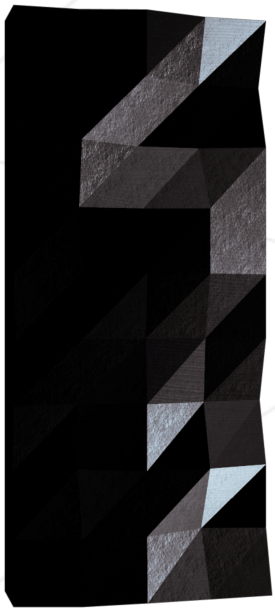

Integração em Nuvem

Módulo 3

Preâmbulo: Neste módulo vamos explorar o serviços 'Simple Storage Service' e 'DynamoDB'.

Versão: 1.0

Sumário

Capítulo I

	Exercício : 01
Buckets	
Pasta de entrega : $ex01/$	
Arquivos para entregar : `create_buckets.sh list_buckets.sh`	
Funções ou bibliotecas autorizadas : `Não se aplica`	

Neste exercício, vamos continuar explorando a ‘Command Line Interface - cli‘ da AWS para provisionar e interagir com seus serviços. Vamos, ainda, utilizar as chaves padrão do seu ambiente dev, se conectando, assim, com o ‘localstack‘.

O Amazon S3 - Simple Storage Service - é um serviço de armazenamento de objetos da AWS. Os dados no S3 são organizados em Buckets, que funcionam como "pastas"onde você pode armazenar arquivos (objetos) com segurança. Cada bucket possui um nome exclusivo globalmente e pode conter desde pequenos arquivos de texto até grandes conjuntos de dados, como imagens e vídeos.

1. Neste exercício, você deverá criar o script `create_buckets.sh`, que utilizará a aws cli para:
   - Criar um ’bucket’ chamado ’42sp-SEU_LOGIN-bucket’
   - Aguardar a criação do buckets.
2. Você também deve criar o script `list_buckets.sh` que liste os buckets em sua conta.

Utilizaremos o Localstack para criar e testar a criação de nossos buckets, assim como durante a avaliação.

Explore os comandos da ‘aws s3api‘, principalmente o ‘delete-bucket‘ de forma a excluir os buckets criados e demonstrar o funcionamento de seus scripts durante a avaliação. Os comandos da ‘aws s3‘ também podem ser utilizados para excluir os buckets mesmo que contenham objetos.

Capítulo II

	Exercício : 02
Objetos	
Pasta de entrega : $ex02/$	
Arquivos para entregar : `upload_object.sh remove_object.sh list_objects.sh download_object.sh`	
Funções ou bibliotecas autorizadas : `Não se aplica`	

No Amazon S3, os dados são armazenados como objetos dentro de buckets. Um objeto é composto por dados, uma chave única e metadados. Para trabalhar com objetos, você pode utilizar a `aws cli` com o comando s3 ou s3api para realizar operações como upload, remoção e listagem.

1. Neste exercício, você deverá criar os scripts:
   - `upload_object.sh` que insere um arquivo pré-determinado por você em um `bucket`.
   - `remove_object.sh`, que remova o objeto correspondente.
   - `list_objects.sh`, que liste os objetos de um `bucket`.
   - `download_object.sh`, que faça o download do objeto inserido em seu `bucket`.

`Para este exercício, não é necessário passar o nome do arquivo e do bucket como argumento de linha de comando ao seu script. Estes dados podem ser configurados internamente em seu script.`

Capítulo III

	Exercício : 03
Upload de arquivos na nuvem	
Pasta de entrega : $ex03/$	
Arquivos para entregar : `decode.py`	
Funções ou bibliotecas autorizadas : `Biblioteca Padrão, boto3`	

Modifique seu programa `decode.py`, criado no módulo anterior, para salvar a imagem decodificada em um bucket s3. Seu programa deve aceitar, como argumentos de linha de comando, o nome do bucket e o caminho da imagem, codificada em base64. Você deve utilizar a biblioteca boto3.

Exemplo de comportamento esperado:

```
?> python3 decode.py "42sp-SEU_LOGIN-bucket" "sample_images/image.txt"
'image.jpg' saved on bucket '42sp-SEU_LOGIN-bucket'!
```

Capítulo IV

	Exercício : 04
	DynamoDB Table
Pasta de entrega : $ex04/$	
Arquivos para entregar : `create_table.sh [table_config.json]`	
Funções ou bibliotecas autorizadas :	

Crie um script `create_table.sh` que utilize o `aws cli` para criar uma tabela chamada `Usuarios` no DynamoDB:

1. A tabela deverá conter uma chave primária de nome `id` e tipo `string` e utilizar throughput provisionado com capacidade mínima:
   - ReadCapacityUnits=1
   - WriteCapacityUnits=1
2. O script deverá aguardar a criação da tabela, antes de devolver o prompt.

Durante o desenvolvimento deste exercício, explore também os comandos ‘aws dynamodb delete-table‘, ‘aws dynamodb describe-table‘ e ‘aws dynamodb wait help‘.
Você pode utilizar a opção ‘-no-cli-pager‘ para mostrar o conteúdo completo no terminal, sem paginação.
Opcionalmente, você pode criar um arquivo ‘table_config.json‘ e a opção ‘-cli-input-json‘, ao invés de configurar todos os valores individualmente via argumentos de linha de comando.

O comportamento do seu script deve ser como abaixo:

```
?> ./create_table.sh
Criando a tabela DynamoDB...
{
```

```
    "TableDescription": {
        "AttributeDefinitions": ...
...
}
Aguardando a tabela estar ativa...
Tabela criada com sucesso.
```

Aproveite para pesquisar sobre os custos que incorrem no ‘throughput provisionado‘.

Capítulo V

	Exercício : 05
DynamoDB Crud	
Pasta de entrega : $ex05/$	
Arquivos para entregar : `dynamo_crud.py`	
Funções ou bibliotecas autorizadas : `Biblioteca Padrão, boto3`	

Implemente operações básicas de CRUD em uma tabela DynamoDB com `boto3`:

1. A tabela Usuarios já deve existir.

2. Seu programa deve funcionar com os seguintes comandos:

```
?> python3 dynamo_crud.py load ./users.csv
Carregando dados de ./users.csv

?> python3 dynamo_crud.py retrieve u304
Usuário encontrado: {'name': 'Gag Halfrunt', 'id': 'u304'}
?> python3 dynamo_crud.py delete u304
Usuário u304 removido.

?> python3 dynamo_crud.py update u304 "Novo Nome"
Usuário u304 atualizado para nome = 'Novo Nome'
```

Separe sua lógica em funções especializadas, utilizando ao menos uma função para cada uma dos comandos suportados.

Capítulo VI

	Exercício : 06
DynamoDB + S3	
Pasta de entrega : $ex06/$	
Arquivos para entregar : `dynamo_crud.py decode.py`	
Funções ou bibliotecas autorizadas : `Biblioteca Padrão, boto3`	

Neste Exercício, vamos integrar o DynamoDB com o S3.

Ajuste seus arquivos de modo que, ao carregar o arquivo `users_with_documents.csv`, a imagem seja convertida, salva em seu bucket, e apenas a `Chave` da imagem seja salva no banco de dados.

```
?> python3 dynamo_crud.py retrieve u882
Usuário encontrado: {'name': 'Zaphod Beeblebrox', 'id': 'u882', 'document_key': '882320-2939223-82823.jpg'}
```

utilize a biblioteca ‘uuid‘ para gerar nomes não repetidos, prevenindo que arquivos homônimos sejam sobrescritos.

A biblioteca ’csv’ possui, por padrão, um limite do tamanho de campo que pode ser lido a partir de um arquivo. Como as imagens podem ultrapassar este tamanho, você pode precisar configurar um novo limite: ’csv.field_size_limit(sys.maxsize)’

Capítulo VII

Validação

- Se você chegou até aqui e executou os exercícios com rigor, já é possível validar este módulo com o resultado final de 80% e já detém os conhecimentos necessários para avançar para os módulos seguintes.

- Os próximos exercícios possibilitam alcançar o resultado de 100%, 110% e 125% respectivamente. Avalie a realização destes exercícios considerando a facilidade/dificuldade encontrada na realização dos exercícios anteriores. Busque o equilíbrio entre se desafiar e avançar para os próximos módulos.

Capítulo VIII

	Exercício : 07
Real AWS	
Pasta de entrega : $ex07/$	
Arquivos para entregar : `table_info.json bucket_arn.txt`	
Funções ou bibliotecas autorizadas : `Biblioteca Padrão, boto3`	

1. Provisione a tabela `Usuarios` utilizando o `Ambiente de Sandbox AWS da 42`.

2. Crie um `bucket` com o nome `42sp-SEU_LOGIN-bucket`.

3. Utilize seus arquivos criados no exercício anterior para carregar os dados do arquivo `users_with_documents.csv` remotamente.

4. Descreva sua tabela em um arquivo `table_info.json`. Este arquivo deve conter, ao menos, o ARN (Amazon Resource Name, identificador único e global para qualquer recurso na AWS) de sua tabela.

5. Use o comando `aws s3api` para obter o ARN de seu bucket criado na AWS.

`Utilize o console da AWS para verificar se os ítens foram inseridos com sucesso, tanto na tabela, quanto no ‘bucket‘. Estes passos deverão ser realizados durante a avaliação.`

Capítulo IX

Bônus

	Exercício : 08
Bucket Policies	
Pasta de entrega : $ex08/$	
Arquivos para entregar : `presign.py`	
Funções ou bibliotecas autorizadas : `Biblioteca Padrão, boto3`	

Por padrão, o acesso ao conteúdo de um bucket é privado. Portanto, as URLs de seus objetos não podem ser acessadas diretamente.

1. Usando `boto3`, e o s3 client, crie um script `presign.py` que cria uma URL temporária para seu bucket, com um período de acesso de 10 minutos.

2. Seu script deve se comportar da seguinte forma:

```
?> python3 presign.py 42sp-SEU_LOGIN-bucket 882320-2939223-82823.jpg
https://42sp-SEU_LOGIN-bucket.s3.amazonaws.com/882320-2939223-82823.jpg?X-Amz-Algorithm=...
```

Capítulo X

Bônus

	Exercício : 09
Bucket Policies	
Pasta de entrega : $ex09/$	
Arquivos para entregar : `get-bucket-policy.sh put-bucket-policy.sh policy.json`	
Funções ou bibliotecas autorizadas : `Biblioteca Padrão, boto3`	

Em alguns casos, pode ser útil tornar os objetos de um bucket publicamente acessíveis para leitura — por exemplo, para compartilhar imagens ou arquivos sem exigir autenticação. Para isso, é necessário configurar a política de acesso bucket policy do S3.

Você irá criar scripts para visualizar e aplicar políticas de acesso, e um arquivo contendo uma política de leitura pública. As políticas devem ser aplicadas em seu bucket `42sp-SEU_LOGIN-bucket`.

1. Crie um script `put-bucket-policy.sh` que utilize o arquivo `policy.json` e atualize as políticas de seu bucket para que o acesso de leitura seja público.

2. Crie um arquivo `policy.json` contendo a política pública a ser aplicada pelo script.

3. Comportamento esperado:

```
?> ./get-bucket-policy.sh
{
  "Version": "2012-10-17",
  "Statement": [
    ...
  ]
}

?> ./put-bucket-policy.sh
Policy applied to bucket.
```

Apenas adicionar a política de acesso público ao bucket não é suficiente. Você também deve remover o bloqueio a todo acesso público, criado por padrão junto com o bucket. Seu script também deve realizar esta tarefa.

Você deve ser capaz de demonstrar que as urls de acesso aos seus objetos podem ser acessadas publicamente, sem a necessidade de assiná-las.

Capítulo XI

Entrega e Avaliação entre Pares

- Entregue seu projeto em seu repositório *Git* disponível na página do projeto na intranet.

- O trabalho dentro do seu repositório será avaliado durante a defesa, juntamente com o conteúdo criado na AWS e localmente. O aluno avaliado deverá estar preparado para excluir e recriar os recursos.

- No horário da avaliação, o avaliador se dirigirá à estação de trabalho do aluno a ser avaliado para realizar os testes. Um clone do repositório deverá ser realizado em uma nova pasta, e estes são os arquivos que serão avaliados.

- Ao término da última avaliação, se tudo tiver corrido bem, avaliado e avaliador deverão se certificar que os recursos criados na AWS sejam excluídos.