**Desafio 1 BGC Brasil**

Pedro Vinícius Dousseau dos Santos

**API REST**

Uma API (Application programming interface) pode ser vista como um contrato entre um provedor de dados (Servidor) e um usuário que deseja acessar e/ou manipular estes dados (Cliente). Sendo um pouco mais técnico, se trata de um estilo de arquitetura de software padronizado utilizado para permitir a comunicação entre aplicações de maneira simples e padronizada.

Com o sucesso e aumento do uso das API's na web, alguns protocolos surgiram com o intuito de fornecer aos usuários um conjunto de regras/recomendações definidas para o design dessas aplicações. Entre estes estão SOAP, CML-RPC, JSON-RPC e o REST.

Uma API REST (Representational State Transfer) nada mais é que uma aplicação que está em conformidade com alguns critérios pré-estabelecidos pelo "padrão" REST.

Na prática, você terá uma aplicação A querendo acessar recursos/serviços de uma aplicação B, essa troca de informações irá ocorrer por uma tecnologia intermediária, uma API, desenvolvida pelo time da aplicação B com o intuito de disponibilizar seus recursos para n clientes que queiram acessá-los de maneira segura e padronizada.

Para executar ações na API você utiliza os CRUDS. Em sistemas WEB, o equivalente são os métodos HTTP.

**ReactJS**

O React é uma biblioteca JavaScript para ajudar na criação de interfaces de usuário

Ele faz com que o trabalho do desenvolvedor de implementar as interações realizadas em UIs, normalmente um trabalho muito desgastante e pouco eficiente, se torne uma tarefa mais fácil e sem dor de cabeça.  
Uma de suas principais características é a criação de UIs por meio da quebra de toda a estrutura da aplicação em componentes.

**Redux**

O Redux é uma outra biblioteca para armazenamento de estados de aplicações JavaScript.

Sua principal função é simplificar o gerenciamento dos estados nas aplicações. Basicamente, ele torna a tarefa de compartilhar estados entre vários componentes algo bem fácil e consistente.

**Serverless Framework**

Antes de mais nada, uma arquitetura serverless é um modelo de infraestrutura que abstrai o gerenciamento direto de um servidor pelo desenvolvedor da aplicação que, normalmente, possuiria um certo grau de controle das requisições feitas ao servidor e seria o responsável por cuidar dos recursos utilizados pela máquina.  
  
Ainda que as aplicações serverless não necessitem de um gerenciamento direto no servidor, há um certo trabalho envolvido em coordenar todos os recursos necessários para colocar a aplicação no ar de maneira eficiente.  
  
Diante disso, o Serverless framework é uma ferramenta que facilita o desenvolvimento, deploy, manutenção e segurança dessas aplicações serverless.  
A ferramenta é composta por uma CLI e uma dashboard que juntas, oferecem uma estrutura, automação e boas práticas permitindo assim, a implementação de arquiteturas serverless sofisticadas e rápidas.  
  
Ele não apenas publica os códigos e os eventos que os ativam, mas também é responsável por criar a infraestrutura necessária que eles dependem.

**AWS API Gateway**

Uma API Gateway é uma ferramenta que fica entre o cliente e uma coleção de serviços backend.  
  
A Amazon API Gateway, por sua vez, é um serviço que permite a criação, o deploy, a manutenção, o monitoramento e a segurança de uma REST API altamente escalável e que possibilita o acesso à uma enorme quantidade de serviços da Amazon.  
  
Em uma arquitetura de microsserviços, uma API Gateway pode se tornar extremamente necessária pois uma única requisição poderia necessitar requisitar diversos outros serviços.

**AWS Lambda**

O AWS Lambda é um serviço de computação onde a AWS é a responsável por executar pedaços de código. Ele é utilizado para implementar aplicações que seguem o modelo serverless.  
Cabe ao desenvolvedor configurar quando o código será acionado, isto é, com requisições HTTP, serviços da AWS, eventos de banco de dados, etc...  
  
Ele escala automaticamente e apenas cobra pelo tempo de execução do código escrito.  
  
Com o Lambda, é possível quebrar a aplicação em pequenas funções que possuem uma responsabilidade única. Dessa maneira, é possível utilizá-lo para implementar aplicações voltadas para a arquitetura de microsserviços.

**AWS Step Functions**

A medida que aplicações distribuídas, compostas por várias aplicações serverless, tornam-se mais complexas, a dificuldade para gerenciar todos esses "pedaços" também aumenta.  
  
O AWS Step Functions é um orquestrador de funções serverless que facilita o gerenciamento e o sequenciamento de funções do AWS Lambda através da criação dos fluxos de trabalho.

**AWS DynamoDB**

O Amazon DynamoDB é um banco de dados NoSQL totalmente gerenciado pela AWS, altamente escalável, consistente e rápido. Possui event driven programming que pode ser utilizado para disparar eventos baseados em alterações que por sua vez, poderão ser observados em aplicações serverless.

**AWS RDS**

O AWS RDS (Relational Database Service) por sua vez, é um serviço de banco de dados relacional (SQL) totalmente gerenciado pela AWS. Sua principal função é facilitar a configuração, a operação e a escalabilidade de banco de dados relacionais na nuvem.

**AWS Kinesis Data Streams**

O Amazon Kinesis Data Streams é um serviço de streaming para coleta de dados em tempo real e de grande escala.  
O serviço pode capturar continuamente uma grande quantidade de dados por segundo de inúmeras origens.  
Assim que coletados, esses dados são disponibilizados em pouquíssimo tempo, possibilitando assim, análises em tempo real.

**AWS S3**

O Amazon S3 (Simple Storage Service) é um serviço para armazenamento de objetos na nuvem com capacidade para armazenar e recuperar qualquer volume de dados independentemente do local.  
O serviço oferece recursos de gerenciamento fáceis de utilizar e foi projetado para facilitar a computação escalável na web.

**AWS Cognito**

O Amazon Cognito é um serviço que cuida da parte da autenticação em aplicativos.  
Ele permite adicionar funcionalidades como cadastro, login e controle de acesso com rapidez e facilidade.  
Com o cognito você pode delegar todo o processo de autenticação da sua aplicação para o serviço.

**AWS CloudWatch**

Com a adoção de inúmeros serviços da Amazon pode se tornar um desafio manter o controle de todos os dados e informações destes serviços. O Amazon CloudWatch é um serviço capaz de coletar, acessar e correlacionar todos os recursos de aplicativos da AWS em tempo real, é basicamente um repositório de métricas/ferramenta de log.  
  
Ele oferece uma visão unificada de todos os recursos dos aplicativos e dos serviços executados na AWS.

**AWS Cloudformation**

O AWS CloudFormation IaC (Infrastructure as Code) é um serviço que facilita a modelação e configuração de todos os recursos da AWS.  
  
Com ele, é possível criar um modelo que descreve todos os recursos da AWS desejados e o próprio serviço é responsável pelo provisionamento e configuração dos recursos, simplificando assim, o gerenciamento de infraestrutura.

**Links e referências**

https://www.youtube.com/watch?v=lsMQRaeKNDk

https://www.ibm.com/cloud/learn/api

https://www.ibm.com/cloud/learn/rest-apis

https://www.redhat.com/pt-br/topics/api/what-is-a-rest-api

https://blog.tecnospeed.com.br/rest-x-soap/

https://pt-br.reactjs.org/

https://tableless.com.br/react-o-que-e-e-como-funciona-essa-ferramenta/

https://kenzie.com.br/blog/react/

https://redux.js.org/

https://www.youtube.com/watch?v=DbEpBeZ6yic

https://www.treinaweb.com.br/blog/o-que-e-redux/https://www.serverless.com/

https://serverless-stack.com/chapters/pt/what-is-serverless.html

https://aws.amazon.com/pt/serverless/

https://github.com/serverless/serverless

https://www.serverless.com/framework/docs/providers/aws/guide/intro/

https://docs.aws.amazon.com/apigateway/index.html

https://aws.amazon.com/pt/api-gateway/

https://www.redhat.com/en/topics/api/what-does-an-api-gateway-do

https://www.youtube.com/watch?v=o1XKj73HN\_8

https://www.youtube.com/watch?v=timsDKQDsD0

https://aws.amazon.com/pt/lambda/

https://www.treinaweb.com.br/blog/o-que-e-aws-lambda/

https://hackernoon.com/a-crash-course-on-serverless-with-node-js-632b37d58b44#422a

https://serverless-stack.com/chapters/pt/what-is-serverless.html

https://aws.amazon.com/pt/step-functions/?step-functions.sort-by=item.additionalFields.postDateTime&step-functions.sort-order=desc

https://aws.amazon.com/pt/dynamodb/

https://medium.com/@vinicius\_roc/dynamodb-o-que-voc%C3%AA-precisa-saber-antes-de-usar-1bcad8d31787

https://aws.amazon.com/pt/rds/

https://flexa.cloud/amazon-rds-o-que-e-e-como-funciona-o-relational-database-service-da-aws/

https://aws.amazon.com/pt/kinesis/data-streams/

https://aws.amazon.com/pt/s3/

https://docs.aws.amazon.com/pt\_br/AmazonS3/latest/dev/Welcome.html

https://www.alura.com.br/conteudo/aws-s3-manipule-e-armazene-na-nuvem

https://aws.amazon.com/pt/cognito/

https://medium.com/@lucaslucm/usando-o-aws-cognito-como-servi%C3%A7o-de-autentica%C3%A7%C3%A3o-com-react-e-spring-33b5ac2e7448

https://www.alura.com.br/conteudo/amazon-cognito

https://aws.amazon.com/pt/cloudwatch/

https://docs.aws.amazon.com/pt\_br/AmazonCloudWatch/latest/monitoring/cloudwatch\_architecture.html

https://www.tiespecialistas.com.br/o-que-e-o-amazon-cloudwatch/

https://aws.amazon.com/pt/cloudformation/

https://docs.aws.amazon.com/pt\_br/AWSCloudFormation/latest/UserGuide/Welcome.html

https://guilhermeteles.com.br/o-que-e-o-aws-cloudformation/