

Framework e-commerce

Elastyczny szablon sklepu internetowego

Przemysław Magiera

29 listopada 2018

Część I

Zdefiniowane problemy

Architektura sklepów internetowych

Co można powiedzieć o architekturze i rozszerzalności sklepów internetowych?

- sklepy nieoparte o frameworki są trudne w utrzymaniu

Architektura sklepów internetowych

Co można powiedzieć o architekturze i rozszerzalności sklepów internetowych?

- sklepy nieoparte o frameworki są trudne w utrzymaniu
- dodawanie nowych funkcjonalności jest bardzo kosztowne

Architektura sklepów internetowych

Co można powiedzieć o architekturze i rozszerzalności sklepów internetowych?

- sklepy nieoparte o frameworki są trudne w utrzymaniu
- dodawanie nowych funkcjonalności jest bardzo kosztowne
- każda nowa funkcjonalność wymaga implementacji interfejsu do administracji i zarządzania

Wydajność i skalowalność

Jaka jest wydajność sklepów internetowych?

- często klasyczne sklepy wymagają skalowania pionowego, które jest bardzo drogie

Wydajność i skalowalność

Jaka jest wydajność sklepów internetowych?

- często klasyczne sklepy wymagają skalowania pionowego, które jest bardzo drogie
- frameworki nie oferują mechanizmów szybkiego dostępu do najbardziej kluczowych danych - katalog produktowy

Wydajność i skalowalność

Jaka jest wydajność sklepów internetowych?

- często klasyczne sklepy wymagają skalowania pionowego, które jest bardzo drogie
- frameworki nie oferują mechanizmów szybkiego dostępu do najbardziej kluczowych danych - katalog produktowy
- najbardziej obciążone punkty aplikacji nie są odseparowane od reszty

Katalog produktowy

Czy katalogi produktowe są zawsze proste i spełniają swoje zadanie?

- główne zadanie katalogu produktowego to zapewnienie łatwego i szybkiego sposobu na dotarcie do interesującej informacji

Katalog produktowy

Czy katalogi produktowe są zawsze proste i spełniają swoje zadanie?

- główne zadanie katalogu produktowego to zapewnienie łatwego i szybkiego sposobu na dotarcie do interesującej informacji
- brak możliwości konfiguracji i modyfikacji wyszukiwarki

Katalog produktowy

Czy katalogi produktowe są zawsze proste i spełniają swoje zadanie?

- główne zadanie katalogu produktowego to zapewnienie łatwego i szybkiego sposobu na dotarcie do interesującej informacji
- brak możliwości konfiguracji i modyfikacji wyszukiwarki
- trudność w dotarciu do informacji

Stosowane technologie

Czy frameworki są na tyle elastyczne aby nie zostać *w tyle*?

- stosowanie zamkniętych komercyjnych technologii

Stosowane technologie

Czy frameworki są na tyle elastyczne aby nie zostać *w tyle*?

- stosowanie zamkniętych komercyjnych technologii
- programiści nie widzą źródeł i nie mają wpływu na rdzeniowe elementy platformy, na której programują

Stosowane technologie

Czy frameworki są na tyle elastyczne aby nie zostać w tyle?

- stosowanie zamkniętych komercyjnych technologii
- programiści nie widzą źródeł i nie mają wpływu na rdzeniowe elementy platformy, na której programują
- w przypadku błędów, nadpisanie komponentów platformy jest bardzo trudne lub nawet niemożliwe

Część II

Zastosowane rozwiązania problemów

Dynamiczny panel administracyjny

Zarządzanie funkcjonalnościami out-of-the-box

- generowana tabelka dla każdej encji danego rodzaju np. kategorii

Dynamiczny panel administracyjny

Zarządzanie funkcjonalnościami out-of-the-box

- generowana tabelka dla każdej encji danego rodzaju np. kategorii
- generowany formularz edycji dowolnej encji

Dynamiczny panel administracyjny

Zarządzanie funkcjonalnościami out-of-the-box

- generowana tabelka dla każdej encji danego rodzaju np. kategorii
- generowany formularz edycji dowolnej encji
- generowany mechanizm zarządzania relacjami dowolnej klasy np. dzieci kategorii

Dynamiczne menu w panelu administracyjnym

Zarządzanie encjami

Przykład

Dodajemy nową klasę do systemu opartego na frameworku, chcemy mieć możliwość zarządzania tą klasą (operacje CRUD). Aby system widział tę klasę (i wszystkie ją rozszerzające) należy dodać rekord do tabeli `admin-menu-item` z nazwą klasy nowej encji. W ten sposób powstały wszystkie funkcjonalności *out-of-the-box*

- Menu w panelu jest oparte na dwóch tabelach `admin-menu-item` oraz `admin-menu-group`

Dynamiczne menu w panelu administracyjnym

Zarządzanie encjami

Przykład

Dodajemy nową klasę do systemu opartego na frameworku, chcemy mieć możliwość zarządzania tą klasą (operacje CRUD). Aby system widział tę klasę (i wszystkie ją rozszerzające) należy dodać rekord do tabeli `admin-menu-item` z nazwą klasy nowej encji. W ten sposób powstały wszystkie funkcjonalności *out-of-the-box*

- Menu w panelu jest oparte na dwóch tabelach `admin-menu-item` oraz `admin-menu-group`
- `AdminMenuItem` i `AdminMenuGroup` to też encje, więc konfigurujemy widok panelu admina

Dynamiczne menu w panelu administracyjnym

Zarządzanie encjami

Przykład

Dodajemy nową klasę do systemu opartego na frameworku, chcemy mieć możliwość zarządzania tą klasą (operacje CRUD). Aby system widział tę klasę (i wszystkie ją rozszerzające) należy dodać rekord do tabeli `admin-menu-item` z nazwą klasy nowej encji. W ten sposób powstały wszystkie funkcjonalności *out-of-the-box*

- Menu w panelu jest oparte na dwóch tabelach `admin-menu-item` oraz `admin-menu-group`
- `AdminMenuItem` i `AdminMenuGroup` to też encje, więc konfigurujemy widok panelu admina
- efekt jest taki, że każdą encję, którą dodamy do tych tabel będziemy mogli zarządzać z poziomu panelu admina

Zarządzanie nadpisanymi klasami przez panel

Zarządzanie funkcjonalnościami, które mogą zostać dopisane do platformy

Przykład

Programista decyduje się na zaimplementowanie nowej funkcjonalności, która będzie polegała na tym, że każde zamówienie będzie mogło mieć uwagę. W systemie istnieje encja `Order`, ma już zdefiniowane pola, ma również swój formularz. Nie jest to jednak problem, gdyż w tym przypadku wystarczy zaimplementować klasę `MyOrder` extends `Order`, która będzie zawierała pole `private String note`. Aplikacja korzystająca z frameworkowego panelu będzie świadoma tego, że encja `order` została nadpisana i będzie szukać w niej pól z aadnotacją `@AdminVisible`, które zostaną wyświetlone w dynamicznym formularzu i dynamicznej tabelce.

- platforma jest świadoma tego, że klasa którą zarządza panel administracyjny została nadpisana

Zarządzanie nadpisanymi klasami przez panel

Zarządzanie funkcjonalnościami, które mogą zostać dopisane do platformy

Przykład

Programista decyduje się na zaimplementowanie nowej funkcjonalności, która będzie polegała na tym, że każde zamówienie będzie mogło mieć uwagę. W systemie istnieje encja `Order`, ma już zdefiniowane pola, ma również swój formularz. Nie jest to jednak problem, gdyż w tym przypadku wystarczy zaimplementować klasę `MyOrder` extends `Order`, która będzie zawierała pole `private String note`. Aplikacja korzystająca z frameworkowego panelu będzie świadoma tego, że encja `order` została nadpisana i będzie szukać w niej pól z aadnotacją `@AdminVisible`, które zostaną wyświetlone w dynamicznym formularzu i dynamicznej tabelce.

- platforma jest świadoma tego, że klasa którą zarządza panel administracyjny została nadpisana
- generowanie formularza i tabeli odbywa się poprzez wgląd do klas rozszerzających bazową encję

Zarządzanie nadpisanymi klasami przez panel

Zarządzanie funkcjonalnościami, które mogą zostać dopisane do platformy

Przykład

Programista decyduje się na zaimplementowanie nowej funkcjonalności, która będzie polegała na tym, że każde zamówienie będzie mogło mieć uwagę. W systemie istnieje encja `Order`, ma już zdefiniowane pola, ma również swój formularz. Nie jest to jednak problem, gdyż w tym przypadku wystarczy zaimplementować klasę `MyOrder` extends `Order`, która będzie zawierała pole `private String note`. Aplikacja korzystająca z frameworkowego panelu będzie świadoma tego, że encja `order` została nadpisana i będzie szukać w niej pól z adnotacją `@AdminVisible`, które zostaną wyświetlone w dynamicznym formularzu i dynamicznej tabelce.

- platforma jest świadoma tego, że klasa którą zarządza panel administracyjny została nadpisana
- generowanie formularza i tabeli odbywa się poprzez wgląd do klas rozszerzających bazową encję
- daje to efekt, że właściwie każde pole w encji, którą zarządza platforma i jest zaadnotowane jako `@AdminVisible` jest widziane w tabeli encyjnej i formularzu encyjnym

Zarządzanie relacjami encji

Wyświetlanie i zarządzanie relacjami w dynamicznym formularzu encyjnym

Przykład

Encja `Product` jest w relacji `OneToMany` z encją `Price`. Aby panel administracyjny był w pełni wartościowy, po wejściu do formularza edycyjnego powinniśmy mieć możliwość dodania ceny do produktu. W tym celu należy umieścić adnotację `@AdminVisible(tableVisible = false, className = "com.example.Price", mappedBy = "product")` nad kolekcją cen. System w panelu administracyjnym wyświetli listę cen w produkcie.

- adnotacja `@AdminVisible(tableVisible = false, className = "com.example.ClassExample", mappedBy = "przyklad")` odpowiada również za wyświetlanie relacji w dynamicznym formularzu edycji.

Zarządzanie relacjami encji

Wyświetlanie i zarządzanie relacjami w dynamicznym formularzu encyjnym

Przykład

Encja `Product` jest w relacji `OneToMany` z encją `Price`. Aby panel administracyjny był w pełni wartościowy, po wejściu do formularza edycyjnego powinniśmy mieć możliwość dodania ceny do produktu. W tym celu należy umieścić adnotację `@AdminVisible(tableVisible = false, className = "com.example.Price", mappedBy = "product")` nad kolekcją cen. System w panelu administracyjnym wyświetli listę cen w produkcie.

- adnotacja `@AdminVisible(tableVisible = false, className = "com.example.ClassExample", mappedBy = "przyklad")` odpowiada również za wyświetlanie relacji w dynamicznym formularzu edycji.
- obsługiwane są relacje `OneToMany` jak i `ManyToMany` - a to nie jest oczywiste!

Zarządzanie relacjami encji

Wyświetlanie i zarządzanie relacjami w dynamicznym formularzu encyjnym

Przykład

Encja `Product` jest w relacji `OneToMany` z encją `Price`. Aby panel administracyjny był w pełni wartościowy, po wejściu do formularza edycyjnego powinniśmy mieć możliwość dodania ceny do produktu. W tym celu należy umieścić adnotację `@AdminVisible(tableVisible = false, className = "com.example.Price", mappedBy = "product")` nad kolekcją cen. System w panelu administracyjnym wyświetli listę cen w produkcie.

- adnotacja `@AdminVisible(tableVisible = false, className = "com.example.ClassExample", mappedBy = "przyklad")` odpowiada również za wyświetlanie relacji w dynamicznym formularzu edycji.
- obsługiwane są relacje `OneToMany` jak i `ManyToMany` - a to nie jest oczywiste!
- wyświetlane relacje można dowolnie modyfikować, usuwać lub dodawać (wszystko jest generowane)

Zarządzanie relacjami encji - jak działa

Jak dokonano implementacji dynamicznego zarządzania relacjami w encji

- Wyświetlenie
- Dodanie
- Usunięcie

Zarządzanie relacjami encji - jak działa

Jak dokonano implementacji dynamicznego zarządzania relacjami w encji

- Wyświetlenie
 - z adnotacji `@AdminVisible` weź nazwę klasy w relacji z encją, dla której generujesz formularz (`className`) oraz nazwę klucza obcego (`mappedBy`)
 - za pomocą dynamicznego dao encyjnego (`DynamicEntityDao`) wyciągnij z bazy danych listę encji powiązanych
- Dodanie
- Usunięcie

Zarządzanie relacjami encji - jak działa

Jak dokonano implementacji dynamicznego zarządzania relacjami w encji

- Wyświetlenie
 - z adnotacji `@AdminVisible` weź nazwę klasy w relacji z encją, dla której generujesz formularz (`className`) oraz nazwę klucza obcego (`mappedBy`)
 - za pomocą dynamicznego dao encyjnego (`DynamicEntityDao`) wyciągnij z bazy danych listę encji powiązanych
- Dodanie
 - znajdź pole w klasie (lub jej pochodnych) odnoszące się do zarządzanej relacji, analogicznie dla drugiej strony
 - Hibernate zwróci `PersistentSet`'y z encjami relacyjnymi, odpowiednio zmodyfikuj je i zakończ transakcję
 - co tu jest nie tak?
- Usunięcie

Zarządzanie relacjami encji - jak działa

Jak dokonano implementacji dynamicznego zarządzania relacjami w encji

- Wyświetlenie
 - z adnotacji `@AdminVisible` weź nazwę klasy w relacji z encją, dla której generujesz formularz (`className`) oraz nazwę klucza obcego (`mappedBy`)
 - za pomocą dynamicznego dao encyjnego (`DynamicEntityDao`) wyciągnij z bazy danych listę encji powiązanych
- Dodanie
 - znajdź pole w klasie (lub jej pochodnych) odnoszące się do zarządzanej relacji, analogicznie dla drugiej strony
 - Hibernate zwróci `PersistentSet`'y z encjami relacyjnymi, odpowiednio zmodyfikuj je i zakończ transakcje
 - co tu jest nie tak?
- Usunięcie
 - analogicznie jak dodanie, tylko usuń wartości z setów...

Funkcjonalności biznesowe

What's in the box?

- prawie wszystkie funkcjonalności biznesowe zostały zaimplementowane na podstawie dynamicznego panelu

Funkcjonalności biznesowe

What's in the box?

- prawie wszystkie funkcjonalności biznesowe zostały zaimplementowane na podstawie dynamicznego panelu
- implementacja ich ogranicza się do stworzenia modelu i napisania funkcji korzystających z tej konkretnej funkcjonalności w kontrolerze/htmlu/komponencie/fasadzie/JS na stronie sklepu/fasadzie

Funkcjonalności biznesowe

What's in the box?

- prawie wszystkie funkcjonalności biznesowe zostały zaimplementowane na podstawie dynamicznego panelu
- implementacja ich ogranicza się do stworzenia modelu i napisania funkcji korzystających z tej konkretnej funkcjonalności w kontrolerze/htmlu/komponencie/fasadzie/JS na stronie sklepu/fasadzie
- pełna obsługa encji i jej relacji robi się sama

System klasyfikacyjny

Schemat pobierania atrybutów produktowych z systemu klasyfikacyjnego

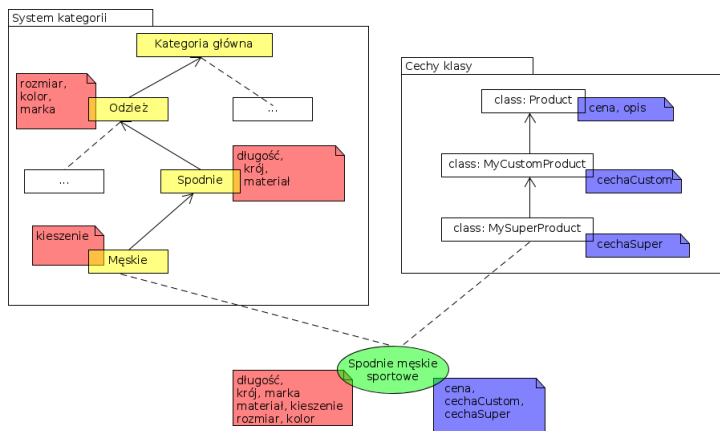
Definicja

Systemu klasyfikacyjny Jest to drzewiasta struktura kategorii połączona z encjami, które mogą przyjmować postać różnych cech produktu, definiować wartości tych cech łączyć je z produktem. Cechy są dziedziczne, zgodnie z ułożeniem drzewa kategorii.

- rysunek 1 bardzo dobrze obrazuje skąd system klasyfikacyjny bierze cechy produktu

System klasyfikacyjny

Schemat pobierania atrybutów produktowych z systemu klasyfikacyjnego



Rysunek: Diagram przykładowy pochodzenia atrybutów produktu

[TODO] System uprawnień

Jak będzie działać system uprawnień w panelu administracyjnym

- użytkownik administracyjny będzie miał kolekcję uprawnień

[TODO] System uprawnień

Jak będzie działać system uprawnień w panelu administracyjnym

- użytkownik administracyjny będzie miał kolekcję uprawnień
- struktura systemu uprawnień będzie podobna do struktury kategorii, uprawnienia będą mogły po sobie dziedziczyć

[TODO] System uprawnień

Jak będzie działać system uprawnień w panelu administracyjnym

- użytkownik administracyjny będzie miał kolekcję uprawnień
- struktura systemu uprawnień będzie podobna do struktury kategorii, uprawnienia będą mogły po sobie dziedziczyć
- AdminMenuGroup i AdminMenuItem będą miały kolekcje encji 'uprawnienie', podczas *renderu* menu, będą brane pod uwagę, względem tego, kto jest aktualnie zalogowany

Klasyczne funkcjonalności

Inne (niekoniecznie ciekawe) funkcjonalności, które muszą być w sklepie

- koszyk
- logowanie jako użytkownik sklepu
- możliwość dokonania zakupu
- zarządzanie zamówieniami/użytkownikami/adresami
- i pare innych...

Mechanizm indeksująco-wyszukujący

Integracja płaskiej bazy no-sql z bazą relacyjną

- do zapytań związanych z katalogiem produktowym użyto serwer Apache Solr 7.5 oparty na silniku Lucene

Mechanizm indeksująco-wyszukujący

Integracja płaskiej bazy no-sql z bazą relacyjną

- do zapytań związanych z katalogiem produktowym użyto serwer Apache Solr 7.5 oparty na silniku Lucene
- Solr umożliwia dobrą i wszechstronną analizę danych tekstowych, podświetlanie, podpowiadanie i facetowanie.

Mechanizm indeksująco-wyszukujący

Integracja płaskiej bazy no-sql z bazą relacyjną

- do zapytań związanych z katalogiem produktowym użyto serwer Apache Solr 7.5 oparty na silniku Lucene
- Solr umożliwia dobrą i wszechstronną analizę danych tekstowych, podświetlanie, podpowiadanie i facetowanie.
- (TODO) co jakiś czas uruchamiany jest Job w systemie, który synchronizuje bazę relacyjną z Solrem (na razie za każdym requestem)

Mechanizm indeksująco-wyszukujący

Integracja płaskiej bazy no-sql z bazą relacyjną

- do zapytań związanych z katalogiem produktowym użyto serwer Apache Solr 7.5 oparty na silniku Lucene
- Solr umożliwia dobrą i wszechstronną analizę danych tekstowych, podświetlanie, podpowiadanie i facetowanie.
- (TODO) co jakiś czas uruchamiany jest Job w systemie, który synchronizuje bazę relacyjną z Solrem (na razie za każdym requestem)
- Indeksacja to wydobywanie atrybutów wyszukiwalnych i facetowalnych z produktu, zapakowanie w dokumenty i wysłanie na serwer Solr - wyciągnięcie atrybutów pokazuje rysunek 1

Indeksacja atrybutów produktu

Jak zachowano elastyczność?

- atrybuty nie pochodzące z systemu klasyfikacyjnego są również indeksowane

Indeksacja atrybutów produktu

Jak zachowano elastyczność?

- atrybuty nie pochodzące z systemu klasyfikacyjnego są również indeksowane
- pola w klasach rozszerzających `Product` (np. `MyProduct extends Product`) są również możliwe do zaindeksowania, wystarczy że z poziomu panelu administracyjnego dodamy do tabeli `SearchField` nazwę pola, reszta zrobi się sama

Część III

Przykłady działania

```
@AllArgsConstructor  
public class Price extends AbstractEntity {  
  
    @ManyToOne  
    @JoinColumn(name = "product_id")  
    private Product product;  
  
    @Column(name = "currency")  
    private Currency currency;  
  
    @Column(name = "amount")  
    private BigDecimal amount;  
}
```

Rysunek: Klasa Price

Dodanie encji do systemu

Tine-commerce admin panel

Catalog

System

Admin group items

Jobs

Admin groups

Security

Commerce

Classification system

ADD

Table of adminGroupItem

Show 10 entries Search:

Id	code	className	name	friendlyName
-1011	adminGroup	com.tinecommerce.admin.panel.model.AdminMenuGroup	AdminGroup	Admin groups
-1010	adminGroupItem	com.tinecommerce.admin.panel.model.AdminMenuItem	AdminGroupItem	Admin group items
-1009	categoryFeature	com.tinecommerce.core.catalog.model.CategoryFeature	CategoryFeature	Category Features
-1008	categoryFeatureValue	com.tinecommerce.core.catalog.model.CategoryFeatureValue	CategoryFeatureValue	Feature Values
-1007	categoryFeatureAssignment	com.tinecommerce.core.catalog.model.CategoryFeatureAssignment	CategoryFeatureAssignment	Feature Assignments
-1006	order	com.tinecommerce.core.cart.Order	Order	Orders
-1005	customer	com.tinecommerce.core.customer.model.Customer	Customer	Customers
-1004	job	tba	tba	Jobs
-1003	price	com.tinecommerce.core.catalog.model.Price	Price	Prices
-1002	category	com.tinecommerce.core.catalog.model.Category	Category	Categories

Showing 1 to 10 of 12 entries

Previous 1 2 Next

Rysunek: Dodanie klasy Price do obsługi systemu

Dodanie encji do systemu

Details of adminMenuItem

code

className

name

friendlyName

Rysunek: Podstawowa tabelka encyjna - formularz

Dodanie encji do systemu

Catalog

- Products
- Categories
- Prices

System

Security

Commerce

Classification system

ADD

Table of price

Search:

Show 10 entries

Id	code
-105	kod6
-104	kod5
-103	kod4
-102	kod3
-101	kod2
-100	kod1

Showing 1 to 6 of 6 entries

Previous1Next

Rysunek: Podstawowa tabelka encyjna

Dodanie encji do systemu

```
@AllArgsConstructor  
public class Price extends AbstractEntity {  
  
    @ManyToOne  
    @JoinColumn(name = "product_id")  
    private Product product;  
  
    @Column(name = "currency")  
    @AdminVisible  
    private Currency currency;  
  
    @Column(name = "amount")  
    @AdminVisible  
    private BigDecimal amount;  
}
```

Rysunek: Dopisanie adnotacji @AdminVisible nad polami ceny

Dodanie encji do systemu

Tine-commerce admin panel

Catalog

System

Security

Commerce

Classification system

ADD

Table of price

Show 10 entries Search:

id	code	currency	amount
-105	kod6	PLN	179.90
-104	kod5	PLN	299.90
-103	kod4	PLN	349.90
-102	kod3	PLN	149.00
-101	kod2	PLN	229.90
-100	kod1	PLN	249.90

Showing 1 to 6 of 6 entries

Previous **1** Next

Rysunek: Wyświetlenie zaadnotowanych pól

Dodanie encji do systemu

```
public class Product extends AbstractNameableEntity {  
    @Setter(AccessLevel.NONE)  
    @OneToMany(cascade = CascadeType.ALL, orphanRemoval = true, mappedBy = "product")  
    @AdminVisible(tableVisible = false, className = "com.tinecommerce.core.catalog.model.Price")  
    private Set<Price> prices;  
    @Setter(AccessLevel.NONE)  
    @OneToMany(cascade = CascadeType.ALL, orphanRemoval = true, mappedBy = "product")  
    @AdminVisible(tableVisible = false, className = "com.tinecommerce.core.catalog.model.ProductFeature")  
    private Set<ProductFeature> productFeatures;  
    @Setter(AccessLevel.NONE)  
    @ManyToOne(mappedBy = Category.FIELD_PRODUCTS, cascade = CascadeType.ALL, fetch = FetchType.LAZY)  
    @AdminVisible(tableVisible = false, className = "com.tinecommerce.core.catalog.model.Category", mappedBy = "products")  
    private Set<Category> categories = new HashSet<>();  
}
```

Rysunek: Naniesienie adnotacji admin visible nad kolekcją klasy produkt

Dodanie encji do systemu

Details of product

id
-105

code
48051878

name
Trampki vans classic

description
Buty cichobiegł, jest idealną propozycją dla mężczyzn ceniących sobie elegancję a zarazem wygodę i styl.

[Submit](#)
[Delete](#)

Relations of product

prices
[Add](#)

id	code	currency	amount
----	------	----------	--------

categories
[Add](#)

id	code	name	description
-105	obuwieTramp	Trampki	X

productFeatures
[Add](#)

id	code	value
----	------	-------

Rysunek: Wygenerowany formularz zmiany relacji

Dodanie encji do systemu

Modal dialog titled "Detail" with a close button (X) in the top right corner. The dialog contains a list of entities to add, each with an ID, code, name, and price. The list is as follows:

ID	Code	Name	Price
-100	kod1	PLN	249.90
-101	kod2	PLN	229.90
-102	kod3	PLN	149.00
-103	kod4	PLN	349.90
-104	kod5	PLN	299.90
-105	kod6	PLN	179.90

Below the list, there are buttons for "Sub", "Delete", and "Add". The "Add" button is highlighted in green. The background of the application shows a section titled "Relations of product" with a table header:

id	code	currency	amount
----	------	----------	--------

Rysunek: Zarządzanie relacją

Dodanie atrybutu klasyfikacyjnego

Catalog

System

Security

Commerce

Classification system

Feature Assignments

Feature Values

Category Features

Product Feature

ADD

Table of categoryFeature

Show 10 entries

Search:

id	code	name	description	type	searchable	isFacet
-106	podszewkaObuwie	Podszewka		ENUM	false	true
-105	dlugoscSpodnie	Długość		ENUM	false	true
-104	marka	Marka		STRING	true	true
-103	material	Material		STRING	false	true
-102	kolor	Kolor		STRING	false	true
-101	rozmiarOdziez	Rozmiar		INTEGER	false	true
-100	rozmiarObuwie	Rozmiar		ENUM	false	true
-99	testowyFeature	feature1			false	false

Showing 1 to 8 of 8 entries

Previous 1 Next

Rysunek: Dodanie nowego atrybutu klasyfikacyjnego

Dodanie atrybutu klasyfikacyjnego

Catalog
System
Security
Commerce
Classification system
Feature Assignments
Feature Values
Category Features
Product Feature

ADD

Table of categoryFeatureValue

Show 10 entries

Search:

Id	code	value	categoryFeature
-109	podszewkaWelna	Welna	Podszewka - podszewkaObuwie
-108	podszewkaMaterial	Material	Podszewka - podszewkaObuwie
-107	podszewkaSkora	Skóra	Podszewka - podszewkaObuwie
-106	spodDlugie	Długie	Długość - dlugoscSpodnie
-105	spodnieKrot	Krótkie	Długość - dlugoscSpodnie
-104	rozmXL	XL	Rozmiar - rozmiarOdziez
-103	rozmL	L	Rozmiar - rozmiarOdziez
-102	rozmM	M	Rozmiar - rozmiarOdziez
-101	rozmS	S	Rozmiar - rozmiarOdziez

Showing 1 to 9 of 9 entries

Previous 1 Next

Rysunek: Dodanie jego wariantów

Dodanie atrybutu klasyfikacyjnego

Catalog

System

Security

Commerce

Classification system

Feature Assignments

Feature Values

Category Features

Product Feature

ADD

Table of categoryFeatureAssignment

Show 10 entries

Search:

Id	code	categoryFeature	category
-107	dlugoscSpodnie	Długość - dlugoscSpodnie	Spodnie- code:odzSpodnie
-106	podszObuwie	Podszewka - podszewkaObuwie	Obuwie- code:cat2
-105	material	Material - material	Odzież- code:cat1
-104	odziezKolor	Kolor - kolor	Odzież- code:cat1
-103	obuwKolor	Kolor - kolor	Obuwie- code:cat2
-102	odziezRozm	Rozmiar - rozmiarOdziez	Odzież- code:cat1
-101	obuwRozm	Rozmiar - rozmiarObuwie	Obuwie- code:cat2
-100	rootMarka	Marka - marka	Main category [root]- code:root
-99	testowyDoKategoriiRoot		Main category [root]- code:root

Showing 1 to 9 of 9 entries

Previous 1 Next

Rysunek: Przypisanie atrybutu do kategorii

Dodanie atrybutu klasyfikacyjnego

Details of product

id
-102

code
40457878

name
Koszulka polo wyjściowa

description
Koszulka polo Nike, jest idealną propozycją dla mężczyzn ceniących sobie elegancję a zarazem wygodę i styl.

[Save](#) [Delete](#)

Relations of product

prices

[Add](#)

id	code	currency	amount	
-102	kod3	PLN	149.00	X

categories

[Add](#)

id	code	name	description	
-107	odaKoszulki	Koszulki		X

productFeatures

[Add](#)

id	code	value	
-102	pf3	S	X
-105	pf6	Nike	X

Rysunek: Pojawienie się atrybutu w produkcie, który jest dzieckiem kategorii do której został przypisany atrybut

Dodanie atrybutu klasyfikacyjnego

Tine-commerce: przykładowy sklep

Home About Services Contact

Filtry

Cena

100.0 - 150.0(1)

200.0 - 250.0(2)

??facet_name_rozmiarObuwie_t

hilfiger (1)

lacoste (1)

nike (1)

Rozmiar

l (1)

s (1)

xl (1)

Kategoria

koszulki (3)

Wyszukaj

koszulka

700 x 400

Koszulka polo Lacoste

229.9zł

Koszulka polo Lacoste, jest idealną propozycją dla mężczyzn ceniących sobie elegancję a zarazem wygodę i styl.

Add to cart

700 x 400

Koszulka polo wyjściowa

149.0zł

Koszulka polo Nike, jest idealną propozycją dla mężczyzn ceniących sobie elegancję a zarazem wygodę i styl.

Add to cart

700 x 400

Koszulka polo Hilfiger slim

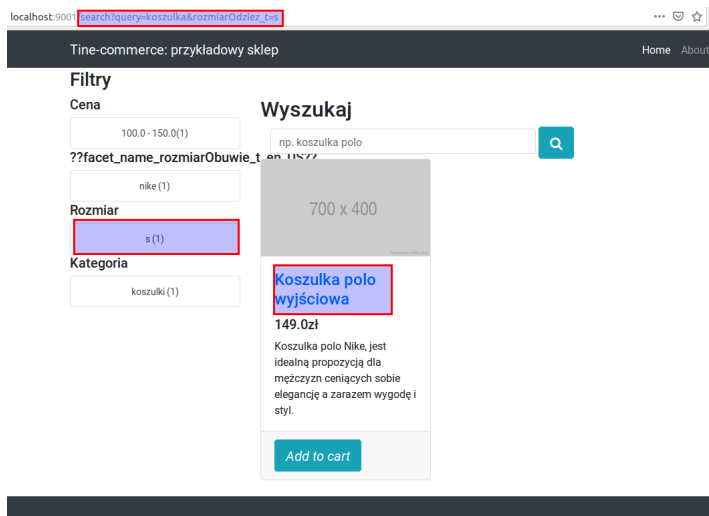
249.9zł

Koszulka polo Tommy Hilfiger, jest idealną propozycją dla mężczyzn ceniących sobie elegancję a zarazem wygodę i styl.

Add to cart

Rysunek: Wyszukiwanie koszulek i wyświetlanie facetów

Dodanie atrybutu klasyfikacyjnego



Rysunek: Przefiltrowanie koszulki po rozmiarze

Dziękuję za uwagę