nimeruumid, funktsioonid, päästikud ja reeglid salvestatakse CouchDB-sse.
- **Kafka**
  - Puhverdab ja jaotab Kontrolleri saadetud sõnumid
- **Väljakutsuja (Invoker)**
  - Käivitab funktsioone, tegevusi
  - Kasutab Dockerit

# OpenWhick FaaS näide



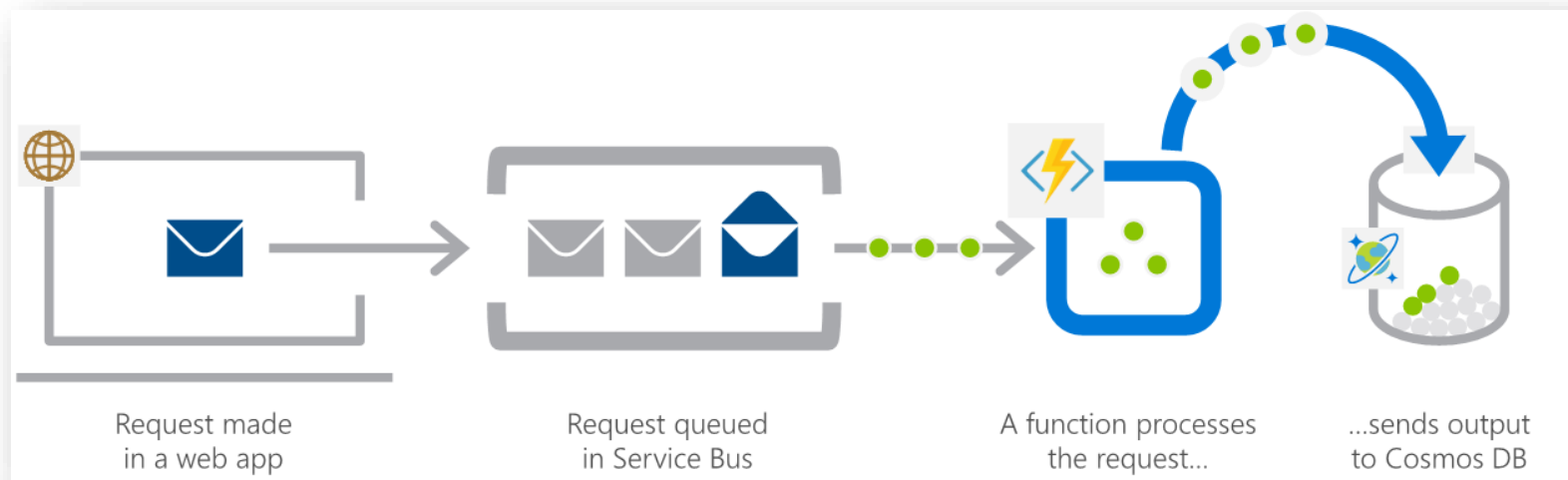
# Azure Functions

- Kasutab sündmustepõhist mudelit
- Päästiku abil käivitatakse "funktsioon"
- Funktsioonide käituskeskkond on [Azure Function Host](#).
  - Built upon the [Azure WebJobs SDK](#)
- Programmeerimiskeeled: C#, Java, JavaScript, PowerShell, Python.

	Functions	WebJobs with WebJobs SDK
Serverless app model <sup>Ⓢ</sup> with automatic scaling	✓	
Develop and test in browser	✓	
Pay-per-use pricing	✓	
Integration with Logic Apps	✓	
Trigger events	Timer Azure Storage queues and blobs Azure Service Bus queues and topics Azure Cosmos DB Azure Event Hubs HTTP/WebHook (GitHub, Slack) Azure Event Grid	Timer Azure Storage queues and blobs Azure Service Bus queues and topics Azure Cosmos DB Azure Event Hubs File system <sup>Ⓢ</sup>
Supported languages	C# F# JavaScript Java Python PowerShell	C# <sup>1</sup>

<https://learn.microsoft.com/en-us/azure/azure-functions/functions-compare-logic-apps-ms-flow-webjobs#compare-functions-and-webjobs>

# Azure Functions – Some examples



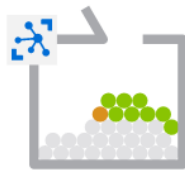
Jaemüügi poe taustateenus: võtab järjekorrast veebitellimusi, töötleb neid ja salvestab saadud andmed andmebaasi.

# Azure Functions – Some examples

Connected IoT devices  
producing data



Data sent to  
IoT Hub



Data with special  
condition routed  
to a function



A function  
processes  
message...



...and calls  
Logic Apps



...which  
invokes  
Zendesk...



...to request  
device repair



Asjade interneti tagateenused tootmises: tootmisettevõte kasutab asjade interneti oma masinate jälgimiseks. Funktsioonid tuvastavad anomaalseid andmeid ja saadavad teenindusosakonnale teate, kui on vaja remonti teha.

# Nanoteenuste eelised

- Väga lihtne skaleerida
- Prototüüpimine on kiire
- Lihtne modifitseerida, ümber disainida, asendada
- Maksame ainult selle eest, kui kaua võttis aega funktsiooni käivitada
  - Ei maksa *idle* aja eest
- Saab kombineerida hulga väiksemaid, erinevates keeltes kirjutatud funktsioone suuremaks rakenduseks

# Puudused

- Raskem vältida kindlast platvormist sõltuvust (vendor lock-in)
  - Sõltub suuresti sisseehitatud päästikutest ja reeglitest
  - Kui toetume S3 päästikul (nt S3 faili muudeti), kas saame sarnast päästikut kaustada teisel platvormil
- Monitoorimise ja silumise tööriistade puudumine
- Komposeerimise keerukus
- Külmkäivitus – Cold start
- Olekupõhised arvutused
- Kulusid on raskem ennustada
- Võib olla palju kallim kui ei ole üldse *Idle* aega

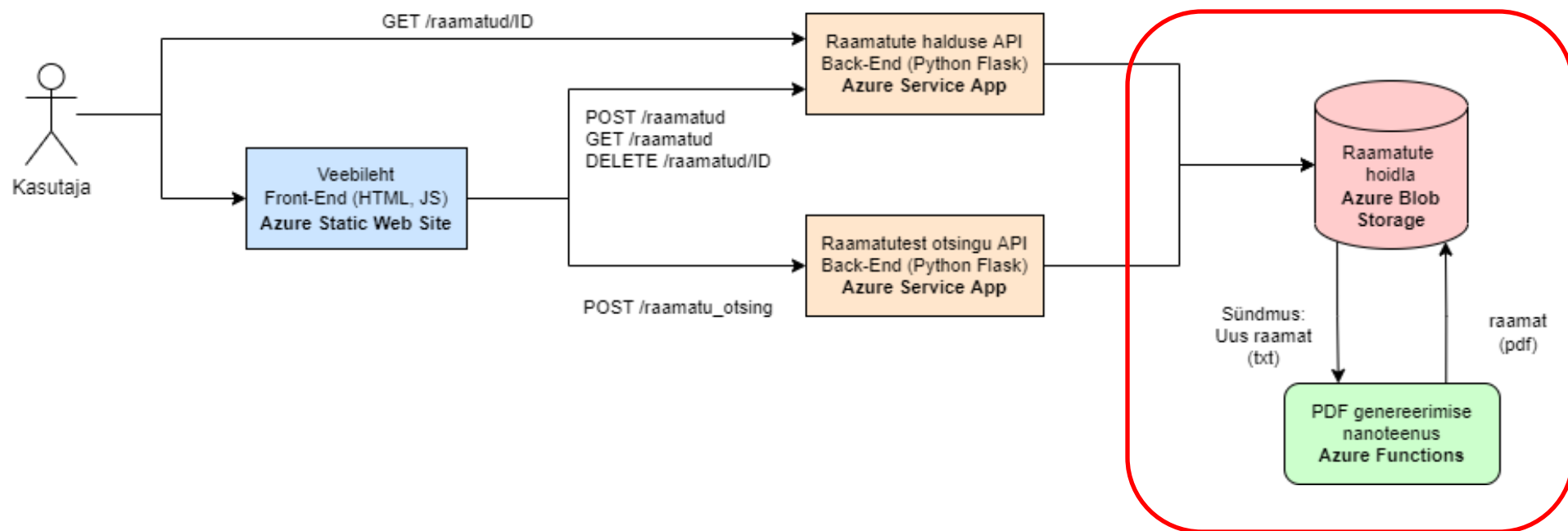
# Kokkuvõte

- Mikroteenused lihtsustavad hajussüsteemide loomist, individuaalsete komponentide uuendamist ning skaleerimist
- Nanoteenused on loogiline jätk mikroteenustest väiksemaks minemisel
  - Teenused on "lihtsalt" väiksemad
- Sündmuste põhine käivitamine vs pidevalt taustal jooksvad teenused
- Mikro- ja nanoteenused lihtsustavad pidevat arendust, pidevat käitlemist ja pidevat integreerimist
  - *Continuous development, CI/CD*



# Praktikum

***Loome Azure funktsiooni raamateute failide PDF'iks genereerimiseks***



# Järgmine loeng

- Pilvepõhiste rakenduste arhitektuurid