Roteiro vídeo fase 3

----------------

1 - Introdução (10-30s)

1.1 - nome curso

1.1 - membros grupo

1.1 - fase do projeto

Olá, somos o grupo G12 do curso de Pós-Graduação em Software Architecture,

Turma 4SOAT da FIAP e estaremos apresentando nosso projeto do Tech Challenge da Fase 3.

O propósito desta fase foi desenvolver uma função serverless para gerenciar a autenticação do sistema. Além disso, conduzimos os deploys em um provedor de nuvem, escolhendo a AWS para hospedar nossa infraestrutura. Nesse processo, empregamos conceitos como pipelines CI/CD para facilitar a criação dos recursos na nuvem por meio do Terraform, assim como para automatizar a compilação e implantação da aplicação.

A seguir mostraremos a arquitetura do sistema na AWS.

-----------------------------------------------------------------------------------------------------------------

2 - Arquitetura

- explicação dos desenhos

- arquitetura k8s

Na arquitetura k8s, para a API e o Worker consumidor da fila de pedidos, utilizamos deployments contendo Probes, HPAs para manter alta disponibilidade e resiliência nos serviços visto que a lanchonete precisaria de todo o ecossistema funcional mesmo em cenários de alta demanda tendo o mínimo ou quase nenhum período de osciosidade devido a escalabilidade dos recursos. Para serviços que dependem de persistencia de dados, como o RabbitMQ, é utilizado Statefulsets pois tem identidade persistente e garantia de DNS estável no cluster, facilitando assim a comunicação de serviços que dependem de um nome fixo, além da persistencia em volumes evitando a perda de dados mesmo após a reinicialização dos pods.

- function lambda

Temos também uma function lambda na aws, que utiliza o Cognito para criar e autenticar usuários de nossa API.

Uma característica única dessa integração é a utilização do CPF como username, proporcionando uma experiência de autenticação simplificada e segura.

A função Lambda atua como um ponto de entrada seguro para nossa aplicação, recebendo requisições de criação e autenticação de usuários. Ao receber uma requisição para criação de usuário, a função verifica se o CPF fornecido é único e válido, antes de encaminhar os dados para o Cognito. Em seguida, o Cognito cria o usuário e o associa ao seu CPF como username, garantindo unicidade e simplificando futuras autenticações. Para autenticação, a função Lambda recebe as credenciais do usuário (CPF), valida-as e as encaminha para o Cognito.

Se as credenciais existirem, o Cognito retorna um token de autenticação e identificação, permitindo que o usuário acesse a API de forma segura.

- API Gateway

Em nossa arquitetura, utilizamos o API Gateway para facilitar a comunicação bidirecional entre uma função Lambda na AWS e um cluster EKS (Amazon Elastic Kubernetes Service), oferecendo assim uma solução completa e integrada para nossas necessidades de computação.

O API Gateway atua como um ponto de entrada unificado para nossos serviços, aceitando requisições HTTP de clientes e direcionando-as de forma inteligente para os destinos apropriados. Quando uma requisição é recebida para uma determinada rota, o API Gateway pode encaminhá-la para a função Lambda ou para os pods do EKS, que executa a lógica de negócio associada.

-----------------------------------------------------------------------------------------------------------------

3 - Demonstração dos repositórios e CI/CD

Utilizamos o GitHub Actions para o gerenciamento de pipelines de CI/CD. A flexibilidade e os recursos abrangentes do GitHub Actions nos permitem personalizar os pipelines de acordo com as necessidades específicas de cada projeto, como criar recursos de infra via terraform, realizar build da aplicação, análise de código com sonarcloud e deploy da aplicação. Agora vamos demonstrar como estruturamos nossas pipelines e suas responsabilidades:

Explicar as pipes e steps abaixo

3.1 - Deploy dos terraform

3.1 - Deploy da aplicação

3.2 - Proteção de branch main/master

3.2 - Demonstração SonarQube

-----------------------------------------------------------------------------------------------------------------

4 - Demonstração do fluxo completo

4.1 - Criação de usuário

4.2 - Autenticação de usuário

4.2.1- Usuário não identificado

4.2.2- Usuário identificado

4.3 - Fluxo de pediddo com techlanches API (API Gateway/ direto no EKS LB)

4.4 - Interação com a API sem token (mostrar erro de autenticação)

-----------------------------------------------------------------------------------------------------------------

5 - Explicação da estrutura de BD + diagrama

Outro requisito dessa fase foi avaliar a nossa estrutura de banco de dados e utilizar um banco de dados na nuvem.

Optamos pelo SQL Server por conta da familiaridade do time com a tecnologia e também pelas características abaixo:

Desempenho e Escalabilidade: O SQL Server é conhecido por seu alto desempenho e capacidade de escalar vertical e horizontalmente.

Isso permite que as empresas cresçam suas bases de dados de acordo com a necessidade, mantendo a eficiência.

Segurança: Oferece recursos robustos de segurança, incluindo criptografia de dados, autenticação integrada e gerenciamento

de permissões, garantindo a proteção dos dados sensíveis.

Compatibilidade com Aplicações: Tem ampla aceitação e integração com várias aplicações e ferramentas de desenvolvimento, facilitando a implementação em ambientes corporativos.

Amazon RDS

Optamos pelo Amazon RDS por conta do custo benefício e também pelas características abaixo

Gerenciamento Simplificado: O Amazon RDS simplifica tarefas de administração de banco de dados, como provisionamento de hardware, configuração de banco de dados, patching de software, e backups, permitindo que os desenvolvedores se concentrem no desenvolvimento de aplicações.

Alta Disponibilidade e Durabilidade: Oferece configurações de replicação fáceis de configurar, backups automatizados, garantindo a disponibilidade e a durabilidade dos dados.

Escalabilidade: Permite ajustar facilmente os recursos computacionais ou a capacidade de armazenamento com apenas alguns cliques ou de forma automática, proporcionando flexibilidade conforme a demanda varia.

Compatibilidade: Suporta vários motores de banco de dados, incluindo MySQL, PostgreSQL, Oracle, SQL Server, e MariaDB, oferecendo flexibilidade para trabalhar com a tecnologia preferida ou exigida pelo projeto.

Mostrar diagrama de dados e explicar as tabelas

-----------------------------------------------------------------------------------------------------------------

6 - Encerramento (5-10s)

Obrigado pela atenção e esperamos que tenham gostado na apresentação, visite o nosso repositório,

deixe uma estrela e em caso de dúvidas, todos estaremos à disposição para te responder.