

# Projekt zaliczeniowy



## Programowanie obiektowe Rok akademicki 2024/2025

### Autorzy:

Hanna Radzivonchuk  
Zuzanna Szafraniec  
Jadwiga Trojan  
Julia Wróbel

### Sklep z Książkami

Projekt przedstawia aplikację do zarządzania magazynem książek. Użytkownik może wprowadzać zmiany w magazynie, takie jak dodawanie i usuwanie książek, sortowanie ich alfabetycznie lub według ceny, a także filtrowanie książek według kategorii. Dodatkowo aplikacja umożliwia wyświetlenie aktualnej wartości książek w magazynie, drukowanie stanu magazynu, sprzedaż i dostawę książek, wyszukiwanie konkretnych pozycji oraz połączenie się ze stroną internetową hurtowni.

### Podział ról

#### Hanna Radzivonczyk

- Przygotowanie diagramu klas.
- Implementacja serializacji obiektów do XML.
- Wdrożenie interfejsów: ICloneable, IComparable, IEquatable.
- Implementacja klasy LiteraturaFaktu.
- Implementacja klasy Fantastyka.
- Obsługa wyjątków w projekcie.
- Funkcje sortujące w klasie Magazyn.
- Metoda wyszukiwania książek po tytule.
- Dodanie części książek do magazynu.

#### Zuzanna Szafraniec

- Implementacja klasy Ksiazka.
- Implementacja klasy Kryminal.
- Implementacja klasy Podrecznik.
- Implementacja klasy Romans.
- W klasie Magazyn zaimplementowano metody:

- ObliczWartosc.
  - DodajDoMagazynu.
  - Dostawa.
  - Sprzedaj.
  - UsunKsiazke.
  - WyszukajPoAutorze.
  - WyszukajPoRodzaju.
  - WyszukajPoKlasie.
  - WyszukajPoPrzedmiocie.
  - ToString.
- Dodanie części książek do magazynu.

### **Jadwiga Trojan**

- Implementacja interfejsu graficznego aplikacji (GUI).
- Stworzenie i obsługa okien w GUI:
  - Heaven – główne okno nawigacyjne.
  - MagazynWindow – okno zarządzania magazynem.
  - Dostawa – okno obsługi dostaw książek.
  - Sprzedane – okno obsługi sprzedaży książek.
  - Wyszukaj – okno wyszukiwania książek.
  - Hurtownia – okno do integracji ze stroną internetową hurtowni.
  - MainWindow – okno logowania.
- Integracja GUI z funkcjami aplikacji:
  - Wyświetlanie książek w magazynie.
  - Obsługa przycisków i formularzy.
  - Przekierowania między oknami.

### **Julia Wróbel**

- Przygotowanie testów jednostkowych dla projektu.
- Implementacja i przeprowadzenie testów dla klas:
  - Ksiazka.
  - Fantastyka.
  - Kryminal.
  - Romans.
  - LiteraturaFaktu.
  - Podrecznik.
  - Magazyn.
- Obsługa mechanizmu „Dodaj Książkę” w klasie MagazynWindow.
- Przygotowanie dokumentacji projektu.

## Klasa Program

W tej klasie znajduje się funkcja **Main()**, czyli główny punkt startowy programu. Najpierw tworzymy obiekt **Magazyn**, który będzie przechowywał wszystkie nasze książki. Potem tworzymy różne książki następnie tworzymy książki z gatunków Kryminał, Podręcznik, Literatura Faktu, Fantastyka, Romans. Wszystkie te książki dodajemy do naszego magazynu za pomocą funkcji **DodajDoMagazynu()**. Sprawdzamy też, czy działają inne funkcje magazynu, takie jak **Dostawa()**, **Sprzedaj()** i **ObliczWartosc()**. Wyniki tych operacji wypisujemy na ekran.

Na koniec zapisujemy magazyn do pliku XML i sprawdzamy czy możemy go poprawnie odczytać. Dzięki temu widzimy, że metody **ZapiszDCXML()** i **OdczytajDCXML()** działają tak, jak powinny.

## Klasa Magazyn

Ta klasa reprezentuje magazyn naszego sklepu z książkami, w którym przechowywane są wszystkie utworzone książki. Odpowiada za zarządzanie kolekcją książek, umożliwiając m.in. dodawanie, usuwanie, wyszukiwanie, sortowanie oraz obliczanie wartości magazynu.

**Pola:**

- **public string nazwa** – Przechowuje nazwę magazynu.
- **public List<Ksiazka> ksiazki** – Lista przechowująca obiekty klasy **Ksiazka** i jej pochodnych.

**Konstruktory:**

- **Magazyn(string nazwa)** – Inicjalizuje magazyn, nadając mu nazwę oraz tworzy pustą listę książek.
- **Konstruktor bezparametrowy** – Ułatwia serializację i tworzenie pustego magazynu.

**Metody:**

- **ObliczWartosc()** – Zwraca łączną wartość magazynu, sumując iloczyn ceny i ilości każdej książki. Korzysta z metody **ObliczCene()** z klasy **Ksiazka**.
- **DodajDoMagazynu()** – Dodaje nową książkę do magazynu. Jeśli książka o tym samym numerze ISBN już istnieje, rzucany jest wyjątek.
- **Dostawa()** – Zwiększa stan magazynowy książki o podanym numerze ISBN.
- **Sprzedaj()** – Zmniejsza stan magazynowy książki
- **UsunKsiazke()** – Całkowicie usuwa książkę o danym numerze ISBN z listy.
- Metody wyszukiwania (**WyszukajPoRodzaju()**, **WyszukajPoAutorze()**, **WyszukajPoTytule()**, **WyszukajPoKlasie()**, **WyszukajPoPrzedmiocie()**) umożliwiają filtrowanie listy książek według różnych kryteriów, takich jak rodzaj, autor, tytuł, klasa czy przedmiot.
- Metody sortujące (**SortujKsiazkiPoTytule()**, **SortujKsiazkiPoCenieRosnaco()**, **SortujKsiazkiPoCenieMalejaco()**) pozwalają uporządkować książki według wybranego kryterium: Tytuł, Cena rosnąca bądź cena malejąca.

- **ZapiszDCXML()** – Zapisuje stan magazynu do pliku XML.
- **OdczytajDCXML()** – Odczytuje dane magazynu z pliku XML o podanej nazwie i zwraca nowy obiekt klasy Magazyn.
- **Clone()** – Tworzy głęboką kopię magazynu za pomocą serializacji.
- **ToString()** – Zwraca tekstowy opis magazynu, zawierający nazwę magazynu i szczegóły każdej przechowywanej książki.

#### Uwagi dotyczące modyfikatorów dostępu:

Metody są publiczne, aby można było nimi zarządzać z klasy Program, a pola z atrybutem [DataMember] umożliwiają łatwą serializację magazynu i zapis danych do pliku.

## Klasa Książka

Klasa Książka jest podstawową klasą, która reprezentuje ogólną strukturę książki. Zawiera informacje takie jak tytuł, autor, cena, data publikacji, liczba egzemplarzy oraz liczba stron. Służy jako baza dla innych klas, które dziedziczą po niej: **Kryminal**, **Romans**, **Fantastyka**, **LiteraturaFaktu** i **Podręcznik**.

#### Pola:

- **private static long globalIsbn** – Statyczne pole, które zapewnia unikalny numer ISBN dla każdej nowej książki.
- **private long isbn** – Numer ISBN przypisany do konkretnej książki.
- **private string tytuł** – Przechowuje tytuł książki.
- **private string autor** – Przechowuje nazwisko autora książki.
- **private decimal cenaBazowa** – Bazowa cena książki.
- **private DateTime dataPublikacji** – Data publikacji książki.
- **private int ilosc** – Liczba egzemplarzy książki dostępnych w magazynie.
- **private int liczbaStron** – Liczba stron w książce.

Niektóre pola są oznaczone atrybutami [DataMember], aby mogły być serializowane, np. podczas zapisywania obiektu do pliku XML.

#### Konstruktory:

- **Książka()**  
Domyślny konstruktor, który przypisuje książce podstawowe wartości:
  - Pusty tytuł i autor,
  - Cenę bazową równą 10,
  - Aktualną datę jako datę publikacji,

- Liczbę stron w zakresie 50–500 (losowaną),
- Automatycznie generowany numer ISBN.
- **Ksiazka(string tytuł, string autor, decimal cena, string dataPublikacji, int ilosc)**  
Konstruktor parametryczny, który pozwala na utworzenie książki z podanymi danymi. Sprawdza, czy:
  - Data publikacji ma poprawny format (jeśli nie, rzuca wyjątek **NiepoprawnyFormatDaty**),
  - Cena i liczba egzemplarzy są większe od 0 (w przeciwnym razie rzuca wyjątek **NiepoprawnaWartość**).

#### Metody:

- **ObliczCene()**  
Metoda zwraca bazową cenę książki. Jest wirtualna, więc klasy dziedziczące mogą ją nadpisać, np. uwzględniając dodatkowe koszty dla określonych gatunków.
- **ObliczCzasCzytania()**  
Oblicza szacowany czas czytania książki, przyjmując, że przeczytanie jednej strony zajmuje średnio 2 minuty. Metoda jest wirtualna i może być zmodyfikowana w klasach pochodnych.
- **ToString()**  
Zwraca tekstowy opis książki, zawierający tytuł, autora, liczbę egzemplarzy, liczbę stron oraz datę publikacji.
- **CompareTo()**  
Umożliwia porównywanie książek według tytułu. Jeśli tytuły są takie same, porównuje daty publikacji. Dzięki temu książki można sortować.
- **OnSerializing()** – Zapisuje datę publikacji w formacie tekstowym przed serializacją.
- **OnDeserialized()** – Odczytuje datę publikacji z tekstowego formatu po deserializacji.

#### Obsługa wyjątków:

Klasa obsługuje dwa wyjątki:

- **NiepoprawnaWartość** – Rzucany, gdy liczba egzemplarzy lub cena są mniejsze lub równe 0.
- **NiepoprawnyFormatDaty** – Rzucany, gdy data publikacji nie jest w jednym z akceptowanych formatów.

Pola takie jak **tytuł**, **autor**, **cenaBazowa** i inne zostały oznaczone jako **private**, aby ograniczyć ryzyko nieprawidłowych operacji na danych. Dostęp do tych pól jest kontrolowany za pomocą publicznych właściwości **get** i **set**, co pozwala na bezpieczne zarządzanie danymi. Metody takie jak **ObliczCene()**, **ObliczCzasCzytania()** czy **ToString()** są oznaczone jako **public**, ponieważ muszą być dostępne w innych częściach programu. Pole **globalIsbn** zostało oznaczone jako **private static**, aby jego dostępność była ograniczona wyłącznie do klasy i aby zapobiec przypadkowym modyfikacjom spoza niej.

# Klasa Romans

Klasa Romans dziedziczy po klasie Książka. Reprezentuje książki należące do gatunku romans. Rozszerza podstawową funkcjonalność klasy Książka, dodając dodatkowe cechy charakterystyczne dla tego gatunku, takie jak rodzaj romansu czy dodatkowa cena.

## Pola:

- **private string id** – Identyfikator książki, unikalny dla każdego romansu, generowany na podstawie numeru ISBN.
- **public EnumRodzaj rodzaj** – Rodzaj romansu (Współczesny, Historyczny, Młodzieżowy, Tragiczny). Pole jest oznaczone jako public, ponieważ wykorzystywane jest w GUI lub przy filtrowaniu danych.
- **private decimal dodatekDoCeny** – Dodatkowa kwota, która jest doliczana do bazowej ceny książki. Jest stała i wynosi 10.

## Konstruktor:

- **Romans (string tytuł, string autor, decimal cena, string dataPublikacji, int ilosc, string rodzaj)**  
Konstruktor inicjalizuje obiekt książki typu romans.
  - Ustawia identyfikator książki (**id**) w formacie {ISBN}-R.
  - Przypisuje wartość **dodatekDoCeny** wynoszącą 10.
  - Sprawdza, czy podany rodzaj książki istnieje w wyliczeniu **EnumRodzaj**. Jeśli nie, rzuca wyjątek **NiepoprawnyRodzaj**.

## Metody:

- **ObliczCene()**  
Nadpisuje metodę z klasy bazowej i dolicza do ceny bazowej wartość z pola **dodatekDoCeny**.
- **ObliczCzasCzytania()**  
Nadpisuje metodę z klasy bazowej i zwiększa czas czytania o 50% w porównaniu do standardowego czasu czytania książki.
- **ToString()**  
Nadpisuje metodę **ToString()**, aby zwrócić szczegółowe informacje o książce, takie jak identyfikator, dane książki, rodzaj romansu oraz cena po uwzględnieniu dodatku.
- **Equals()**  
Implementuje porównywanie obiektów klasy **Romans**. Dwie książki są uznawane za identyczne, jeśli mają ten sam numer ISBN, rodzaj, datę publikacji i autora.
- **Clone ()**  
Implementuje interfejs **ICloneable** i pozwala na stworzenie płytkiej kopii obiektu za pomocą metody **MemberwiseClone()**.

### Uwagi dotyczące modyfikatorów dostępu:

Pola takie jak **id** i **dodatekDoCeny** są oznaczone jako **private**, aby zapewnić kontrolowany dostęp do nich tylko w ramach klasy. Pole **rodzaj** jest oznaczone jako **public**, aby umożliwić łatwy dostęp do rodzaju książki w GUI oraz przy filtrowaniu i wyświetlaniu informacji o książkach.

## Klasa Kryminał

Klasa **Kryminał** dziedziczy po klasie **Ksiazka** i reprezentuje książki należące do gatunku kryminał. Rozszerza podstawową funkcjonalność klasy **Ksiazka**, dodając unikalne cechy charakterystyczne dla kryminałów, takie jak rodzaj kryminału i dodatkowa cena.

### Pola:

- **private string id** – Identyfikator książki, unikalny dla każdego kryminału. Generowany automatycznie na podstawie numeru ISBN w formacie {ISBN}-K.
- **public EnumRodzaj rodzaj** – Rodzaj kryminału (Detektywistyczny, Sensacyjny, Psychologiczny, Polityczny). Pole jest **public**, aby umożliwić łatwe filtrowanie i wyświetlanie informacji.
- **private decimal dodatekDoCeny** – Dodatkowa opłata doliczana do ceny bazowej książki, charakterystyczna dla tego gatunku. Wartość jest stała i wynosi 15.

### Konstruktor:

- **Kryminał(string tytuł, string autor, decimal cena, string dataPublikacji, int ilosc, string rodzaj)**  
Konstruktor inicjalizuje obiekt książki z gatunku kryminał.
  - Generuje unikalny identyfikator książki (**id**) w formacie {ISBN}-K.
  - Przypisuje wartość **dodatekDoCeny** wynoszącą 15.
  - Sprawdza, czy podany rodzaj książki istnieje w wyliczeniu **EnumRodzaj**. Jeśli nie, rzuca wyjątek **NiepoprawnyRodzaj**.

### Metody:

1. **ObliczCene()**  
Nadpisuje metodę z klasy bazowej i dolicza do ceny bazowej wartość z pola **dodatekDoCeny**.
2. **ObliczCzasCzytania()**  
Nadpisuje metodę z klasy **Ksiazka** i podwaja czas czytania w porównaniu do standardowego czasu.
3. **ToString()**  
Nadpisuje metodę **ToString()** i zwraca szczegółowe informacje o książce, takie jak identyfikator, dane książki, rodzaj kryminału oraz cena z dodatkiem.
4. **Equals()**  
Implementuje porównywanie obiektów klasy **Kryminał**. Dwie książki są uznawane za identyczne, jeśli mają ten sam ISBN, rodzaj, datę publikacji i autora.

## 5. Clone()

Implementuje interfejs **ICloneable** i umożliwia stworzenie kopii obiektu za pomocą metody **MemberwiseClone()**.

### Uwagi dotyczące modyfikatorów dostępu:

Pola **id** i **dodatekDoCeny** są oznaczone jako **private**, aby zapewnić ich kontrolowany dostęp tylko w ramach klasy. Pole **rodzaj** jest **public**, aby umożliwić łatwy dostęp do informacji o rodzaju kryminału w GUI oraz przy filtrowaniu i wyświetlaniu informacji o książkach.

## Klasa Fantastyka

Klasa **Fantastyka** dziedziczy po klasie **Ksiazka** i reprezentuje książki należące do gatunku fantastyka. Rozszerza podstawową funkcjonalność klasy **Ksiazka**, dodając unikalne cechy charakterystyczne dla fantastyki, takie jak rodzaj literatury fantastycznej oraz dodatkowa cena.

### Pola:

- **private string id**  
Identyfikator książki, unikalny dla każdego egzemplarza fantastyki. Generowany automatycznie na podstawie numeru ISBN w formacie {ISBN}-Ft.
- **public EnumRodzaj rodzaj**  
Rodzaj literatury fantastycznej (Horror, Fantasy, ScienceFiction). Pole jest oznaczone jako **public**, ponieważ wykorzystywane jest w GUI lub przy filtrowaniu danych.
- **private decimal dodatekDoCeny**  
Dodatkowa opłata doliczana do ceny bazowej książki, charakterystyczna dla tego gatunku. Wartość jest stała i wynosi 15.
- 

### Konstruktor:

- **Fantastyka(string tytuł, string autor, decimal cena, string dataPublikacji, int ilosc, string rodzaj)**  
Konstruktor inicjalizuje obiekt książki z gatunku fantastyka.
  - Generuje unikalny identyfikator książki (id) w formacie {ISBN}-Ft.
  - Przypisuje wartość **dodatekDoCeny** równą 15.
  - Sprawdza, czy podany rodzaj książki istnieje w wyliczeniu **EnumRodzaj**. Jeśli nie, rzuca wyjątek **NiepoprawnyRodzaj**.



### Metody:

1. **ObliczCene()**  
Nadpisuje metodę z klasy bazowej i dolicza do ceny bazowej wartość z pola **dodatekDoCeny**.
2. **ObliczCzasCzytania()**  
Nadpisuje metodę z klasy bazowej. Dla książek z gatunku fantastyka czas czytania jest obliczany w standardowy sposób, czyli proporcjonalnie do liczby stron.
3. **ToString()**  
Nadpisuje metodę **ToString()**, aby zwrócić szczegółowe informacje o książce, takie jak identyfikator, dane książki, rodzaj literatury fantastycznej oraz cena z uwzględnieniem dodatku.
4. **Equals()**  
Implementuje porównywanie obiektów klasy **Fantastyka**. Dwie książki są uznawane za identyczne, jeśli mają ten sam numer ISBN, rodzaj, datę publikacji i autora.
5. **Clone()**  
Implementuje interfejs **ICloneable** i umożliwia stworzenie płytkiej kopii obiektu za pomocą metody **MemberwiseClone()**.

### Uwagi dotyczące modyfikatorów dostępu:

Pola takie jak **id** i **dodatekDoCeny** są oznaczone jako **private**, aby zapewnić kontrolowany dostęp do nich tylko w ramach klasy. Pole **rodzaj** jest oznaczone jako **public**, aby umożliwić łatwy dostęp do rodzaju książki w GUI oraz przy filtrowaniu i wyświetlaniu informacji o książkach.

## Klasa LiteraturaFaktu

Klasa **LiteraturaFaktu** dziedziczy po klasie **Ksiazka** i reprezentuje książki należące do gatunku fantastyka. Rozszerza podstawową funkcjonalność klasy **Ksiazka**, dodając unikalne cechy charakterystyczne dla Literatury Faktu, takie jak rodzaj oraz dodatkowa cena.

### Pola:

- **private string id**  
Unikalny identyfikator książki, generowany automatycznie na podstawie numeru ISBN w formacie {ISBN}-F.
- **public EnumRodzaj rodzaj**  
Rodzaj literatury faktu (Reportaż, Biografia, Pamiętnik). Pole jest oznaczone jako publiczne, aby umożliwić łatwe filtrowanie i wyświetlanie informacji w GUI lub przy filtrowaniu danych.
- **private decimal dodatekDoCeny**  
Stała opłata doliczana do ceny bazowej książki, charakterystyczna dla literatury faktu. Wartość wynosi 30.

### Konstruktor:

- **LiteraturaFaktu(string tytuł, string autor, decimal cena, string dataPublikacji, int ilosc, string rodzaj)**

Konstruktor inicjalizuje obiekt książki z gatunku literatury faktu.

- Generuje unikalny identyfikator książki (**id**) w formacie {ISBN}-F.
- Przypisuje wartość **dodatekDoCeny** równą 30.
- Sprawdza, czy podany rodzaj książki istnieje w wyliczeniu **EnumRodzaj**. Jeśli rodzaj jest nieprawidłowy, rzuca wyjątek **NiepoprawnyRodzaj**.

### Metody:

1. **ObliczCene()**  
Nadpisuje metodę z klasy bazowej i dolicza do ceny bazowej wartość z pola **dodatekDoCeny**.
2. **ObliczCzasCzytania()**  
Nadpisuje metodę z klasy bazowej, zwiększając czas czytania 2,5-krotnie w porównaniu do standardowego czasu.
3. **ToString()**  
Nadpisuje metodę **ToString()** i zwraca szczegółowe informacje o książce, takie jak identyfikator, dane książki, rodzaj literatury faktu oraz cena z dodatkiem.
4. **Equals()**  
Implementuje porównywanie obiektów klasy **LiteraturaFaktu**. Dwie książki są uznawane za identyczne, jeśli mają ten sam numer ISBN, rodzaj, datę publikacji i autora.
5. **Clone()**  
Implementuje interfejs **ICloneable** i umożliwia stworzenie płytkiej kopii obiektu za pomocą metody **MemberwiseClone()**.

### Uwagi dotyczące modyfikatorów dostępu:

Pola **id** i **dodatekDoCeny** są oznaczone jako **private**, aby zapewnić ich kontrolowany dostęp tylko w ramach klasy. Pole **rodzaj** jest **public**, aby umożliwić łatwy dostęp do informacji o rodzaju literatury faktu w GUI oraz przy filtrowaniu i wyświetlaniu informacji o książkach.

# Klasa Podrecznik

**Klasa Podrecznik** dziedziczy po klasie **Ksiazka** i reprezentuje książki z gatunku podręczników. Rozszerza funkcjonalność klasy **Ksiazka**, dodając dodatkowe cechy, takie jak przedmiot, poziom nauki czy dodatkowa cena.

## Pola:

- **private string id**  
Unikalny identyfikator książki, generowany automatycznie na podstawie numeru ISBN w formacie {ISBN}-PODR.
- **public EnumPrzedmiot przedmiot**  
Przedmiot, do którego odnosi się podręcznik (Matematyka, Fizyka, Informatyka, Chemia). Pole jest publiczne, co umożliwia łatwe filtrowanie i wyświetlanie informacji.
- **private decimal dodatekDoCeny**  
Stała opłata doliczana do ceny bazowej książki, wynosząca 40.
- **private EnumKlasa klasa**  
Poziom nauki, dla którego przeznaczony jest podręcznik (Podstawowka, Liceum, Studia). Pole oznacza, na jakim etapie edukacji książka może być używana.

## Konstruktor:

- **Podrecznik(string tytul, string autor, decimal cena, string dataPublikacji, int ilosc, string przedmiot, string klasa)**  
Konstruktor inicjalizuje obiekt podręcznika, przypisując odpowiednie wartości dla jego pól.
  - Generuje unikalny identyfikator książki (**id**) w formacie {ISBN}-PODR.
  - Przypisuje wartość **dodatekDoCeny** równą 40.
  - Sprawdza, czy podany rodzaj książki istnieje w wyliczeniu **EnumRodzaj**. Jeśli nie, rzuca wyjątek **NiepoprawnyRodzaj**. To samo robi dla **EnumKlasa**.

## Metody:

1. **ObliczCene()**  
Nadpisuje metodę z klasy bazowej, dodając do ceny bazowej wartość z pola **dodatekDoCeny**.
2. **ObliczCzasCzytania()**  
Nadpisuje metodę z klasy bazowej, zwiększając czas czytania 3-krotnie w porównaniu do standardowego czasu.
3. **ToString()**  
Nadpisuje metodę **ToString()**, zwracając szczegółowe informacje o książce, takie jak identyfikator, dane książki, przedmiot, poziom nauki oraz cena z dodatkiem.

#### 4. **Equals()**

Implementuje porównywanie obiektów klasy **Podrecznik**. Dwie książki są uznawane za identyczne, jeśli mają ten sam numer ISBN, przedmiot, poziom nauki, datę publikacji i autora.

#### 5. **Clone()**

Implementuje interfejs **ICloneable**, umożliwiając stworzenie płytkiej kopii obiektu za pomocą metody **MemberwiseClone()**.

#### **Uwagi dotyczące modyfikatorów dostępu:**

Pola **id**, **dodatekDoCeny** są oznaczone jako prywatne, aby kontrolować ich dostępność wyłącznie w obrębie klasy. Pole **przedmiot** jest publiczne, aby umożliwić łatwy dostęp do informacji w GUI oraz przy filtrowaniu i wyświetlaniu danych.

## Opis Testów

Testy jednostkowe zostały przygotowane, aby upewnić się, że wszystkie klasy w projekcie, takie jak **Ksiazka**, **Fantastyka**, **Kryminal**, **Romans**, **LiteraturaFaktu**, **Podrecznik** oraz **Magazyn**, działają poprawnie i zgodnie z założeniami. W trakcie testowania skupiono się na sprawdzeniu konstruktorów, metod, wyjątków oraz różnych funkcjonalności specyficznych dla każdej klasy.

#### **Główne cele testów:**

##### 1. **Sprawdzenie konstruktorów:**

Testy weryfikują, czy obiekty są poprawnie tworzone z prawidłowymi danymi. Dodatkowo, sprawdzono, czy w przypadku błędnych danych, takich jak niepoprawna ilość, cena lub format daty, rzucane są odpowiednie wyjątki.

##### 2. **Działanie metod:**

Przetestowano kluczowe metody, takie jak obliczanie ceny (**ObliczCene**), czasu czytania (**ObliczCzasCzytania**) oraz generowanie opisów książek (**ToString**), aby upewnić się, że działają zgodnie z założeniami.

##### 3. **Obsługa wyjątków:**

Zweryfikowano, czy program reaguje poprawnie na błędne dane, np. nieznaną gatunek książki, brak książki w magazynie czy próba operacji na zbyt małej ilości egzemplarzy.

##### 4. **Sortowanie i porównywanie:**

Testy sprawdzają działanie metod do sortowania książek według tytułów i cen, a także ich porównywanie w zależności od tytułów i dat publikacji.

##### 5. **Zapisywanie i odczytywanie danych:**

Przetestowano, czy magazyn można poprawnie zapisać do pliku XML oraz odczytać z niego dane, aby zapewnić integralność zapisywanych informacji.

## Szczegóły testów:

### 1. Klasa Książka:

- Sprawdzono konstruktor dla poprawnych i błędnych danych.
- Zweryfikowano, czy każdy obiekt **Książka** otrzymuje unikalny numer ISBN.
- Przetestowano metody **ObliczCene**, **ObliczCzasCzytania** oraz **ToString**.
- Przeprowadzono testy właściwości **Ilosc** i **CenaBazowa**, w tym przypadki graniczne (np. wartości równe 0).
- Sprawdzono działanie sortowania książek według tytułu i daty publikacji przy użyciu metody **CompareTo**.

### 2. Klasy dziedziczące (Fantastyka, Kryminal, Romans, LiteraturaFaktu, Podrecznik):

- Przetestowano, czy konstruktory poprawnie inicjalizują obiekty dla różnych rodzajów książek.
- Zweryfikowano, czy metody obliczające cenę uwzględniają dodatkowe opłaty specyficzne dla gatunku.
- Przeprowadzono testy metod **Equals**, **Clone** oraz **ToString**.
- Sprawdzono, czy czas czytania jest obliczany prawidłowo w zależności od specyfiki gatunku.

### 3. Klasa Magazyn:

- Zweryfikowano, czy magazyn jest poprawnie inicjalizowany.
- Przetestowano dodawanie książek do magazynu, w tym obsługę przypadków duplikatów (rzucenie wyjątku).
- Sprawdzono metody operujące na stanie magazynowym, takie jak **Dostawa**, **Sprzedaj** i **UsunKsiazke**.
- Przeprowadzono testy wyszukiwania książek według różnych kryteriów, np. autor, przedmiot, klasa czy rodzaj.
- Sprawdzono sortowanie książek w magazynie (po tytule, cenie rosnąco i malejąco).
- Zweryfikowano działanie serializacji i deserializacji magazynu do/z pliku XML.

# GUI

## App

Klasa App w projekcie Gui jest odpowiedzialna za podstawową konfigurację i zarządzanie aplikacją

## Heaven

Klasa **Heaven** pełni funkcję głównego okna nawigacyjnego aplikacji. Jest to centralny punkt interfejsu użytkownika, umożliwiający przechodzenie do innych modułów aplikacji, takich jak:

- **Magazyn** – zarządzanie stanem magazynowym.
- **Dostawa** – obsługa przyjmowania dostaw.
- **Sprzedaż** – sprzedaż książek
- **Wyszukiwanie** – umożliwia wyszukiwanie danych według kategorii
- **Hurtownia** – przejście do strony hurtowni internetowej
- **Wylogowanie** – powrót do głównego okna logowania.

Metody obsługi zdarzeń są publiczne, ponieważ muszą być dostępne z poziomu XAML w celu obsługi zdarzeń wywoływanych przez interfejs użytkownika.

### Metody:

Klasa zawiera szereg metod obsługujących kliknięcia przycisków w interfejsie. Każda z nich otwiera odpowiednie okno i zamyka bieżące:

- **Sprzedane\_Click** – Otwiera okno pozwalające na sprzedanie książki (**Sprzedane**).
- **Magazyn\_Click** – Otwiera okno zarządzania magazynem (**MagazynWindow**).
- **Dostawa\_Click** – Otwiera okno obsługi dostaw (**Dostawa**).
- **Wyszukaj\_Click** – Otwiera okno wyszukiwania danych (**Wyszukaj**).
- **Wyloguj\_Click** – Przenosi użytkownika do ekranu logowania (**MainWindow**).
- **Strona\_Click** – Otwiera sekcję dotyczącą hurtowni (**Hurtownia**).

# Hurtownia

Klasa **Hurtownia** odpowiada za integrację z przeglądarką WebView2, pozwalając użytkownikowi na otwarcie zewnętrznej strony internetowej hurtowni książek. Dzięki temu użytkownik może szybko przejść do witryny <https://hurtksiazki.pl/> bez opuszczania aplikacji.

Metoda **Wstecz\_Click** jest publiczna, ponieważ obsługuje zdarzenie wywoływane z poziomu XAML (kliknięcie przycisku).

## Metody:

### 1. Konstruktor Hurtownia:

- Odpowiada za inicjalizację komponentów okna i konfigurację WebView2.
- Obsługuje błędy inicjalizacji przeglądarki, informując użytkownika w przypadku problemów.
- Po poprawnej inicjalizacji ładuje stronę <https://hurtksiazki.pl/>.

### 2. Wstecz\_Click:

- Umożliwia powrót do głównego okna nawigacyjnego aplikacji (**Heaven**).
- Otwiera nowe okno **Heaven** i zamyka aktualne okno.

# Magazyn Window

Klasa **MagazynWindow** jest jednym z głównych okien w aplikacji, służącym do zarządzania magazynem książek. Tutaj użytkownik może dodawać nowe książki, usuwać je, filtrować, sortować, sprawdzać wartość magazynu, a nawet drukować dane o książkach. Klasa łączy funkcjonalności z klasy **Magazyn** z interfejsem graficznym.

Pole magazyn jest prywatne, ponieważ przechowuje dane magazynu książek i działa wewnątrz klasy, aby chronić je przed przypadkowymi modyfikacjami spoza klasy.

Wszystkie metody obsługujące zdarzenia (np. kliknięcia przycisków) są publiczne, ponieważ muszą być wywoływane przez interfejs graficzny (XAML).

## Metody:

### 1. Konstruktor:

- Ładuje dane magazynu z pliku magazyn.xml i ustawia widok listy książek.
- Inicjalizuje rozwijane listy (ComboBox) dla różnych typów książek, takich jak rodzaj fantastyki czy klasa podręcznika.

## 2. DodajKsiazke\_Click:

- Obsługuje dodawanie nowej książki do magazynu.
- Sprawdza poprawność danych wprowadzonych przez użytkownika (np. tytuł, autor, cena).
- Tworzy nową książkę odpowiedniego typu (np. Kryminał, Romans) i dodaje ją do magazynu.
- Zapisuje zmiany w magazynie do pliku XML.

## 3. WartoscMagazynu\_Click:

- Wyświetla łączną wartość książek w magazynie, korzystając z metody **ObliczWartosc** klasy **Magazyn**.

## 4. Drukuj\_Click:

- Pozwala użytkownikowi wydrukować informacje o wszystkich książkach przechowywanych w magazynie.

## 5. Wstecz\_Click:

- Zamienia aktualne okno na główne okno nawigacyjne Heaven.

## 6. Usun\_Click:

- Usuwa wybraną książkę z magazynu na podstawie jej numeru ISBN.
- Aktualizuje widok listy książek po usunięciu.

## 7. Sortuj\_SelectionChanged:

- Umożliwia sortowanie książek według wybranego kryterium (np. cena rosnąco, cena malejąco, tytuł alfabetycznie).

## 8. DodajRodzajKsiazki\_SelectionChanged:

- Dynamicznie zmienia widoczne pola w formularzu w zależności od wybranego typu książki (np. fantastyka, kryminał, podręcznik).

## 9. RodzajKsiazki\_SelectionChanged:

- Filtruje książki na podstawie wybranego rodzaju (np. tylko kryminały, podręczniki lub wszystkie książki).



# Main Window

Klasa **MainWindow** pełni funkcję ekranu logowania w aplikacji. Umożliwia użytkownikowi wprowadzenie nazwy użytkownika oraz hasła w celu uzyskania dostępu do głównej części aplikacji. Klasa zapewnia również odtwarzanie dźwięków w zależności od wyniku próby logowania.

Pole **correctUsername** przechowuje poprawną nazwę użytkownika potrzebną do zalogowania. Jest polem prywatnym, ponieważ dane uwierzytelniające nie powinny być modyfikowane ani widoczne z zewnątrz.

Pole **correctPassword** przechowuje poprawne hasło. Analogicznie do nazwy użytkownika, pole to jest prywatne dla zachowania bezpieczeństwa danych.

## Metody:

- **PlayLoginSound()** - Odtwarza plik dźwiękowy po wykonaniu próby logowania.
- **Button\_Click()** - Obsługuje zdarzenie kliknięcia przycisku logowania.

# Sprzedane

Klasa **Sprzedane** jest jednym z okien aplikacji, które umożliwia użytkownikowi zarządzanie procesem sprzedaży książek z magazynu. Pozwala na wybór książki, określenie liczby egzemplarzy do sprzedaży oraz aktualizację stanu magazynu. Użytkownik ma również możliwość wyszukiwania książek na podstawie tytułu.

## 1. Konstruktor Sprzedane()

- Inicjalizuje okno i ładuje dane magazynu z pliku XML.
  - Próbuje odczytać dane z pliku magazyn.xml.
  - Jeśli magazyn zostanie poprawnie załadowany, ustawia źródło danych dla kontrolki listy książek.
  - Obsługuje sytuacje błędne, wyświetlając komunikat o pustym magazynie lub błędzie podczas ładowania.

## 2. Sprzedaj\_Click()

- Obsługuje proces sprzedaży wybranej książki.
  - Pobiera wybraną książkę z listy.
  - Weryfikuje poprawność liczby egzemplarzy do sprzedaży (liczba większa od 0).
  - Wywołuje metodę Sprzedaj z klasy **Magazyn**, aby zmniejszyć liczbę egzemplarzy w magazynie.
  - Aktualizuje listę książek i zapisuje zmiany w pliku XML.
  - Wyświetla komunikat o udanej sprzedaży.

## 3. Wstecz\_Click()

Powraca do głównego okna nawigacyjnego aplikacji (**Heaven**). Tworzy nowe okno **Heaven**, zamyka bieżące okno.

#### 4. Szukaj\_Click()

Umożliwia wyszukiwanie książek w magazynie na podstawie tytułu.

- Pobiera tekst wprowadzony przez użytkownika.
- Wykonuje wyszukiwanie książek metodą **WyszukajPoTytule()** z klasy **Magazyn**.
- Aktualizuje listę wyświetlanych książek w kontrolce.

#### 5. lista\_SelectionChanged()

Obsługuje zdarzenie zmiany wyboru w liście książek.

Pole **magazyn** jest publiczne, aby umożliwić operacje na danych magazynu, takie jak sprzedaż i wyszukiwanie książek. Metody obsługujące zdarzenia (np. **Sprzedaj\_Click**, **Szukaj\_Click**) są publiczne, ponieważ muszą być wywoływane przez interfejs graficzny (XAML).

## Wyszukaj

Klasa **Wyszukaj** służy do przeszukiwania książek w magazynie według różnych kryteriów, takich jak autor, przedmiot, rodzaj książki czy poziom edukacyjny (klasa). Dzięki tej klasie użytkownik może w łatwy sposób filtrować i wyświetlać interesujące go dane z magazynu.

#### Metody:

##### 1. Konstruktor Wyszukaj()

Odpowiada za inicjalizację okna i załadowanie danych magazynu.

- Wczytuje dane magazynu z pliku magazyn.xml za pomocą metody **OdczytajDCXML**.
- W przypadku pomyślnego wczytania ustawia dane jako źródło dla listy książek (**listaKsiazek**).
- Obsługuje błędy związane z niepoprawnym załadowaniem magazynu, wyświetlając komunikat.

##### 2. Wstecz\_Click()

Umożliwia powrót do głównego okna nawigacyjnego aplikacji (**Heaven**). Tworzy nowe okno **Heaven** i zamyka bieżące okno.

### 3. Szukaj\_Click()

Umożliwia wyszukiwanie książek według nazwiska autora.

- Pobiera nazwisko autora z kontrolki tekstowej.
- Wywołuje metodę **WyszukajPoAutorze** z klasy **Magazyn**, aby znaleźć książki pasujące do wprowadzonego nazwiska.
- Wyświetla wyniki w kontrolce listy książek (**listaKsiazek**).
- Informuje użytkownika, jeśli nie wprowadzono nazwiska autora.

### 4. Rodzaj\_SelectionChanged()

Filtrowanie książek na podstawie wybranego rodzaju (np. Kryminał, Fantastyka).

- Pobiera wybraną opcję z kontrolki ComboBox i wyszukuje książki za pomocą metody **WyszukajPoRodzaju**.
- Aktualizuje listę książek na podstawie wyników.

### 5. Klasa\_SelectionChanged()

Filtrowanie książek na podstawie poziomu edukacyjnego (np. Podstawówka, Liceum).

- Pobiera wybrany poziom z ComboBox i konwertuje go na wartość **EnumKlasa**.
- Wywołuje metodę **WyszukajPoKlasie**, aby znaleźć książki z określonym poziomem edukacyjnym.
- Aktualizuje listę książek na podstawie wyników.

### 6. Przedmiot\_SelectionChanged()

Filtrowanie książek podręczników na podstawie przedmiotu (np. Matematyka, Fizyka).

- Pobiera wybraną opcję z ComboBox i konwertuje ją na wartość **EnumPrzedmiot**.
- Wywołuje metodę **WyszukajPoPrzedmiocie** z klasy **Magazyn**, aby znaleźć podręczniki dla danego przedmiotu.
- Aktualizuje listę książek na podstawie wyników.

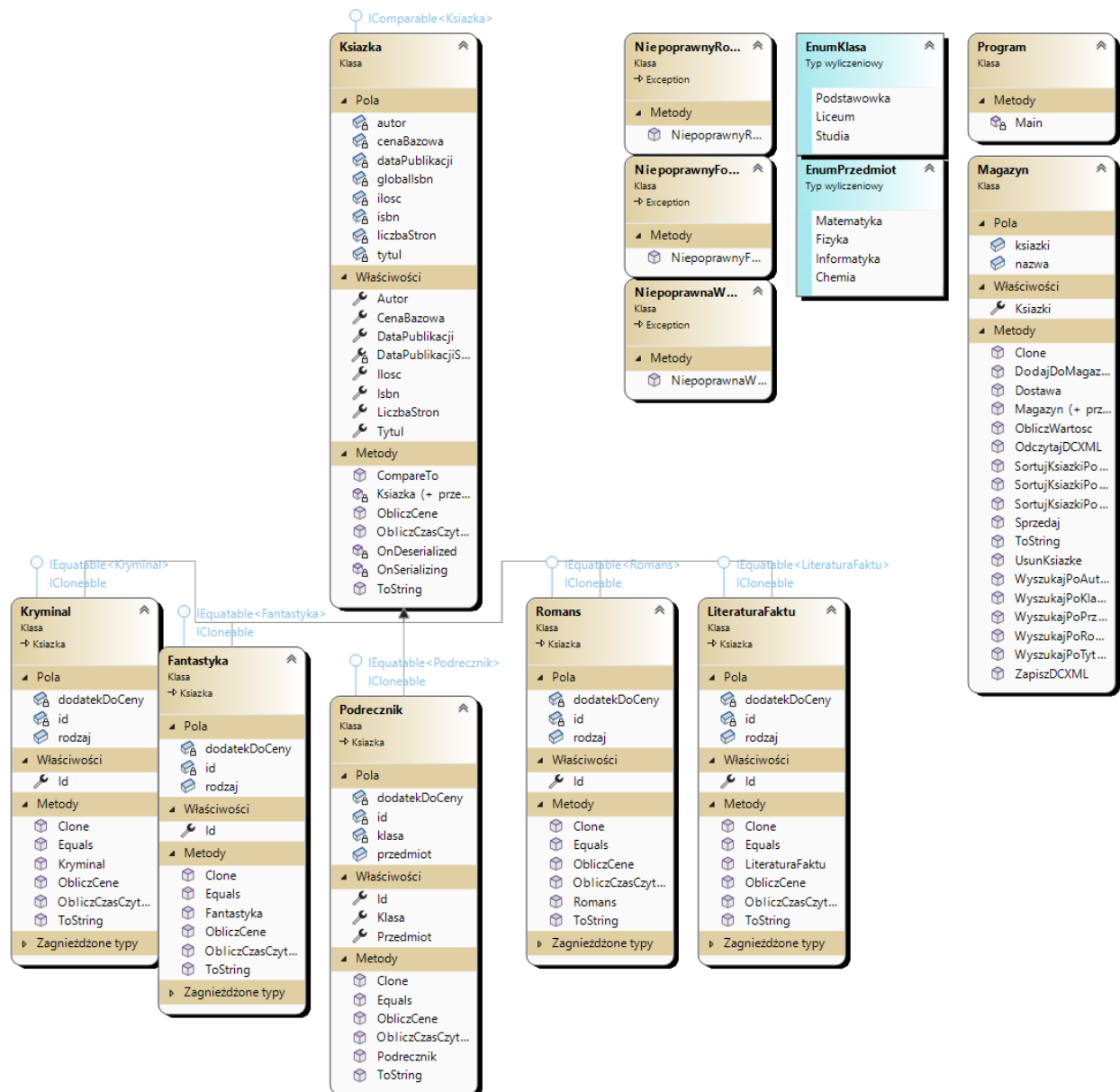
### 7. Autor\_TextChanged()

Obsługuje zdarzenie zmiany tekstu w polu autora.

#### Uzasadnienie modyfikatorów dostępu:

Pole **magazyn** jest prywatne, aby dane magazynu były bezpieczne i dostępne tylko w ramach tej klasy. Metody obsługujące zdarzenia są publiczne, ponieważ muszą być dostępne z poziomu interfejsu graficznego (XAML).

## Diagram klas



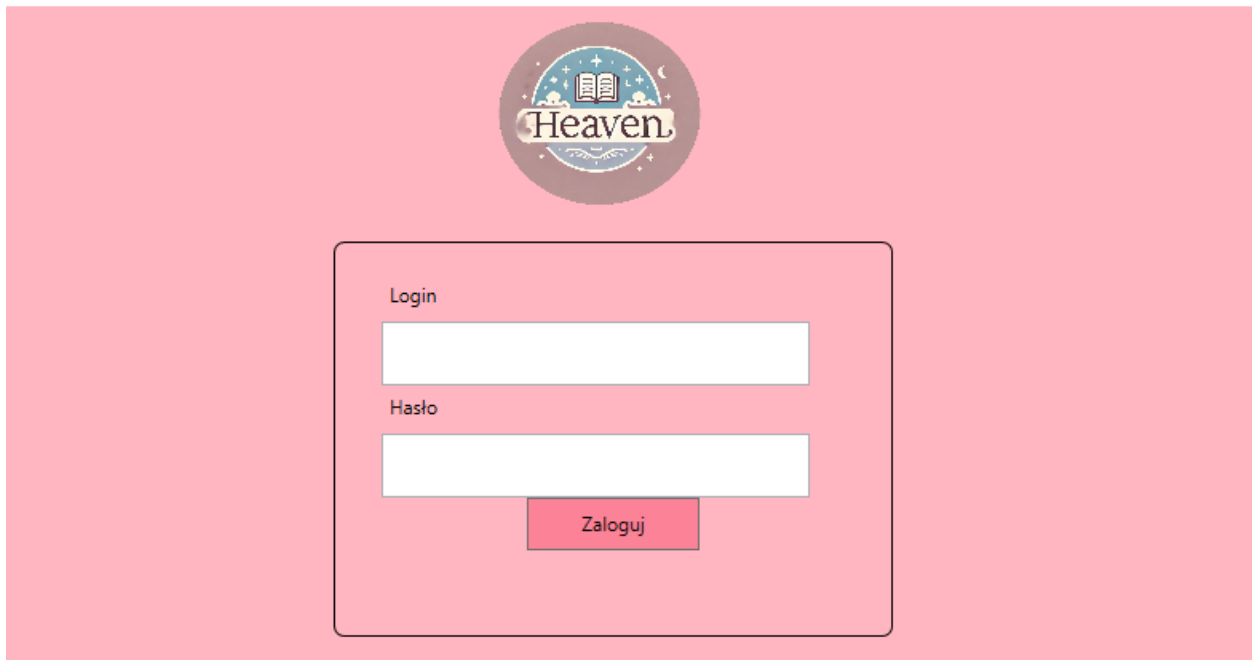
## Opis funkcjonalności

### 1. Logowanie do aplikacji

Po otwarciu aplikacji pojawia się ekran logowania, gdzie należy wprowadzić:

- **Login:** heaven
- **Hasło:** 123

Po poprawnym zalogowaniu usłyszymy dźwięk potwierdzający zalogowanie. Jeśli dane są niepoprawne, pojawi się dźwięk i komunikat o błędnym logowaniu.

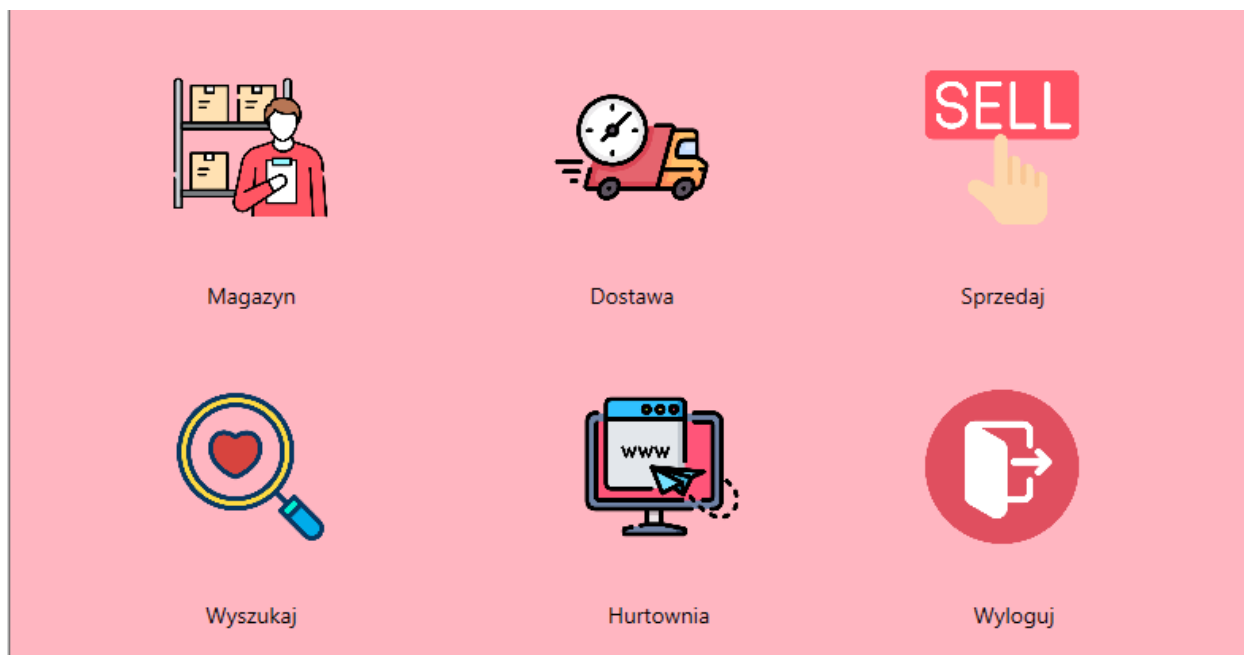


### 2. Główne menu aplikacji

Po zalogowaniu wyświetla się ekran główny z sześcioma ikonami:

1. **Magazyn**
2. **Dostawa**
3. **Sprzedaj**
4. **Wyszukaj**
5. **Hurtownia**
6. **Wyloguj**

Każda ikona prowadzi do odpowiedniego modułu aplikacji.



### 3. Magazyn

Po kliknięciu ikony **Magazyn**, przechodzimy do sekcji zarządzania książkami w magazynie. W tej zakładce możemy:

- **Filtrować książki:** Wyświetlić książki według ich rodzaju (np. Fantastyka, Romans, Kryminał).
- **Sortować książki:** Możliwość sortowania według:
  - Tytułu (alfabetycznie),
  - Ceny rosnąco,
  - Ceny malejąco.
- **Dodawać nową książkę do magazynu:**
  - Wypełnij formularz z informacjami o książce (tytuł, autor, rodzaj itp.).
  - Pasek ładowania wskazuje postęp dodawania książki.
- **Usuwać książki:** Wybierz książkę z listy i kliknij "Usuń".
- **Wyświetlić wartość magazynu:** Kliknij "Wartość magazynu", aby zobaczyć łączną wartość wszystkich książek.
- **Drukować magazyn:** Wydrukuj listę książek znajdujących się w magazynie.

Tytuł	Autor	DataPublikacji	Isbn
Współcześnie - Romeo i Julia	Anna Pałka	1/15/2023 12:00:00 AM	12345
Duma i Pycha	Jane Hapawe	6/1/2024 12:00:00 AM	12345
Kocham Cię	Martyna Kulig	8/15/2025 12:00:00 AM	12345
The Great Gatsby	F. Scott Fitzgerald	7/10/2023 12:00:00 AM	12345
Bridget Jones's Diary	Helen Fielding	9/10/2022 12:00:00 AM	12345
The Fault in Our Stars	John Green	12/1/2021 12:00:00 AM	12345
P.S. I Love You	Cecelia Ahern	2/14/2024 12:00:00 AM	12345
P&P	Cecelia Ahern	2/14/2024 12:00:00 AM	12345
Zbrodnia i Kara	Fiodor Dostojewski	1/1/2023 12:00:00 AM	12345
Milczenie owiec	Thomas Harris	1/1/2015 12:00:00 AM	12345
Obama Trail	Stieg Larsson	11/1/2020 12:00:00 AM	12345
Sherlock Holmes: A Study in Scarlet	Arthur Conan Doyle	10/20/2022 12:00:00 AM	12345
Gone Girl	Gillian Flynn	3/25/2019 12:00:00 AM	12345

Filtr

Rodzaj książki

Sortowanie

Sortuj

Operacje

Usuń

Wartość magazynu

Drukuj

Dodawanie książki

Wybierz gatunek...

Wpisz tytuł...

Wpisz autora...

Wpisz ilość...

Wpisz datę wydania...

Wpisz cenę bazową...


Wpisz liczbę stron...

Dodaj książkę

#### 4. Dostawa

W zakładce **Dostawa** możesz dodać nowe egzemplarze książek do magazynu:

1. Wprowadź tytuł książki, którą chcesz wyszukać, w polu wyszukiwania.
2. Kliknij **Szukaj**, aby znaleźć książkę w magazynie.
3. Zaznacz wybraną książkę z listy wyników.
4. Wprowadź liczbę egzemplarzy, które chcesz dodać do magazynu.
5. Kliknij **Dostawa**, aby zaktualizować stan magazynu.



Tytuł	Autor	DataPublikacji	Ilość
Współcześnie - Romeo i Julia	Anna Pałka	1/15/2023 12:00:00 AM	123 ^
Duma i Pycha	Jane Hapawe	6/1/2024 12:00:00 AM	123
Kocham Cię	Martyna Kulig	8/15/2025 12:00:00 AM	123
The Great Gatsby	F. Scott Fitzgerald	7/10/2023 12:00:00 AM	123
Bridget Jones's Diary	Helen Fielding	9/10/2022 12:00:00 AM	123
The Fault in Our Stars	John Green	12/1/2021 12:00:00 AM	123
P.S. I Love You	Cecelia Ahern	2/14/2024 12:00:00 AM	123
P&P	Cecelia Ahern	2/14/2024 12:00:00 AM	123
Zbrodnia i Kara	Fiodor Dostojewski	1/1/2023 12:00:00 AM	123
Milczenie owiec	Thomas Harris	1/1/2015 12:00:00 AM	123
Obama Trail	Stieg Larsson	11/1/2020 12:00:00 AM	123
Charles Holmes & Studio 54	Arthur Conan Doyle	10/20/2023 12:00:00 AM	123

#### 5. Sprzedaż

W zakładce **Sprzedaż** możesz sprzedać książki z magazynu:

1. Wprowadź tytuł książki.
2. Podaj liczbę egzemplarzy, które chcesz sprzedać.
3. Kliknij "Sprzedaj", aby zaktualizować stan magazynu.



Wprowadź tytuł

Szukaj

Wprowadź ilość

Sprzedaj

Tytuł	Autor	DataPublikacji	Isbn
Współcześnie - Romeo i Julia	Anna Pałka	1/15/2023 12:00:00 AM	123 ^
Duma i Pycha	Jane Hapawe	6/1/2024 12:00:00 AM	123
Kocham Cię	Martyna Kulig	8/15/2025 12:00:00 AM	123
The Great Gatsby	F. Scott Fitzgerald	7/10/2023 12:00:00 AM	123
Bridget Jones's Diary	Helen Fielding	9/10/2022 12:00:00 AM	123
The Fault in Our Stars	John Green	12/1/2021 12:00:00 AM	123
P.S. I Love You	Cecelia Ahern	2/14/2024 12:00:00 AM	123
P&P	Cecelia Ahern	2/14/2024 12:00:00 AM	123
Zbrodnia i Kara	Fiodor Dostojewski	1/1/2023 12:00:00 AM	123
Milczenie owiec	Thomas Harris	1/1/2015 12:00:00 AM	123
Obama Trail	Stieg Larsson	11/1/2020 12:00:00 AM	123
Shakespeare's A Study in Scarlet	Arthur Conan Doyle	10/20/2023 12:00:00 AM	123 v

## 6. Wyszukaj

Zakładka **Wyszukaj** umożliwia wyszukiwanie książek według różnych kryteriów:

- **Rodzaj książki**
- **Przedmiot**
- **Klasa edukacyjna**
- **Autor**

Po wybraniu kryterium wyniki zostaną wyświetlone w liście książek.

←

Tytuł	Autor	DataPublikacji	Isbn
Współcześnie - Romeo i Julia	Anna Pałka	1/15/2023 12:00:00 AM	1234
Duma i Pycha	Jane Hapawe	6/1/2024 12:00:00 AM	1234
Kocham Cię	Martyna Kulig	8/15/2025 12:00:00 AM	1234
The Great Gatsby	F. Scott Fitzgerald	7/10/2023 12:00:00 AM	1234
Bridget Jones's Diary	Helen Fielding	9/10/2022 12:00:00 AM	1234
The Fault in Our Stars	John Green	12/1/2021 12:00:00 AM	1234
P.S. I Love You	Cecelia Ahern	2/14/2024 12:00:00 AM	1234
P&P	Cecelia Ahern	2/14/2024 12:00:00 AM	1234
Zbrodnia i Kara	Fiodor Dostojewski	1/1/2023 12:00:00 AM	1234
Milczenie owiec	Thomas Harris	1/1/2015 12:00:00 AM	1234
Obama Trail	Stieg Larsson	11/1/2020 12:00:00 AM	1234
Sherlock Holmes: A Study in Scarlet	Arthur Conan Doyle	10/20/2022 12:00:00 AM	1234
Gone Girl	Gillian Flynn	3/25/2019 12:00:00 AM	1234
The Silent Patient	Alex Michaelides	1/15/2020 12:00:00 AM	1234
Big Little Lies	Liane Moriarty	6/12/2021 12:00:00 AM	1234

Wyszukaj po rodzaju

Romans i Kryminał

Wyszukaj po przedmiocie

Podreczniki

Wyszukaj po klasie

Podreczniki

Wyszukaj po autorze

Szukaj

Wszystkie


## 7. Hurtownia

Kliknięcie ikony **Hurtownia** przenosi użytkownika na stronę internetową hurtowni książek:

<https://hurtksiazki.pl/>.

←

Status zamówienia



**hurtksiążki.pl**

☰

🔍

🛒 0

👤



**Ponad 100 tys. artykułów  
w jednym miejscu!**

## 8. Wylogowanie

Kliknięcie ikony **Wyloguj** zamyka bieżące okno i przenosi użytkownika z powrotem na ekran logowania.

# Źródła grafik i dźwięków użytych w projekcie

Źródło: <https://www.flaticon.com/search?word=browser>

Autorzy:

Magazyn:

[Limited stock icons](https://www.flaticon.com/free-icons/limited-stock "limited stock icons") created by Backwoods - Flaticon

Wyloguj:

[Logout icons](https://www.flaticon.com/free-icons/logout "logout icons") created by SumberRejeki - Flaticon

Wyszukaj:

[Love and romance icons](https://www.flaticon.com/free-icons/love-and-romance "love and romance icons") created by Kiranshastry - Flaticon

Dostawa:

[Fast delivery icons](https://www.flaticon.com/free-icons/fast-delivery "fast delivery icons") created by Freepik - Flaticon

Wstecz:

[Back arrow icons](https://www.flaticon.com/free-icons/back-arrow "back arrow icons") created by chehuna - Flaticon

Sprzedaj

[Sell icons](https://www.flaticon.com/free-icons/sell "sell icons") created by Smashicons - Flaticon

Strona internetowa:

[Website icons](https://www.flaticon.com/free-icons/website "website icons") created by Freepik - Flaticon

Dźwięki:

<https://pixabay.com/pl/sound-effects/search/powiadomienie/>

Logo wygenerowane za pomocą Sztucznej Inteligencji