

# Dokumentacja Projektu Zarządzania Magazynem

Autor Szymon Janiak

## Spis Treści

1. Wprowadzenie
2. Struktura Projektu
3. Główne Klasy i Interfejsy
  - Main
  - DatabaseConnector
  - ProductDao
  - Product
  - Category
  - CloneableProduct
  - ProductQuantityComparator
4. Baza Danych
5. Wzorzec Prototypu
6. Użyte Technologie
7. Jak Uruchomić Projekt

## Wprowadzenie

Projekt zarządzania magazynem ma na celu umożliwienie użytkownikom efektywnego zarządzania produktami w magazynie. Aplikacja umożliwia dodawanie, sortowanie oraz klonowanie produktów. Wykorzystuje wzorzec projektowy Prototypu do klonowania obiektów.

## Struktura Projektu

```
src/
├─ main/
│   └─ Main.java
├─ dao/
│   └─ DatabaseConnector.java
│   └─ ProductDao.java
├─ exceptions/
│   └─ ProductNotFoundException.java
├─ model/
│   └─ Product.java
│   └─ Category.java
├─ prototype/
│   └─ CloneableProduct.java
├─ utils/
│   └─ ProductQuantityComparator.java
```

## Główne Klasy i Interfejsy

**Main**- Klasa główna, uruchamiająca aplikację. Inicjalizuje połączenie z bazą danych, dodaje produkty, sortuje je według ilości i klonuje wybrany produkt.

```
public class Main { public static void main(String[] args) { // Logika uruchamiania aplikacji } }
```

**DatabaseConnector**- Klasa odpowiedzialna za zarządzanie połączeniem z bazą danych MySQL. Oferuje metodę getConnection() do uzyskania połączenia.

```
public class DatabaseConnector { // Metoda do uzyskania połączenia z bazą danych }
```

**ProductDao** - Klasa DAO (Data Access Object), która zarządza operacjami na produktach w bazie danych. Używa HashMap do przechowywania produktów w pamięci oraz zapewnia metody do dodawania, usuwania i pobierania produktów.

```
public class ProductDao { // Metody do zarządzania produktami w bazie danych }
```

**Product**- Klasa reprezentująca produkt w magazynie. Zawiera właściwości

takie jak name, price, quantity i category. Implementuje interfejs CloneableProduct, co pozwala na klonowanie obiektów.

```
public class Product implements CloneableProduct { // Właściwości i metody klasy Product }
```

**Category Enum**- reprezentujący kategorie produktów, takie jak ELECTRONICS, GROCERIES, TOYS i OTHERS.

```
public enum Category { ELECTRONICS, GROCERIES, TOYS, OTHERS }
```

**CloneableProduct**- Interfejs definiujący metodę clone(), która pozwala na klonowanie obiektów.

```
public interface CloneableProduct { Product clone(); // Definicja metody klonującej }
```

**ProductQuantityComparator**- Klasa komparatora, która implementuje interfejs Comparator i umożliwia sortowanie produktów według ich ilości w magazynie.

```
public class ProductQuantityComparator implements Comparator<Product> { // Implementacja metody compare }
```

## Baza Danych

Projektu wymaga utworzenia bazy danych o nazwie warehouse w MySQL. Aplikacja sprawdza, czy baza danych istnieje, a jeśli nie, tworzy ją automatycznie. W bazie danych znajduje się tabela products z kolumnami odpowiadającymi właściwościom produktów.

## Wzorzec Prototypu

W projekcie zastosowano wzorzec projektowy Prototypu, co pozwala na klonowanie obiektów bez potrzeby ich tworzenia od podstaw. Klasa Product implementuje interfejs CloneableProduct, co umożliwia łatwe tworzenie kopii obiektów.

## Użyte Technologie

- Java - Język programowania.
- MySQL - System zarządzania bazą danych.
- JDBC - API do łączenia się z bazą danych.

## Jak Uruchomić Projekt

1. Upewnij się, że masz zainstalowaną bazę danych MySQL.
2. Utwórz bazę danych o nazwie warehouse.
3. Skonfiguruj połączenie w klasie DatabaseConnector.
4. Uruchom klasę Main, aby zainicjować aplikację i przetestować jej funkcjonalności.