DOCUMENTAÇÃO – SEGUNDA ENTREGA CLOUD COMPUTING E DEVOPS TOOLS

Tipo de cloud, recursos utilizados e explicação da arquitetura de solução

Resumo

TRABALHO FEITO PARA A SEGUNDA ENTREGA DO CHALLENGE DA DISCIPLINA DE CLOUD

COMPUTING E DEVOPS TOOLS, REFERENTE AO CHALLENGE DA FIAP EM PARCERIA COM A

PLUSOFT.

**Grupo Poagilers**

Componentes do grupo –

Henrique Neves Lago – RM84549

Larissa Alves – RM86351,

Giovanna de Mello Leiva – RM85817,

Daniel Sanchez Melero – RM85109,

Kaue Augusto - RM85707,

Eric Pessoa – RM86405.

**O melhor tipo de nuvem que se encaixaria na nossa solução**

Por conta da natureza do nosso serviço, que é um aplicativo de cardápio que propõe uma maior e melhor experiência gastronômica, estaremos presentes ao longo do Brasil inteiro, não importando de onde o restaurante está geograficamente localizado, se ele quiser se inscrever e começar a utilizar nosso aplicativo concomitantemente com o atendimento padrão do estabelecimento, ele não terá problema algum com isso. Tendo isto em mente, e já pensando na escala que o aplicativo pode vir a tomar, decidimos, por diversos motivos, utilizar 100% do tipo de nuvem pública, e os recursos nela disponíveis, por meio da Azure e IBM cloud.

O tipo de nuvem pública nos proporcionará elasticidade, segurança, disponibilidade, tempo de colocação competitivo no mercado e queda de custos de capex (despesas ou investimentos em bens de capital). Vamos falar sobre cada um desses pontos:

**Elasticidade** – Por não termos controle da magnitude de adesão que o público possa ter do nosso aplicativo, precisamos de uma solução que consiga se adaptar rapidamente à nossa demanda de poder computacional, e a nuvem pública nos dá justamente essa vantagem. Conseguiríamos começar de uma forma modesta, e, se for um sucesso, ou se acontecer algo que não previmos, como pico de acesso e uso em algum dia aleatório, poderemos nos adaptar de forma rápida e eficaz, não perdendo tempo fora do ar, e mantendo nosso serviço ativo e funcional para todos que quiserem acessá-lo. Além disso, poderemos pagar apenas o que estamos utilizando e precisando de poder computacional em qualquer momento, coisa que, se comprássemos um servidor privado, seria impossível.

Uma imagem contendo Diagrama

Descrição gerada automaticamente

**Segurança e disponibilidade** – Tendo gigantes como a Azure e IBM cloud nos proporcionando o que vamos precisar, de Iaas a Saas, podemos ter certeza que nosso serviço estará seguro, com uma disponibilidade altíssima, e até com segurança contra desastres naturais, se assim quisermos.

Interface gráfica do usuário, Ícone

Descrição gerada automaticamente

**Tempo de colocação competitivo no mercado** - Com um meio de configuração fácil, e atendimento praticamente imediato de qualquer demanda que quisermos, podemos garantir que nossos processos de implementação do serviço no mercado e administração de características do nosso aplicativo serão velozes.

Logotipo

Descrição gerada automaticamente

**Troca de custos capex por opex** - Colocando toda nossa infraestrutura como responsabilidade da azure, utilizando a Iaas (Infraestrutura como serviço), vamos ter uma queda de custos tremenda, pois não precisaremos nos preocupar com diversos fatores, como manutenção de servidores físicos, alocação de um espaço físico adequado para esses servidores se situarem, mão de obra qualificada para manutenir o bom funcionamento deles, medidas de segurança adicionais que devem ser tomadas, pagar a energia para os servidores... Tudo isso ficaria na conta da Azure, e esses “custos fantasmas” devem ser considerados quando pensamos se a nuvem será uma boa escolha, e não termos que nos preocupar com isso com certeza é uma vantagem. Outro benefício é que não precisaríamos desembolsar uma quantia muito grande de capital inicial para podermos começar e colocar nosso serviço online. Veja neste exemplo gráfico:

Gráfico, Histograma

Descrição gerada automaticamente

Como fica evidente na linha vermelha, é muito comum gastarmos bem mais do que de fato precisamos quando estamos lidando com capex, justamente para se ter um “espaço de folga” nesse processo que é demorado e engessado. Também é possível ver que, com a escalabilidade facilitada que a nuvem proporciona, gastamos apenas com o que de fato vamos precisar em qualquer época.

**Os possíveis recursos de nuvem que serão utilizados**

**Web apps –** Pensamos que para nossa aplicação ter uma implementação rápida na nuvem e atualizações constantes por parte da nossa equipe, uma das melhores opções seria utilizar o PaaS, e, com isso, não precisaremos nos preocupar nem com a parte de infraestrutura nem com a parte de configuração e administração da mesma, fazendo com que possamos focar na parte mais importante, que é a codificação do aplicativo em si.

Ícone

Descrição gerada automaticamente

**Banco de dados Oracle SQL –** Os bancos de dados oferecidos pelos serviços computacionais em nuvem são uma alternativa simples, de baixo nível administrativo (comparado a um servidor físico que hospeda um banco de dados), pois, além termos uma altíssima disponibilidade e escalabilidade, poderemos contratar apenas o que estamos precisando no momento, nos dando espaço para focar em questões mais importantes para um negócio em ascensão.

Diagrama

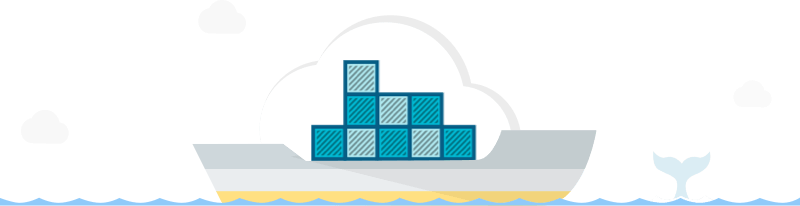
Descrição gerada automaticamente

**Azure functions –** Esse recurso nos disponibilizará uma plataforma para criarmos códigos que ficam armazenados e rodando na nuvem, prontos para serem executados a partir de gatilhos que definirmos. Esses códigos, quando disparados, acionarão uma determinada lógica, podendo mudar uma ampla gama de aspectos da nossa aplicação. Isso será de grande ajuda para a administração tanto da infraestrutura da nossa solução quanto do nosso web app, deixando nossa solução mais otimizada, escalada e planejada. Um outro benefício é que, quando já temos um web app, a azure functions consegue rodar dentro do mesmo plano sem custo adicional. Um resumo em imagem para uma explicação fácil seria esse:

Uma imagem contendo Word

Descrição gerada automaticamente

**Container –** Para podermos utilizar o react native em nosso web app (linguagem na qual vamos desenvolver o front-end das telas mobile), iremos utilizar do serviço de conteinerização que a azure oferece. Este serviço nos ajuda a ter softwares contidos, com todas as bibliotecas, dependências, e coisas necessárias a serem instaladas para ele funcionar, dentro de uma unidade padronizada que consiga ser lida em qualquer lugar que acharmos conveniente. Isso nos ajuda a economizar tempo não precisando fazer essa configuração que já viria pronta, e nos permitiria trabalhar com a linguagem de nossa escolha em nossa aplicação web.



**Desenho e explicação da nossa arquitetura de solução**

**Diagrama

Descrição gerada automaticamente**

Nosso projeto se constitui de uma arquitetura em 2 camadas: “Cloud e aplicações” e “parte física”. A primeira é constituída por todas os recursos que estamos utilizando em nuvem, além dos APIs e serviços fora dela. A segunda diz respeito a placa arduino e componentes que serão utilizados para execução da parte IOT do nosso sistema. A requisição pode tomar 2 caminhos, dependendo do que está sendo solicitado.

Se estiver sendo solicitada alguma informação que necessite de informações do nosso banco de dados (também presente na nuvem), a requisição irá tomar o caminho que utiliza as operações CRUD com o backend JAVA para resgatar as informações do banco de dados pertinentes a cada estabelecimento.

Se estiver sendo solicitada alguma coisa a ser realizada no espaço físico (iluminação ou eventos relacionados à música), a solicitação irá passar pelo node-red (controlador de fluxos e orquestrador), passar pelo broker (mosquito), que pegará qual grupo realizou a solicitação e saberá qual mesa iluminar ou qual caixa de som tocar a partir disso. Pode ser feito o mesmo caminho de volta para mandar alertas para o front informando se a solicitação foi bem-sucedida ou não.

Temos APIs integradas, do Telegram, em que pode ser enviado uma mensagem por voz ou por chat para acionar os recursos IOT do nosso aplicativo, e o Spotify, para as pessoas poderem realizar funções relacionadas às músicas que estão tocando, quais serão tocadas, sugestões e busca de músicas.

Os dispositivos de hardware utilizados serão a placa micro controladora ESP WROOM-32 e o NeoPixel Ring de 24 leds (este para implementar a funcionalidade de configuração da luz ambiente).