

Bazy danych – Northwind

System do składania zamówień

Autorzy

- Kamil Gliński
- Mateusz Popielarz
- Michał Flak

Spis treści

- [Bazy danych – Northwind](#)
- [System do składania zamówień](#)
- [Spis treści](#)
- [Wstęp](#)
 - [Przebieg prac:](#)
 - [Adres do repozytorium:](#)
 - [Użyte technologie:](#)
 - [Uruchomienie dla developera:](#)
 - [Uruchomienie:](#)
 - [Odnosińki w aplikacji](#)
- [Dokumentacja funkcjonalna](#)
 - [Interfejs użytkownika](#)
 - [Konfiguracja](#)
 - [Dodać paczki](#)
 - [Skonfigurować middleware](#)

Wstęp

Przebieg prac:

Do synchronizowania efektów pracy używamy oprogramowania GIT i serwisu GitHub

Adres do repozytorium:

<https://github.com/pixellos/agh.6.bd>

Użyte technologie:

- PostgreSQL,
- Hibernate,
- Java,
- Spring boot
- Swagger
- SwaggerUI

Uruchomienie dla developera:

W celu uruchomienia aplikacji należy:

- Sklonować repozytorium,
- Zainstalować na lokalnym komputerze bazę danych PostgreSQL
- Wykonać na bazie danych skrypty które znajdują się w repozytorium w lokalizacji `/resources/db-schema`
- uruchomić aplikację backendową przez klasę `NorthwindApplication.java`

Uruchomienie:

W celu uruchomienia aplikacji należy zainstalować:

- Docker for windows
- WSL2

Wykonujemy `initialize.ps1` i aplikacja działa na

<http://localhost:5000>

jest też hostowana

<https://northwind-java-pixellos.cloud.okteto.net/swagger-ui/>

Odnośniki w aplikacji

- pobranie produktów po kategorii

<http://localhost:8080/products/category/Beverages>

- pobranie produktów po kraju zapewniającego

<http://localhost:8080/products/supplier/country/USA>

<http://localhost:8080/products/supplier/country/Japan>

- pobranie produktów po zapewniającym

<http://localhost:8080/products/supplierId/1>

- pobranie produktów po id klienta

<http://localhost:8080/orders/customer/SUPRD>

- poranie zamówień po id klienta

<http://localhost:8080/orders/customer/VINET>

- pobranie zamówień po id klienta

<http://localhost:8080/orders/employee/2>

- pobranie pracowników po id

<http://localhost:8080/employees/2>

- pobranie detale zamówień po id zamówienia

<http://localhost:8080/orderDetails/order/10248>

- pobranie detali zamówień po id produktu

<http://localhost:8080/orderDetails/product/11>

- pobranie detali zamówień w kategorii produktów

<http://localhost:8080/orderDetails/product/category/Beverages>

- pobranie detali zamówień po id zapewniającego produkt

<http://localhost:8080/orderDetails/product/supplier/1>

- pobranie zmaowien po id spedytora

<http://localhost:8080/orders/shipper/1>

Dokumentacja funkcjonalna

Interfejs użytkownika

Podstawowym interfejsem użytkownika jest Swagger UI, który pozwala na łatwy dostęp do endpointów aplikacji z poziomu przeglądarki

Rysunek X. Swagger UI W aplikacji

Konfiguracja

Aby go skonfigurować trzeba:

Dodać paczki

Rysunek X. Zrzut ekranu z paczkami

Skonfigurować middleware

```
package com.agh;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.context.annotation.Import;
import org.springframework.web.servlet.view.InternalResourceViewResolver;
import springfox.documentation.spring.data.rest.configuration.SpringDataRestConfiguration;
import springfox.documentation.swagger2.annotations.EnableSwagger2;

@SpringBootApplication
@EnableSwagger2
@Configuration
@Import(SpringDataRestConfiguration.class)
public class NorthwindApplication {

    @Bean
    public InternalResourceViewResolver defaultViewResolver() {
        return new InternalResourceViewResolver();
    }

    public static void main(String[] args) {
        SpringApplication._run_(NorthwindApplication.class, args);
    }
}
```

Rysunek X. Konfiguracja middleware

Trzeba zwrócić uwagę na linię

```
@Bean
public InternalResourceViewResolver defaultViewResolver() {
    return new InternalResourceViewResolver();
}
```

Rysunek 4. Konfiguracja ViewResolvera

W obecnej wersji w swaggerUI występuje błąd, przez który ViewResolver działa niepoprawnie z najnowszym springiem. Rozwiązaniem jest ustawienie defaultViewResolvera na właściwy typ.