Tworzymy zasoby do CICD infrastruktury

Zasoby CICD do testowania, budowania i wdrożenia infrastruktury pod aplikację:

1. Repozytorium z kodem infrastruktury pod aplikację
2. Repozytorium z kodem zasobów CICD
3. Parametry SSM z ID kont AWS, na których znajdują się poszczególne środowiska
4. Klucz KMS do szyfrowania artefaktów
5. Zestaw ról do wdrażania infrastruktury pod aplikację
6. Parametry SSM z nazwami ról
7. Dokument SSM z kodem tworzenia użytkownika w bazie danych
8. Pozostałe zasoby CICD
   - Bucket S3 na artefakty
   - Role dla AWS CodeBuild i AWS CodePipeline
9. Projekty AWS CodeBuild:
   - Testowanie szablonów i zbudowanie artefaktów
   - Utworzenie użytkownika w bazie danych
   - Przesłanie plików do bucketów S3
   - Specyfikacje buildspec.yaml dla każdego z projektów
   - Skrypty bashowe
10. Pipeliny AWS CodePipeline
    - Pipeline do tworzenia zasobów

Dodatkowo:

11. Skrypt klonujący parametry środowiska (pod dodatkowe środowisko developerskie)
12. Pipeline usuwający część zasobów
13. AWS EventBridge rule + rola - do automatycznego uruchamiania pipeline po aktualizacji kodu na określonym branchu w określonym repozytorium kodu infrastruktury
14. AWS EventBridge rule + rola - do uruchamiania pipelinów o określonej porze

Uwaga

1. Zasoby należy tworzyć w określonej kolejności
2. Konta AWS z poszczególnymi środowiskami mogą być różne, jednak w tym projekcie używamy jednego konta dla wszystkich środowisk

Repozytoria kodu

Instrukcja dotycząca repozytoriów znajduje się w odrębnym dokumencie. Nie rozpoczynaj kolejnych kroków bez utworzenia i konfiguracji repozytoriów.

Parametry SSM z ID kont AWS

W tych parametrach ustawiamy ID kont dla poszczególnych środowisk (w tym projecie będzie to wszędzie to samo konto)

Pliki:

```
memes-generator/cicd/templates/account-parameters.yaml
memes-generator/cicd/parameters/account-parameters-shared-dev.json
memes-generator/cicd/commands/deploy-account-params.sh
```

Część kodu jest zakomentowana (jest to zamierzone). Plik z parametrami należy uzupełnić odpowiednim numerem konta

Klucz KMS do szyfrowania artefaktów

Artefakty szyfrujemy kluczem - albo zarządzanym przez AWS, albo przez nas. Tu utworzymy własny klucz. Własny klucz jest niezbędny, gdy chcemy tworzyć zasoby na innym koncie, niż konto, gdzie znajduje się pipeline.

Pliki:

```
memes-generator/cicd/templates/kms.yaml
memes-generator/cicd/parameters/kms-shared-dev.json
memes-generator/cicd/commands/deploy-kms.sh
```

Role

Do wdrażania zasobów będzie potrzebnych kilka ról.

Pliki:

```
memes-generator/cicd/templates/cicd-roles.yaml
memes-generator/cicd/parameters/cicd-roles-dev.json
memes-generator/cicd/commands/deploy-cicd-roles.sh
```

Zwróć uwagę, że w przypadku ról oznaczenie `Stage` to `dev`, `test` lub `prod`. Role są tworzone na środowiskach aplikacyjnych. Plik z parametrami wymaga uzupełnienia o ARN klucza KMS i ID konta z zasobami do CICD.

Parametry SSM z nazwami ról

Nazwy ról musimy umieścić w parametrach SSM.

Pliki:

```
memes-generator/cicd/templates/cicd-roles-parameters.yaml
memes-generator/cicd/parameters/cicd-roles-parameters-shared-dev.json
memes-generator/cicd/commands/deploy-cicd-roles-parameters.sh
```

Plik z parametrami należy uzupełnić o nazwy utworzonych ról

Dokument SSM z kodem tworzącym użytkownika w bazie danych

Dokument ten zawiera polecenia, które zostaną wykonane na bazie danych z jumphosta - ale tym razem bez naszego udziału.

Pliki:

```
memes-generator/cicd/templates/ssm-command-create-db-user.yaml
memes-generator/cicd/parameters/ssm-command-create-db-user-shared-dev.json
memes-generator/cicd/commands/deploy-ssm-command-create-db-user.sh
```

Dokument zostanie użyty w poleceniu `aws ssm send-command` w skrypcie (a skrypt zostanie uruchomiony za pomocą usługi AWS CodeBuild)

Projekty AWS Codebuild, specyfikacje buildspec i skrypty

Projekty AWS CodeBuild służą do testowania i budowania kodu, ale można ich również użyć do automatyzacji w ramach pipeline.

Wszystkie niezbędne specyfikacje (pliki buildspec do danego projektu) oraz skrypty są umieszczane w repozytorium z kodem infrastruktury pod aplikację w katalogach:

```
memes-generator/cicd/buildspec
memes-generator/cicd/scripts
```

Pliki te są następnie wykorzystywane przez AWS CodeBuild.

Tworzymy 3 projekty (z tego samego szablonu):

Testowanie szablonów i budowanie artefaktów

W ramach tego projektu szablony AWS CloudFormation są walidowane i testowane. Pozwala to uniknąć części błędów podczas wdrożeń. Następnie - z plików z parametrami - budowane są pliki konfiguracyjne zawierające parametry oraz tagi do stacków tworzonych przez AWS CodePipeline.

Pliki:

```
memes-generator/cicd/templates/build-project.yaml
memes-generator/cicd/parameters/test-and-create-config-project-shared-
dev.json
memes-generator/cicd/commands/deploy-test-config-project.sh
```

Tworzenie użytkownika w bazie danych

Tworzenie użytkownika w bazie danych odbywa się za pomocą dokumentu SSM wykonywanego na jumphoście. Procesem tym steruje AWS CodeBuild.

Pliki:

```
memes-generator/cicd/templates/build-project.yaml
memes-generator/cicd/parameters/create-db-user-project-shared-dev.json
memes-generator/cicd/commands/deploy-create-db-user-project.sh
```

Przesyłanie plików do S3

Pliki z konfiguracją oraz obrazki pod memy również przesyłane będą automatycznie za pomocą tego projektu.

Pliki:

```
memes-generator/cicd/templates/build-project.yaml
memes-generator/cicd/parameters/upload-files-project-shared-dev.json
memes-generator/cicd/commands/deploy-upload-files-project.sh
```

Pozostałe zasoby

Instrukcje do tworzenia pipelinów oraz pozostałych zasobów znajdują się w odrębnych plikach.