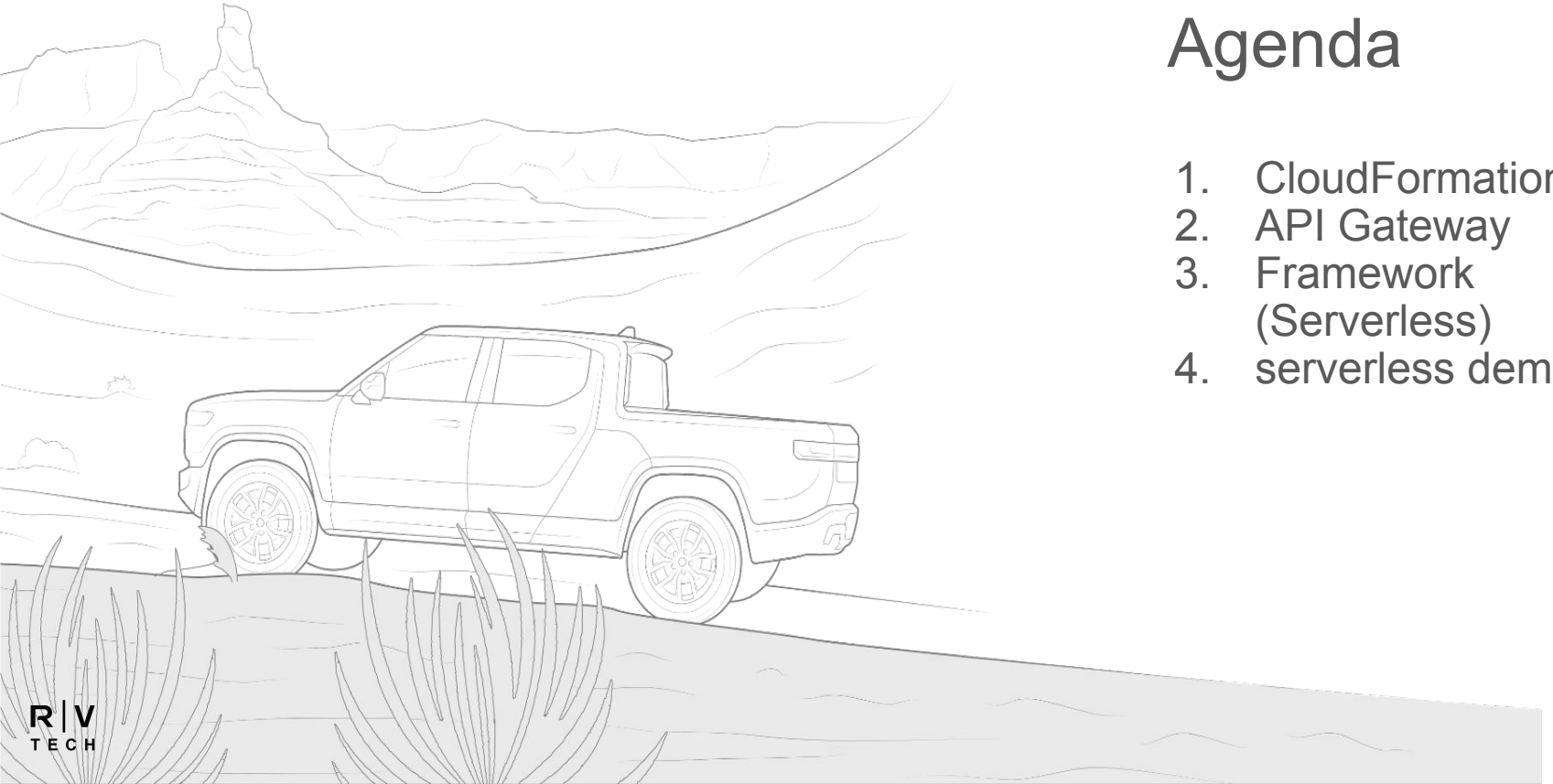

Dobro došli



Agenda

1. CloudFormation
2. API Gateway
3. Framework (Serverless)
4. serverless demo

R|V TECH

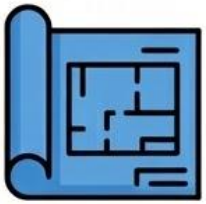
IaC koncept i zašto?

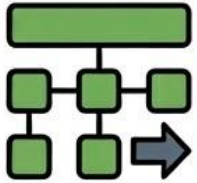
- Infrastruktura se tretira kao programski kod
- Omogućava automatizaciju i upravljanje infrastrukturom
- IaC garantuje doslednost i ponovljivost okruženja
- Verzioniranje konfiguracije

R|V TECH

CloudFormation Koncepti

- Template (Šablon)
- Stack (Stek)
- Change Sets
- Drift Detection





Templates

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Description": "A sample template",
    "Resources": {
        "MyEC2Instance": {
            "Type": "AWS::EC2::Instance",
            "Properties": {
                "ImageId": "ami-0ff8a91507f77f867",
                "InstanceType": "t2.micro",
                "KeyName": "testkey",
                "BlockDeviceMappings": [
                    {
                        "DeviceName": "/dev/sdm",
                        "Ebs": {
                            "VolumeType": "io1",
                            "Iops": 200,
                            "DeleteOnTermination": false,
                            "VolumeSize": 20
                        }
                    }
                ]
            }
        },
        "MyEIP": {
            "Type": "AWS::EC2::EIP",
            "Properties": {
                "InstanceId": {
                    "Ref": "MyEC2Instance"
                }
            }
        }
    }
}
```

```
AWSTemplateFormatVersion: 2010-09-09
Description: A sample template
Resources:
  MyEC2Instance:
    Type: 'AWS::EC2::Instance'
    Properties:
      ImageId: ami-0ff8a91507f77f867
      InstanceType: t2.micro
      KeyName: testkey
      BlockDeviceMappings:
        - DeviceName: /dev/sdm
          Ebs:
            VolumeType: io1
            Iops: 200
            DeleteOnTermination: false
            VolumeSize: 20
  MyEIP:
    Type: 'AWS::EC2::EIP'
    Properties:
      InstanceId: !Ref MyEC2Instance
```

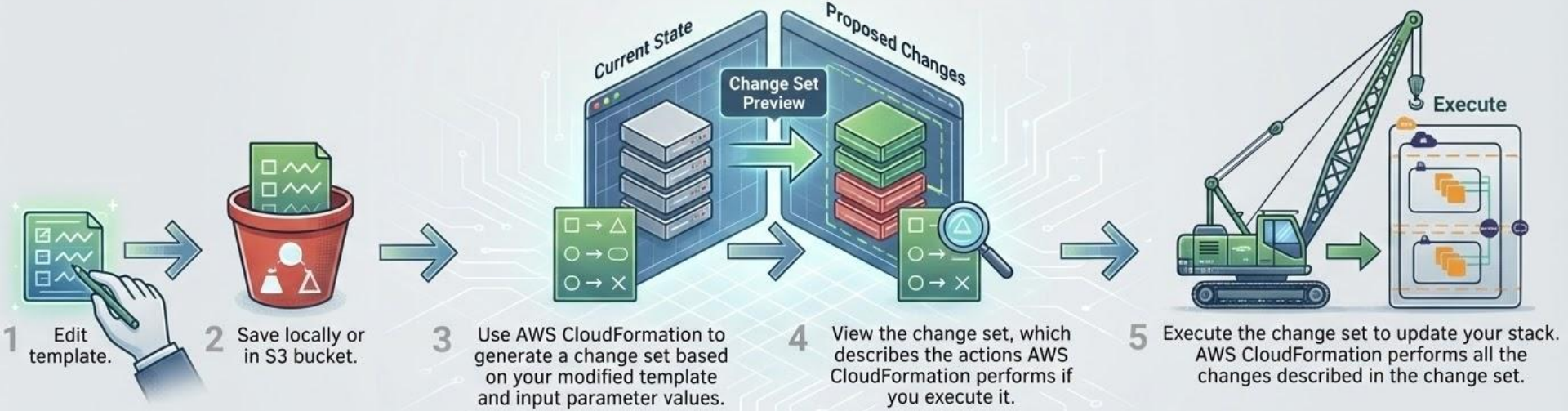
R|V TECH

Stack

- Jedna instanca template-a
- Brisanjem stacka se brišu svi resursi

Change Set

- Git diff
- Skup promena koje će biti izvršene nakon promene template-a



Rollback i Stack Policy

- Rollback

  Vraća promene na prethodno uspešno stanje

- Stack Policy
- Štiti ključne resurse od nenamernog brisanja
- Definišete pravila za update i brisanje resursa

- Osigurava kontinuitet usluge tokom promena infrastrukture

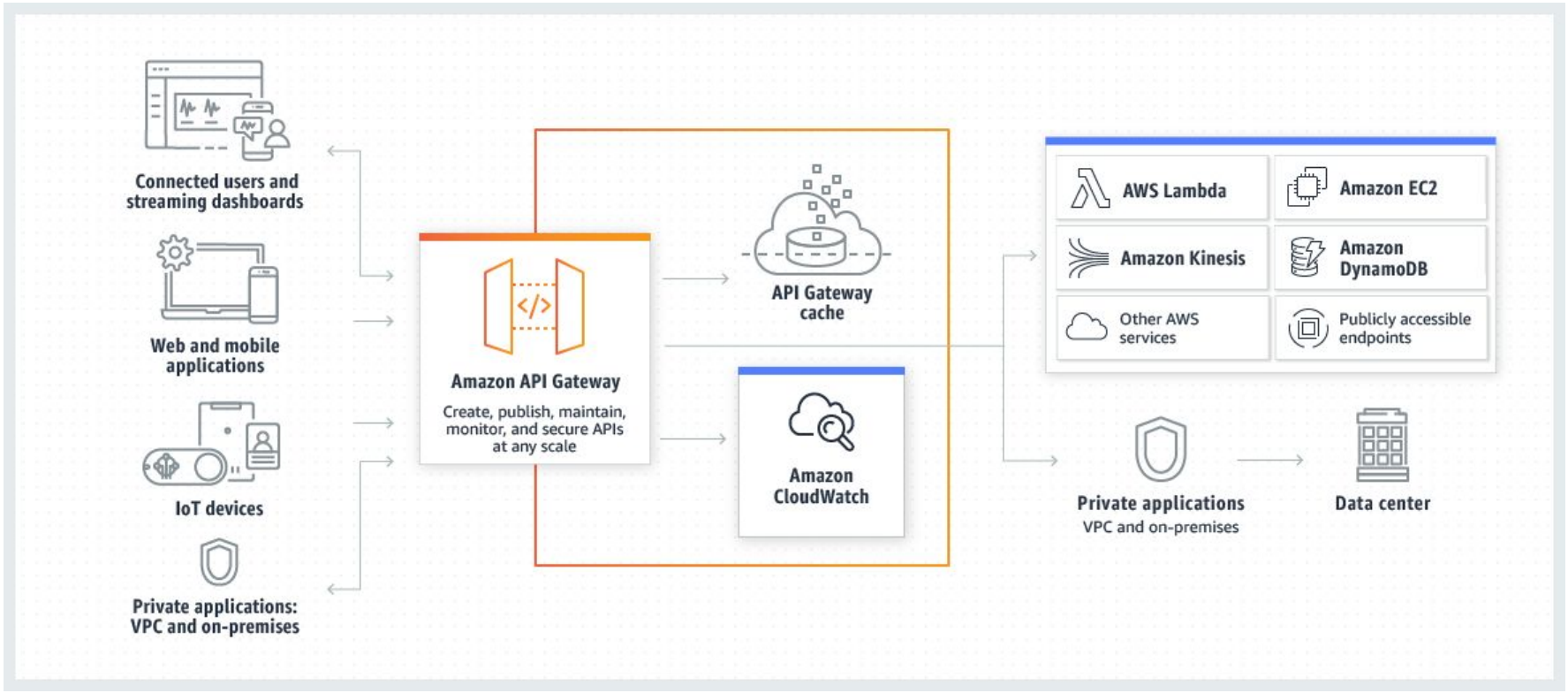
R|V TECH

API Gateway [koncept]

- još jedan serverless servis (ne vidiš servere, skalira se, pay-per-request)
- olakšava kreiranje i održavanje API (Application Programming Interface)
- posrednik između klijenata i nekog našeg backend sistema (EC2, Lambda, DynamoDB)

- funkcionalnosti:
  - usmeravanje saobraćaja
  - autentifikacija i autorizacija (Lambda Authorizer, JSON Web Token u header…)
  - zaštita od previše zahteva (throttling)

API Gateway [koncept]



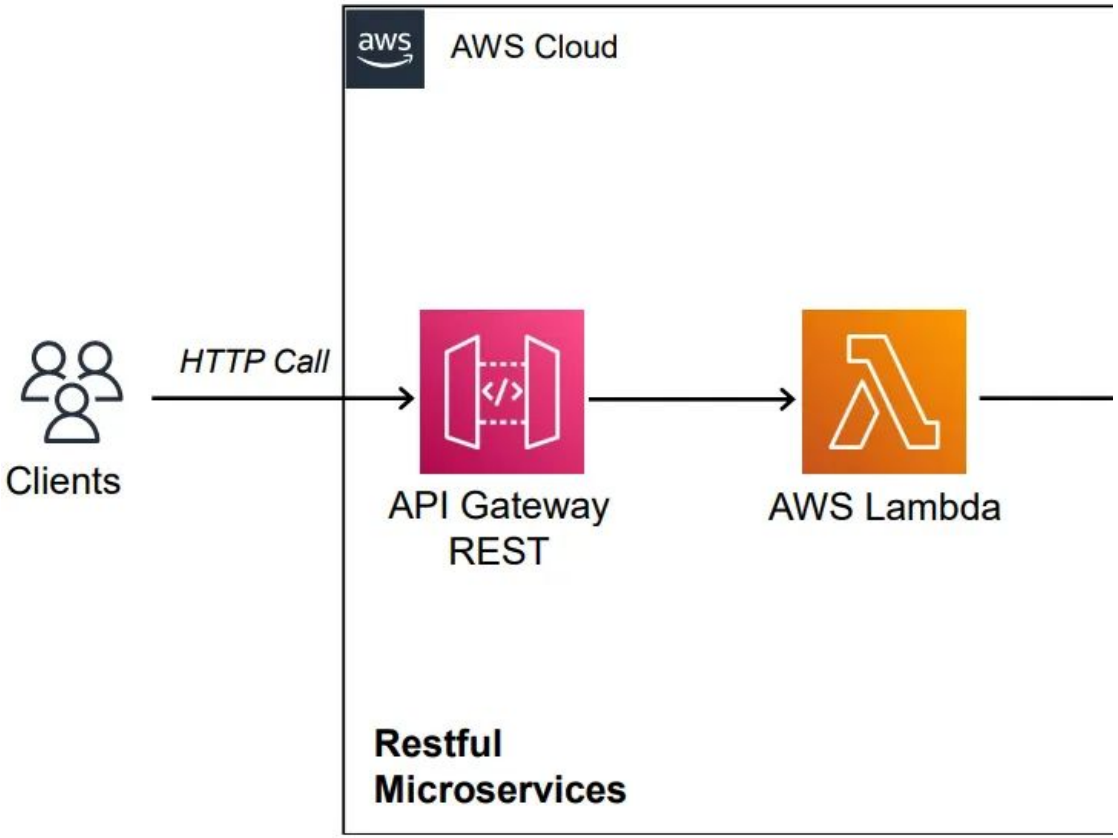
API Gateway [koncept]

- **REST API**
  - dosta funkcionalnosti (keširanje, throttling…)
- **HTTP API**
  - brži i jeftiniji
  - dizajniran za lambda funkcije
- **WebSocket API** (dvosmerna komunikacija, stateful)

- **Edge-optimized** (globalno)
  - manji latency
- **Regional** (lokalno)
  - API živi u istom regionu kao i serveri
  - manji latency za lokalce, brži deploy
- **Private** (VPC)
  - samo iz privatne mreže se pristupa
  - nema ga javno
  - interni alati i mikroservisi ga koriste

R|V TECH

API Gateway [click-ops primer]



CloudFormation [click-ops primer]

Serverless ϟ Framework

- ALAT za IaC (Infrastructure as Code)
- fokus na kod aplikacije, ne na infrastrukturu (ali je tu za oboje) (automatski kreira IAM role, logove i neka podešavanja koja ne moramo mi)
- pomaže nam da iz neke naše napisane “namere” šta želimo, dobijemo odgovarajuće resurse na AWSu
- bilo koji AWS resurs može da se kreira kroz njega koristeći CloudFormation sintaksu (u okviru **resources** dela)

- instalacija: https://www.serverless.com/framework/docs/getting-started
  - npm (Node Package Manager)

R|V TECH

Serverless ϟ Framework

- Functions (funkcije)
  - definišu poslovnu logiku
  - jedna definicija = jedna AWS Lambda
  - mapiranje
- Events (događaji)
  - okidači koji pokrenu funkciju
  - automatski se kreira potrebna infrastruktura (http event -> API Gateway)
- Resources (resursi)
  - čist CloudFormation kod za kreiranje infrastrukture koja se ne pravi automatski
- Plugins

```yaml
service: matf-serverless-3
frameworkVersion: '3'

provider:
  name: aws
  runtime: python3.9

# you can overwrite defaults here
#  stage: dev
#  region: us-east-1

functions:
  dobar_dan:
    handler: handler.dobar_dan
    events:
      - http:
          path: /
          method: get
```

```yaml
# 1. Prvo definišemo da koristimo plugin
plugins:
  - serverless-localstack

# 2. Onda ga konfigurišemo (opciono, ali često potrebno)
custom:
  localstack:
    stages:
      - local  # Plugin se aktivira samo kad radimo "deploy --stage local"
    host: http://localhost  # Gde se vrti LocalStack
    edgePort: 4566
```

R|V TECH

Serverless ϟ Framework

- smanjuje **vendor lock-in** (podržava AWS, Microsoft Azure, Google Cloud Platform…)
- nastao je za razvoj aplikacija na AWS Lambdama -> podrška za AWS dosta jaka
- konfiguracija ne mora samo da se piše u **yaml** formatu, podržani su i json/ts/js

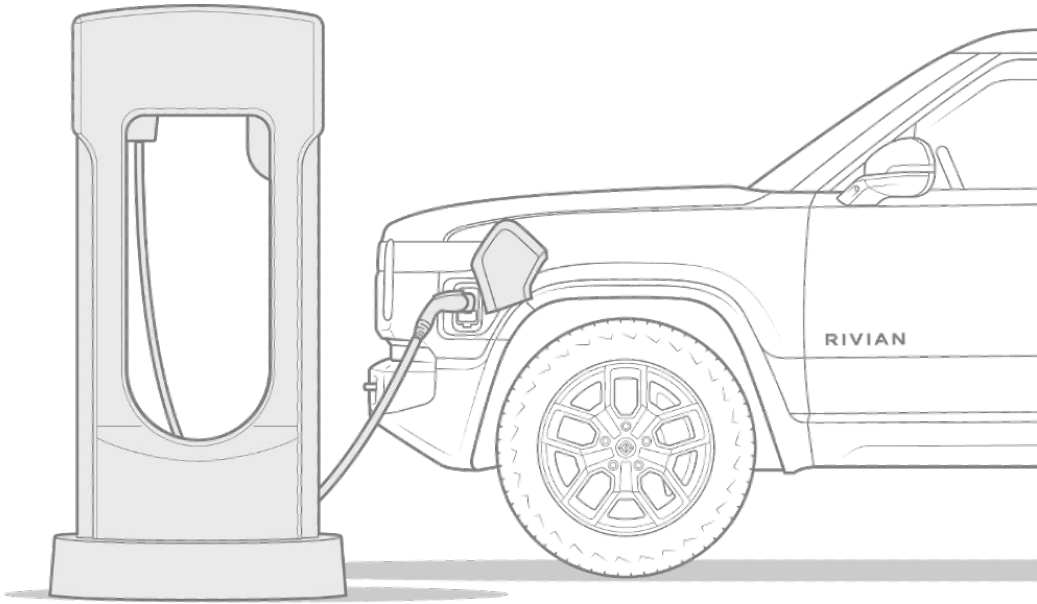
- **localstack** podrška:
  https://docs.localstack.cloud/aws/integrations/app-frameworks/serverless-framework/

Serverless ϟ Framework

DEMO U SANDBOXu

- **serverless create** --template aws-python3 --path matf-serverless-3
- **serverless package**
- **serverless deploy**
- **serverless remove**

PITANJA



R|V
TECH