Politechnika Wrocławska
Wydział Informatyki i Telekomunikacji

Kierunek: IST

ZESPOŁOWE PRZEDSIĘWZIĘCIE INFORMATYCZNE

Scan and Store
System do Zarządzania Inwentarzem z Użyciem Kodów Kreskowych i Kodów QR

Adam Klementowski
Adam Rudnicki
Adam Skowron
Wojciech Skrzypiec

Opiekun pracy: dr inż. Rafał Palak

zarządzanie inwentarzem, PWA

WROCŁAW 2024

SPIS TREŚCI

1 Wykaz symboli, oznaczeń i akronimów **3**
1.1 Symbole . . . 3
1.2 Oznaczenia . . . 3
1.3 Akronimy . . . 3

2 Cel i zakres przedsięwzięcia **4**
2.1 Cel . . . 4
2.2 Zakres . . . 4

3 Słownik pojęć **5**

4 Stan wiedzy w obszarze przedsięwzięcia **6**

5 Założenia wstępne **7**

6 Specyfikacja wymagań na produkt programowy **8**
6.1 Baza danych . . . 8

7 Projekt produktu programowego **9**
7.1 Baza danych . . . 9
7.2 Architektura . . . 9
7.3 Pozostałe schematy i diagramy . . . 9

8 Implementacja **11**
8.1 Warstwa użytkownika - frontend . . . 11
8.1.1 Struktura plików . . . 11
8.1.2 Widoki . . . 11
8.1.3 Routing . . . 11
8.1.4 Testy . . . 11
8.1.5 Konteneryzacja - Docker . . . 11
8.2 Warstwa systemowa - backend . . . 12
8.2.1 Struktura plików . . . 12
8.2.2 Konfiguracja . . . 12
8.2.3 Kontrolery . . . 12
8.2.4 Inne . . . 12
8.2.5 Testy . . . 12
8.2.6 Konteneryzacja - Docker . . . 12
8.3 Architektura chmurowa . . . 13
8.3.1 Wstęp . . . 13
8.3.2 Struktura plików . . . 13
8.3.3 Wykorzystane zasoby . . . 14

9 Demonstracja produktu programowego **15**

1 WYKAZ SYMBOLI, OZNACZEŃ I AKRONIMÓW

1.1 Symbole

1.2 Oznaczenia

1.3 Akronimy

- SNS - Scan and Store, Nazwa robocza projektu
- AWS - Amazon Web Services
- PWA - Progressive Web App

2 CEL I ZAKRES PRZEDSIĘWZIĘCIA

2.1 Cel

Celem przedsięwzięcia jest zbadanie dojrzałości technologii PWA na przykładzie systemu do zarządzania inwentarzem z użyciem kodów kreskowych i kodów QR.

2.2 Zakres

W fazie planowania projektu ustalono, iż aplikacja końcowa powinna korzystać z funkcjonalości oferowanych przez techonologię PWA oraz umożliwiać zarządzanie stanem magazynu w podstawowym zakresie.

Z funkcjonalości PWA jakie powinny zostać zaimplementowane wybrano dostęp do kamery urządzenia w celu skanowania kodów kreskowych i kodów QR oraz dostęp do geolokacji.

3 SŁOWNIK POJĘĆ

- framework - biblioteka narzędzi wspomagających implementację danego rozwiązania w danym języku

4 STAN WIEDZY W OBSZARZE PRZEDSIĘWZIĘCIA

5 ZAŁOŻENIA WSTĘPNE

W fazie wstępnej planowania projektu ustalone zostały następujące założenia:

Klient

- Aplikacja wytwarzana jest dla jednego klienta
- Klient posiada wiele magazynów
- Klient chce być w stanie sprawdzić stan każdego z magazynów
- Klient chce dać swoim pracownikom możliwość dodawania artykułów do stanu magazynu
- Klient chce mieć wyłączny dostęp do dodawania i modyfikowania artykułów, magazynów i kategorii

Aplikacja

- Aplikacja musi wspierać techonolgię PWA
- Aplikacja musi być instalowalna na różnych urządzeniach
- Aplikacja musi wspierać skanowanie kodów kreskowych i kodów QR

6 SPECYFIKACJA WYMAGAŃ NA PRODUKT PROGRAMOWY

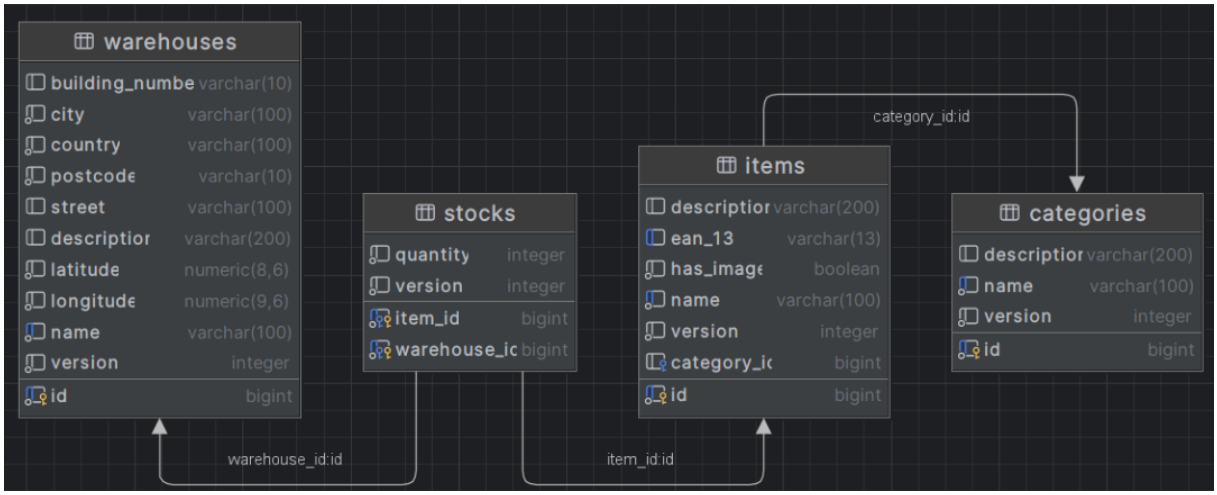
6.1 Baza danych

- Magazyn posiada nazwę i adres: państwo, miasto, ulica, kod pocztowy oraz opcjonalnie numer domu
- Magazyn może posiadać opis
- Każdy magazyn może mieć artykuły
- Artykuły posiadają nazwę i kategorię
- Artykuły mogą posiadać zdjęcie, opis i kod EAN13
- Kategorie posiadają nazwę
- Kategorie mogą posiadać opis
- Kategorie mogą być przypisane do więcej niż jednego artykułu
- Jeden artykuł może mieć nie więcej niż jedną kategorię
- Artykuł może się znajdować w wielu magazynach w różnych ilościach

7 PROJEKT PRODUKTU PROGRAMOWEGO

7.1 Baza danych

Baza danych została zaimplementowana w PostgreSQL według diagramu na Rysunku 1.



Rysunek 1: Diagram fizyczny bazy danych

7.2 Architektura

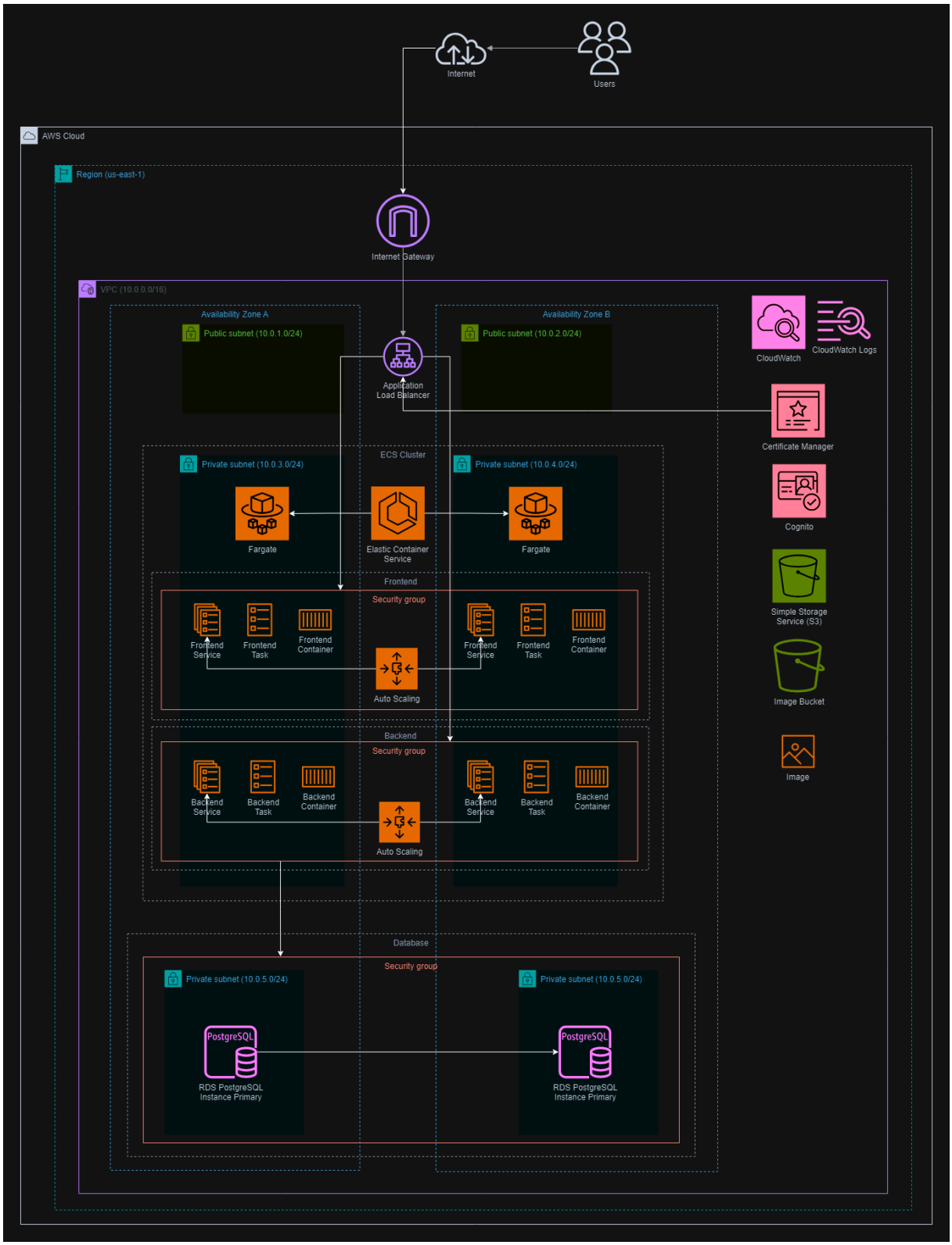
Architektura aplikacji została wdrożona przy pomocy narzędzia HashiCorp Terraform na platformie AWS. Schemat zawierający najważniejsze elementy architektury przedstawiony jest na Rysunku 2

7.3 Pozostałe schematy i diagramy

Diagram przypadków użycia

Diagram klas

Schemat przebiegu aplikacji



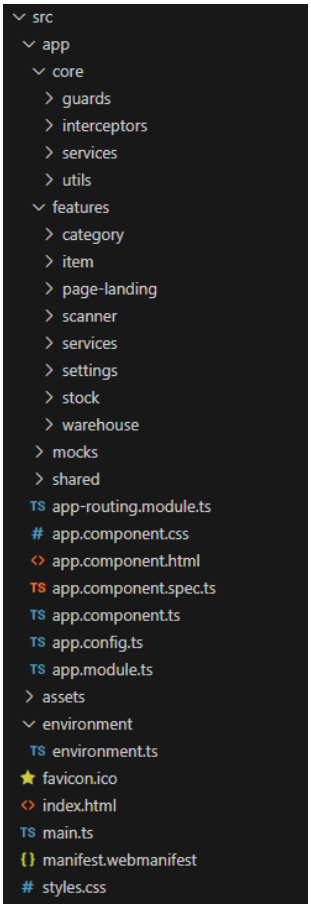
Rysunek 2: Diagram architektury AWS

8 IMPLEMENTACJA

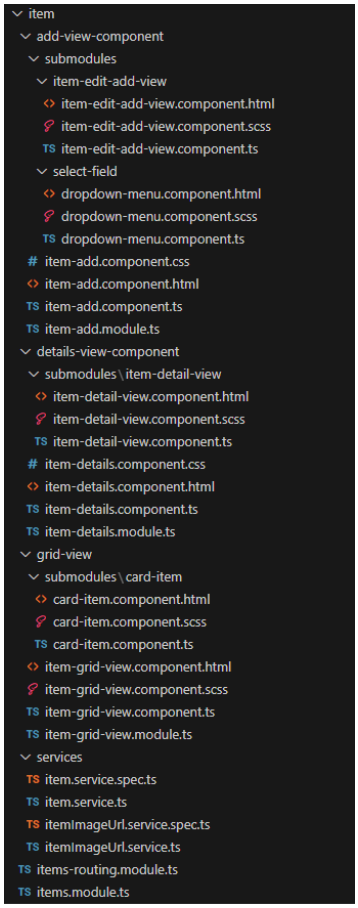
8.1 Warstwa użytkownika - frontend

8.1.1 Struktura plików

Warstwa użytkownika została zaimplementowana przy pomocy frameworku Angular [2] opartego o język TypeScript (TS). Strukturę plików przedstawia Rysunek 3.



(a) Ogólna struktura plików



(b) Dokładna struktura jednego z komponentów

Rysunek 3: Struktura plików frontendu

Aplikacja jest tworzona przy pomocy komponentów. Każdy komponent składa się ze skryptu TS, pliku stylu SCSS oraz pliku układu HTML. Dodatkowo niektóre komponenty korzystają z serwisów napisanych w języku TypeScript. Komponenty graficzne zazwyczaj mogą być wykorzystywane wiele razy i są one umieszczone w folderze *shared*. Komponenty jednokrotne są umieszczone w folderze features i zawierają one kokretne implementacje poszczególnych widoków.

8.1.2 Widoki

8.1.3 Routing

8.1.4 Testy

8.1.5 Konteneryzacja - Docker

8.2 Warstwa systemowa - backend

8.2.1 Struktura plików

8.2.2 Konfiguracja

8.2.3 Kontrolery

8.2.4 Inne

8.2.5 Testy

8.2.6 Konteneryzacja - Docker

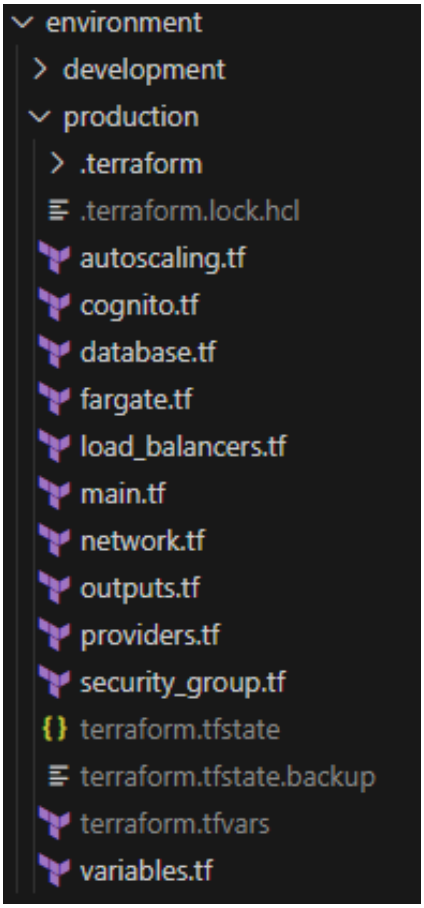
8.3 Architektura chmurowa

8.3.1 Wstęp

Architektura chmurowa została napisana w oprogramowaniu HashiCorp Terraform i wdrożona na serwery Amazon Web Services (AWS) [1]. HashiCorp Terraform (albo krótko Terraform) [3] to oprogramowanie Infrastructure as Code (IaC) pozwalające na skryptowe wdrażanie infarstruktury na serwery dostawcy.

8.3.2 Struktura plików

Struktura plików została podzielona ze względu na środowisko produkcyjne i testowe. Na Rysunku 4 przedstawione jest tylko środowisko produkcyjne ze względu na tą samą strukturę systemu plików. Z tego samego względu omówione zostanie tylko środowisko produkcyjne.



Rysunek 4: Struktura plików Terraform

Pliki .tf zawierają opis zasobów wdrażanych na serwery. Plik .tfstate przechowuje dokładny spis zasobów już wdrożonych wraz z ich unikalnymi identyfikatorami, sekretami, itd. Opis pozostałych plików:

- main.tf - wersje oprogramowania wykorzystywane w projekcie
- providers.tf - konfiguracja dostawcy AWS
- variables.tf - spis wszystkich zmiennych wraz z ich opisami i niektórymi wartościami domyślnymi
- terraform.tfvars - niewersjonowany plik zawierający wartości niektórych zmiennych; przede wszystkim przechowuje sekrety
- outputs.tf - zbiór przydatnych wartości wyjściowych, np. nazwa DNS Application Load Balancera
- network.tf - zasoby sieciowe: VPC, brama sieciowa, tablice routingu, podsieci
- security_groups.tf - grupy bezpieczeństwa (działają jak zapora ogniowa) konfigurujące przychodzący i wychodzący ruch sieciowy
- cognito.tf - dostawca tożsamości Amazon Cognito i jego konfiguracja

- load_balancers.tf - Application Load Balancer, zasoby nasłuchujące na portach 80, 443 i 8080 oraz grupy celów
- fargate.tf - definicje zadań wykonywanych przy pomocy AWS Fargate oraz serwisy które nimi zarządzają
- autoscaling.tf - zestaw reguł pozwalających na automatyczne zmniejszanie lub zwiększanie liczby działających instancji modułów (frontendu i backendu)
- database.tf - konfiguracja bazy danych RDS

8.3.3 Wykorzystane zasoby

- **Amazon Virtual Private Cloud (VPC)**
- **Amazon Cognito**
- **Amazon Relational Database Service (RDS)**
- **Amazon Application Load Balancer (ALB)**
- **Amazon Elastic Container Service (ECS)**
- **Amazon Certificate Manager (ACM)**
- **Amazon CloudWatch**
- **Amazon Simple Storage Service (S3)**
- **AWS Auto Scaling**

9 DEMONSTRACJA PRODUKTU PROGRAMOWEGO

LITERATURA

[1] Matt Garman (CEO). Aws dokumentacja. `https://docs.aws.amazon.com`. Dostęp: 2024-11-30.

[2] Google. Angular dokumentacja. `https://angular.dev`. Dostęp: 2024-11-30.

[3] HashiCorp. Hashicorp terraform dokumentacja. `https://developer.hashicorp.com/terraform/docs`. Dostęp: 2024-11-30.