IMPLANTANDO NA AWS UMA APLICAÇÃO COM LAMBDA E DYNAMODB

ROTEIRO

- Objetivos e limitações do escopo
- Resumo da aplicação demo
  - Visão geral dos endpoints
  - Execução da aplicação
  - Visão geral da arquitetura
  - CRUD com o `DynamoDBMapper`
  - Provisionamento dos recursos necessários
- Recursos AWS utilizados
  - AWS CLI
  - AWS SAM CLI
  - Simple Storage Service (S3)
  - CloudFormation
  - Lambdas
  - DynamoDB
- Hands-on AWS
- Para se aprofundar

OBJETIVOS

- Implantar na AWS uma aplicação REST que usa funções Lambdas para manipular dados do DynamoDB
  - Apresentar os conceitos dos recursos envolvidos
  - Apresentar um passo-a-passo prático com aplicação dos conceitos apresentados

LIMITAÇÕES DO ESCOPO

- A aplicação demo apresentada não contempla:
  - O uso do framework Quarkus
  - O uso da arquitetura hexagonal ou *service layer*

RESUMO DA APLICAÇÃO DEMO

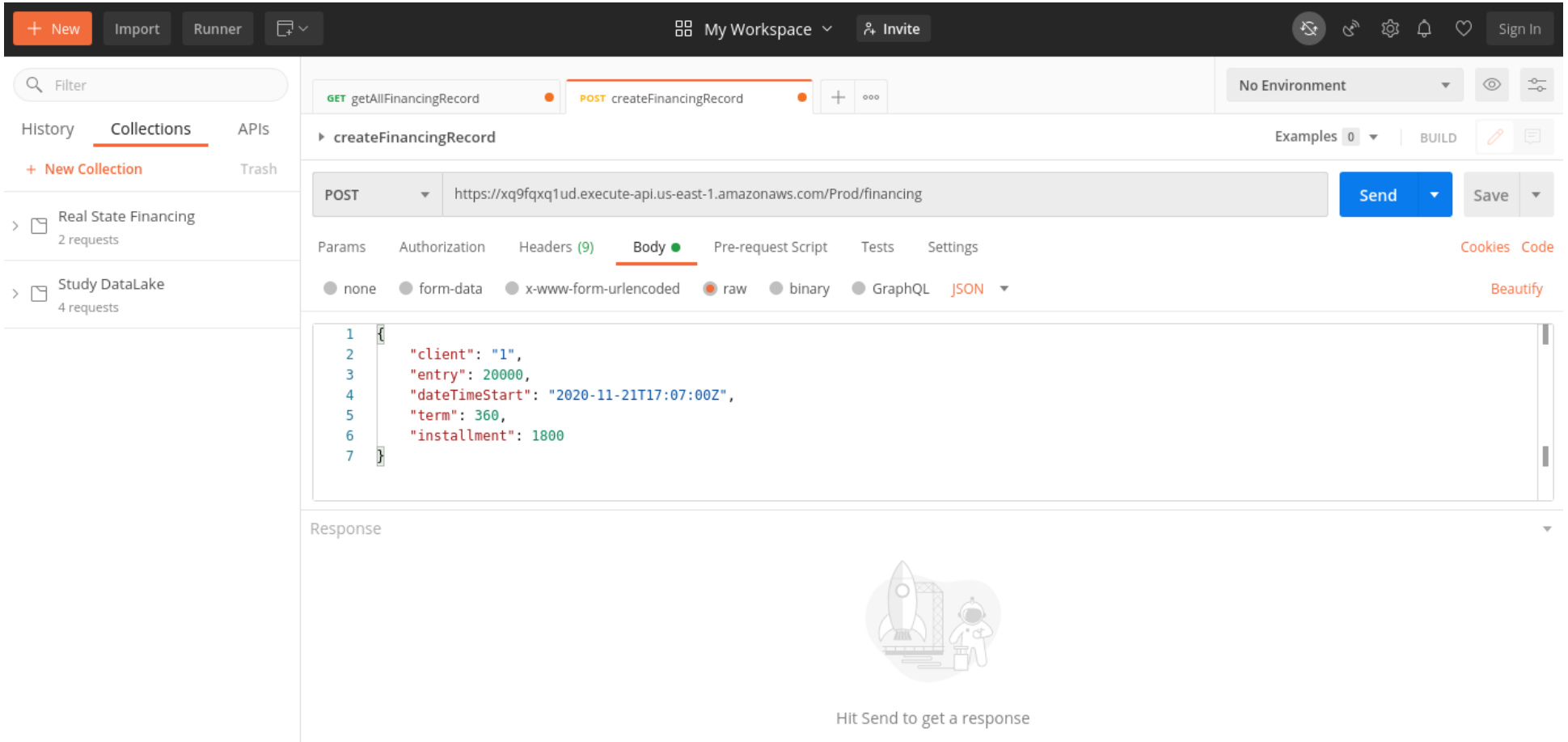
VISÃO GERAL DOS *ENDPOINTS*

VERBO	URI	AÇÃO	LAMBDA
POST	/financing	criar	CreateFinancingRecordFunction
GET	/financing/{cliente}	mostrar	GetFinancingRecordsByClientFunction

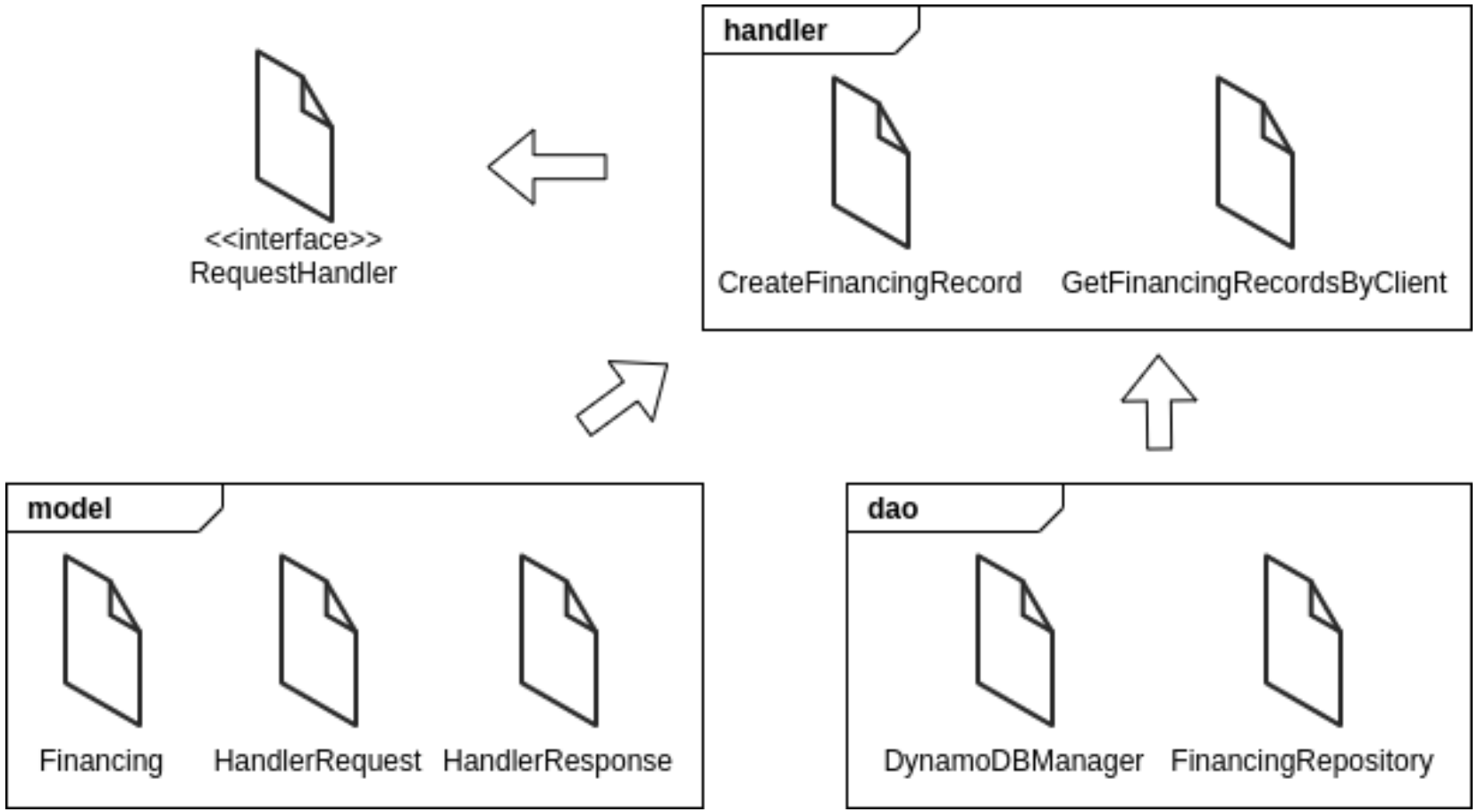
Código da aplicação disponível em:

https://github.com/villani/real-estate-financing

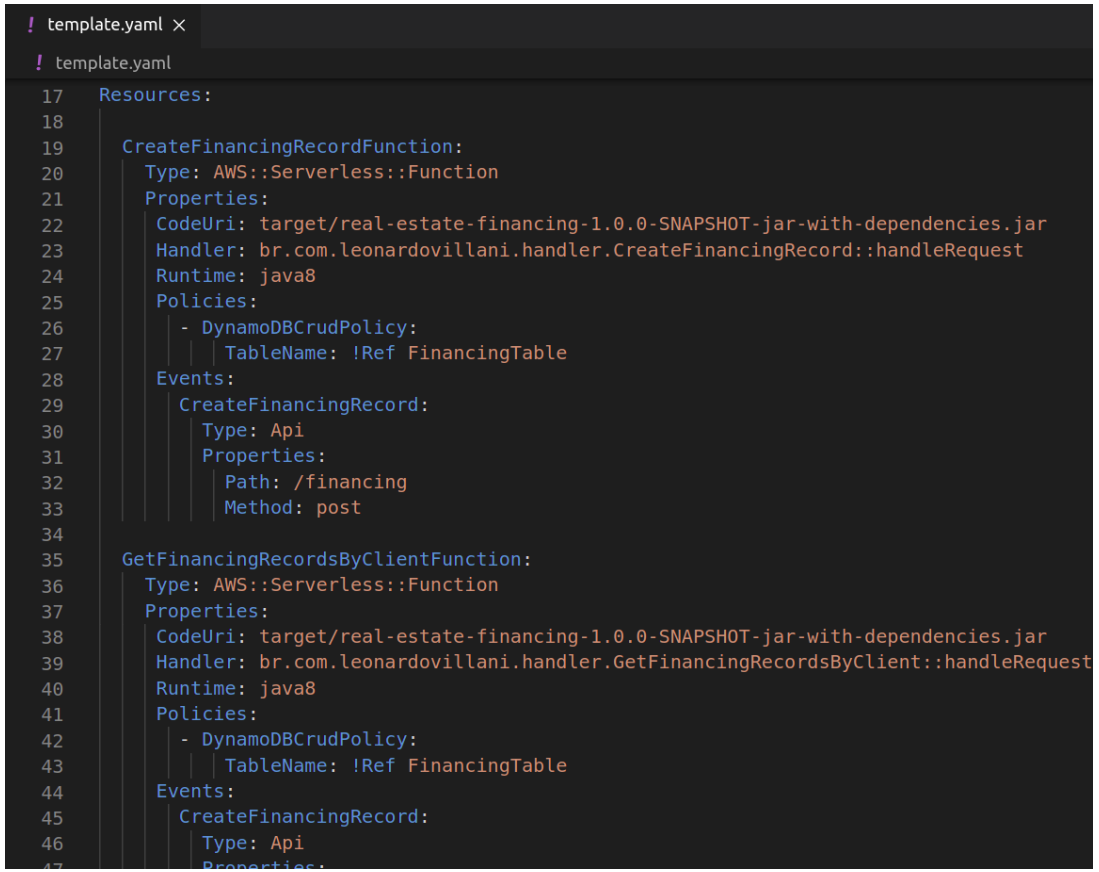
EXECUÇÃO DA APLICAÇÃO



VISÃO GERAL DA ARQUITETURA



PROVISIONAMENTO DOS RECURSOS NECESSÁRIOS



- Arquivo template.yaml na raiz do projeto
- Contém as definições dos recursos que devem ser provisionados
- Contém também os endpoints que serão usados para acionar as lambdas

CRUD COM O DYNAMODBMAPPER

DynamoDBMapper	
Create	save
Read	query, load
Update	save
Delete	delete

```java
public Financing save(final Financing Financing) {
    mapper.save(Financing);
    return Financing;
}

public List<Financing> findByClient(final String client) {

    final Map<String, AttributeValue> eav = new HashMap<String, AttributeValue>();
    eav.put(":val1", new AttributeValue().withS(client));


    final DynamoDBQueryExpression<Financing> queryExpression = new DynamoDBQueryExpression<Financing>()
            .withKeyConditionExpression("client = :val1").withExpressionAttributeValues(eav);

    final List<Financing> financing = mapper.query(Financing.class, queryExpression);

    return financing;
}
```

Mais exemplos disponíveis em:

https://docs.aws.amazon.com/pt_br/amazondynamodb/latest/developerguide/DynamoDBMapper.CRUDExample1.html

RECURSOS AWS UTILIZADOS

AWS CLI

- Uma ferramenta para gerenciar serviços AWS via linha de comando
- Possibilita automatizar o gerenciamento desse serviços por meio de scripts
- É necessário verificar se as credencias estão dentro da validade antes de usar

- Exemplo de conteúdo em ~/.aws/credentials

```
[default]
aws_access_key_id=ASIAZO2CDVCEJJTAZWL2
aws_secret_access_key=AsAVZzpz7TsLzzGd8+82SgM94zgPA16V
zpd+Qp5h
aws_session_token=FwoGZXIvYXdzEHgaDHGMZqn23OodiwpinyLE
AfMml7YwzvoadHWb3NOW5ZwcfGDmQyv70SiSoaczISB1k79TVIw2Iy
JgZuyFKrA9EL9FB6MEHYoiSVZ/vIBJc7P5a9mTrKLRR463z+06x9ii
P8SrW2V/rdHSMBEbXze+xEDkSGzNKMxpql+fmCC5UdX6Q4p6gXAm6V
HUW3Y/Llvin46CfoFZgefFEJ5qV+EN/8RlOFtBTXblcJsmCgK1e25n
pTRP4x1ViSDrqkYNX1ykKqXtAxTRJinbw1euuYPV0rdM/GQo5If2/Q
UyLR29Bg+G6tseqkuQiCDJqVzzeIhTNEZMxD9OUvF3ift8mqHRFHGU
Rj87IgTPFQ==
```

AWS CLI

- Uma ferramenta para gerenciar serviços AWS via linha de comando
- Possibilita automatizar o gerenciamento desse serviços por meio de scripts
- É necessário verificar se as credencias estão dentro da validade antes de usar

- Exemplo de comando utilizado:

```
aws s3 mb s3://$BUCKET_NAME
```

➢ Cria um bucket no S3

AWS SAM CLI

- *Servless Application Model* – Modelo de Aplicações sem Servidor
- Funções lambda, fontes de eventos e outros recursos são combinadas para realizar tarefas
- O objetivo é diminuir a preocupação com gerenciamento de infraestrutura do servidor, como provisionamento e correção da sua capacidade
- A AWS SAM CLI é um ferramenta que permite criar esse tipo de aplicação via linha de comando
  - Ela possibilita a execução local de funções lambdas, bem como o empacotamento e implantação dessas implantações

AWS SAM CLI

Para empacotar uma aplicação

```
sam package \
    --template-file template.yaml \
    --output-template-file packaged.yaml \
    --s3-bucket $BUCKET_NAME
```

Para implantar uma aplicação

```
sam deploy \
    --template-file packaged.yaml \
    --stack-name study-datalake \
    --capabilities CAPABILITY_IAM
```

SIMPLE STORAGE SERVICE (S3)

- Um serviço de armazenamento de objetos imutáveis
- Os repositórios são chamados de buckets
- O S3 foi usado para armazenar os códigos compilados da aplicação apresentada

- Exemplos de uso

  - Criação do bucket

```
aws s3 mb s3://$BUCKET_NAME
```

  - Uso do bucket

```
sam package \
    --template-file template.yaml \
    --output-template-file packaged.yaml \
    --s3-bucket $BUCKET_NAME
```

CLOUDFORMATION

- Um serviço na AWS que auxilia na modelagem e configuração dos recursos a serem utilizados por uma aplicação
- Em um arquivo, em formato JSON ou YAML, é possível descrever todos os recursos e configurações necessárias a uma aplicação
- Nesta aplicação, o AWS SAM CLI foi o recurso utilizado para consumir esse arquivo e criar a *stack* (pilha) visível no painel do CloudFormation

Exemplos de uso:

```
sam package \
    --template-file template.yaml \
    --output-template-file packaged.yaml \
    --s3-bucket $BUCKET_NAME
```

```
sam deploy \
    --template-file packaged.yaml \
    --stack-name study-datalake \
    --capabilities CAPABILITY_IAM
```

AWS LAMBDA

- Um serviço de computação que permite executar um código sem provisionar ou gerenciar servidores
- É necessário que o código seja escrito em uma das linguagens suportadas pelo serviço: Node.js, Python, Ruby, Java, Go e C#
- Em Java, a classe que define uma função Lambda deve implementar a interface **`RequestHandler`** e o respectivo método **`handleRequest`**

DYNAMODB

- Um serviço de banco de dados NoSQL
- Na aplicação demo foi utilizada a AWS SDK para:
  - Definir o modelo de objeto relacional (ORM) por meio de anotações
  - O CRUD da tabela criada

HANDS-ON AWS

PARA SE APROFUNDAR

PARA SE APROFUNDAR

- Repositório da aplicação:
  - https://github.com/villani/real-estate-financing

- Repositório de referência para a construção da aplicação
  - https://github.com/iworks-education/study-datalake

- Repositório com exemplo de CRUD usando Lambda e Quarkus
  - https://github.com/aws-samples/aws-quarkus-demo