**Plataforma AWS**

Escolhemos a plataforma AWS considerando as constantes atualizações de segurança e também pelas aplicações que podem ser integradas.

Nossa aplicação utiliza os conceitos de CD/CI, Deploy automático e Monitoramento

Aplicativos utilizados da plataforma AWS:

* **AWS CodeBuild** - Serviço de integração contínua totalmente gerenciado que compila código-fonte, executa testes e produz pacotes de software prontos para serem implantados, em um servidor de compilação criado dinamicamente.
* **AWS Codedeploy** - Serviço de implantação totalmente gerenciado que automatiza implantações de software. Permite que as implantações de rolamento e reversão automática, ou seja, quando ocorre o erro na implantação é feito um rollback. Os erros podem ser acompanhados através dos eventos que aparecem.
* **AWS CodePipeline** - Serviço de [entrega contínua](https://aws.amazon.com/devops/continuous-delivery/) totalmente gerenciado que ajuda você a automatizar seus pipelines de lançamento para atualizações rápidas e confiáveis de aplicativos e infraestrutura. A cada push de “commit” realizado, a aplicação transfere o processo para o próximo passo automaticamente.
* **AWS Cloudwatch** - Uma regra de eventos do AWS CloudWatch é criada para acionar o CodePipeline em um compromisso do Git com o repositório CodeCommit.

**Etapas de construção**

Passo 1: Crias as funções do IAM (para cada papel – HML/PROD)

* Criar função para a instancia EC2 com permissões para codeDeploy e acesso S3 para baixar o código
* Criar uma função para o serviço CodeDeploy com permissões para acessar as funcionalidades S3 e CodeBuild básicas

Na plataforma AWS acesse Identity and Acess (IAM), opção Acess Management > Roles. Clicar em “Create Role” , selecionar “EC2” e clicar em “Next” , filtrar as politicas S3, selecionar “AmazonS3FullAccess”, após filtrar as politicas EC2 e selecionar “AmazonEC2RoleforAWSCodeDeploy”, clicar em “Next Tags”, na parte “Role Name” selecionar “ServiceRole-EC2” e após clicar em “Create Role”.

Após isso, clicar novamente em “Create Role”, nesta parte, selecionar “CodeDeploy”, após isso, selecionar novamente “CodeDeploy” e clicar em “Next Permissions” , esperar carregar e clicar em “Next tags”, na opção “Role name” selecionar “ServiceRole-CodeDeploy” e clicar em “Create Role”. Após isso, clicar na barra verde que será demonstrada na parte superior da tela, no link “ServiceRole-CodeDeploy”, esperar carregar a informação na parte “Role description” e após isso, clicar em “Anexar politicas”, filtrar por “S3” e selecionar “AmazonS3FullAccess” e clicar em “Attach policy”.

Passo 2: Criar um repositório S3 para armazenar o código fonte e construir artefatos

Na plataforma AWS, entrar na S3, opção “Create Bucket”, escolher o nome (ex. ng-codedeploy-repositorio) e clicar em “Create bucket”

Passo 3: Criar um projeto de construção do Codigo

Aqui criaremos um projeto de compliação que tomara código-fonte do repositório do GitHub e autenticara o AWS CodeBuild para usar o código de nossa conta Github

Na plataforma AWS acessar CodeBuild, na opção “Build” > Build projects, clicar em “Create build Project”, escolher o nome do projeto (Ex. demo-build), na parte “Source”, selecionar “Github”, e após, colocar em “Repositório em minha conta Github”, escolher o repositório. Após isso, rolar a página até a parte de “Environment” e selecionar o sistema operacional utilizado e na parte “Runtime(s)”, selecionar “Standard”, na parte “Image”, selecionar a image criada para o contêiner. Na parte “Role name”, selecionar os tipos de serviços utilizados (ex. codebuild-demo-build-service-role), após rolar a página até o final, na parte “Logs” selecionar “CloudWatch” e clicar em “Create build Project”.

Após isso, na plataforma AWS acessar o IAM, na parte “Role”, ir na “Revoke session” e selecionar o projeto criado no passo anterior, esperar carregar a pagina e cliclar em “Attach policies” filtrar por “S3” e selecionar “AmazonS3FullAccess”, cliclar em “Attach policy”.

Passo 4: Criar o aplicativo de Deploy

Este aplicativo baseia-se no arquivo “appspec.yaml” que serve para mostrar “da onde eu quero copiar” e “para onde eu quero levar”.

Na plataforma AWS, na busca, procurar por “codedeploy” , após isso, clicar em “criar um aplicativo”, escolher um nome e após isso, escolher a plataforma (clicar em EC2 que são as instâncias). Dentro deste aplicativo, criar um grupo de implantação (que são os servidores), escolher um nome e escolher a função “codedeploy”.

Passo 5: Criação do grupo de Implantação

Na parte de “configuração de ambiente”, selecionar a opção “Grupos de Auto Scaling do Amazon EC2” e clicar em “Criar Grupo de Implantação”.

Feito esses processos, integraremos o Github ao deploy e para isso, utilizaremos o Pipeline, ferramenta responsável pela automação do processo de deploy quando alterado o código da aplicação.

Passo 6: Criação Pipeline

Na plataforma AWS, clicar em Pipeline (CodePipeline), clicar em “Criar novo Pipeline”, escolher um nome, clicar em próximo e escolher a origem, neste caso, Github. Ao selecionar esta opção, será necessário digitar seu usuário e senha (caso ainda não possua conta, será necessário criar através do site: [www.github.com](http://www.github.com)). Após se logar com o Github, escolher o repositório que será utilizado, escolher a ramificação (master ou outra), selecionar a opção “Webhooks do Github” (o Github comunica quando houver alteração), clicar em próximo, a parte de compilação, selecionar “AWS CodeBuild”, na parte “Project name”, selecionar o projeto criado anteriormente e clicar em “Next”, após isso, escolher o provedor de implantação, neste caso, “AWS CodeDeploy” , selecionar o nome do aplicativo criado anteriormente e também o nome do grupo de implantação, selecionar "próximo" e clicar em “Criar Pipeline”.

Feito isso, a aplicação já estará rodando e qualquer alteração que seja realizada no código, após os “push” de commit, as alterações serão automáticas na aplicação.