Dokumentacja - System Zarządzania Biblioteką

Eryk Zarębski
Wydział Fizyki i Informatyki Stosowanej AGH

25 stycznia 2023

1 Koncepcja

1.1 Temat projektu

Celem projektu było stworzenie podstawowego interfejsu pozwalającego zarządzać biblioteką, tzn. przechowywać informacje na temat książek, pracowników biblioteki, członków oraz czynności tj. wypożyczenie książki czy nałożenie opłaty oraz modyfikować je tak by odzwierciedlały świat rzeczywisty.

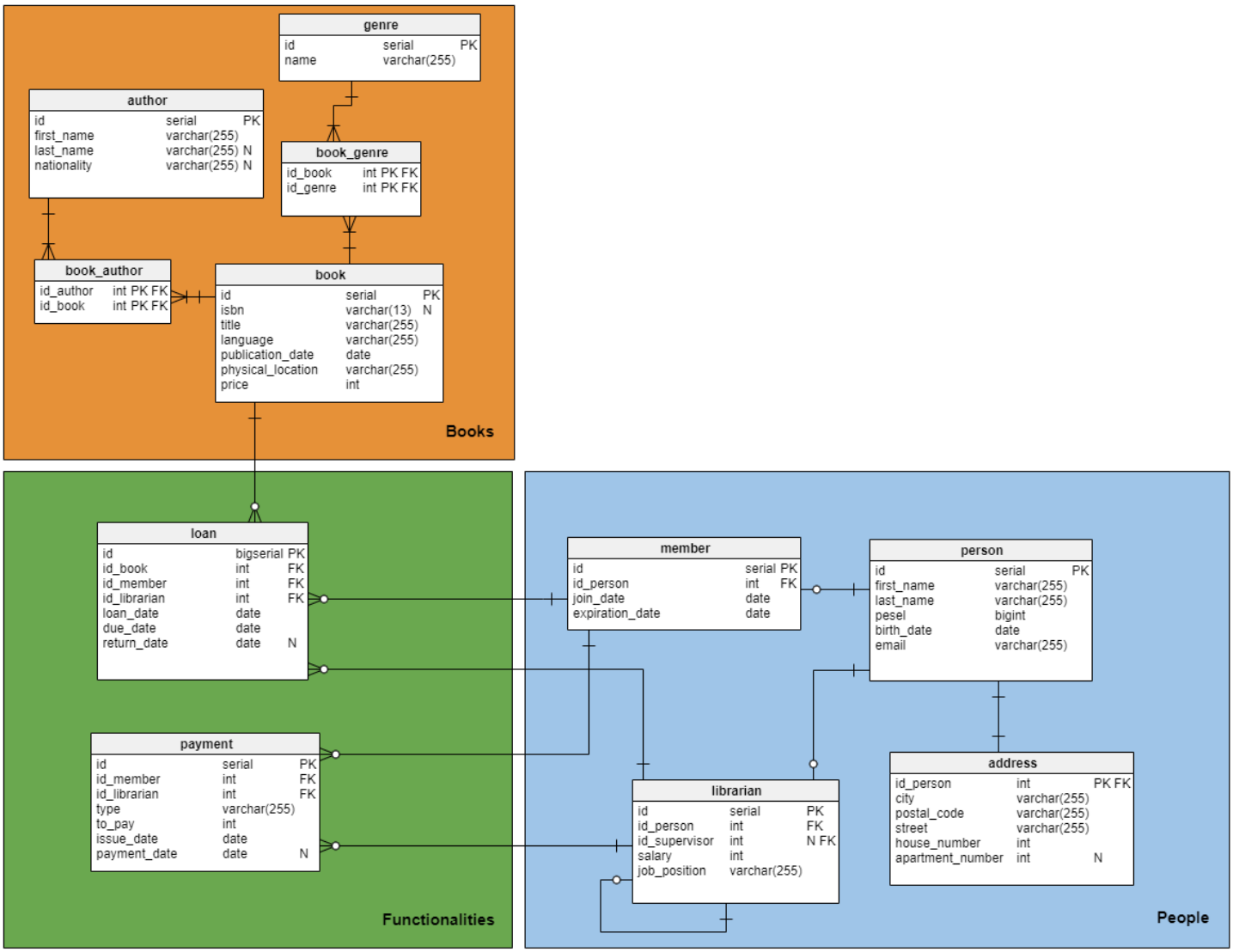
1.2 Funkcje

Funkcjami, które uznałem istnymi do zaimplementowania zostały:

- Możliwość dodania nowego członka do bazy danych
- Możliwość dodania nowego pracownika „
- Możliwość dodania książki „
- Możliwość dodania autora „
- Możliwość dodania gatunku „
- Możliwość wyszukania książki oraz sprawdzenia czy jest dostępna
- Możliwość wypożyczenia książki jeśli jest dostępna, a członkostwo wypożyczającego jest ważne i nie ma on żadnych opłat do zapłacenia
- Możliwość zwrócenia książki i nałożenia opłaty jeśli do zwrotu doszło po terminie
- Możliwość uiszczenia wybranej prze członka opłaty
- Możliwość obejrzenia listy członków, autorów, pracowników, wypożyczeń, opłat, książek oraz ich autorów i gatunków

2 Projekt bazy danych

2.1 Diagram ERD



Rysunek 1: Diagram ERD

Wyżej przedstawiony diagram prezentuje jak wyglądają tabele w bazie danych oraz relacje między nimi. Są na nim umieszczone jedynie tabele rzeczywiste, bez widoków.

2.2 Omówienie diagramu

Diagram pozwoliłem sobie podzielić semantycznie na kilka części.

Część *People* reprezentuje ludzi będących częścią biblioteki - jej członkami lub pracownikami. W tabeli *person* znajdują się podstawowe dane osobowe, tj. imię, nazwisko, czy pesel. Na pesel nałożone jest ograniczenie unikalności oraz długości dokładnie 11 cyfr. Tabela *address* reprezentuje adres zamieszkania osoby - powiązana jest relacją 1-1 z *person*. *member* reprezentuje członkostwo w bibliotece - każda osoba może posiadać jedno lub zero. Członkostwo ma swoją datę rozpoczęcia oraz datę wygaśnięcia, po dacie wygaśnięcia można je przedłużyć za opłatą o np. rok licząc od daty dzisiejszej (nie zaimplementowane). *librarian* przedstawia atrybuty pracownika tj. jego przełożonego, pozycję oraz wypłatę. Przełożony jest referencjonowany przez *id_ supervisor*, jeśli wartość ta jest null'em to znaczy, że pracownik nie ma przełożonego.

Część *Books* reprezentuje książki. *book* zawiera atrybuty opisujące książkę w bibliotece. Każda książka, nawet z tym samym *isbn* jest rozróżnialna, mogą więc istnieć "te same"pozycje pod różnymi *id*, sam ISBN ponadto nie jest prawnie wymagany, jest więc null'owalny. *physical_ location* wskazuje na fizyczne położenie w bibliotece gdzie powinna znajdować się książka. *price* to cena katalogowa, która powinna stanowić podstawę do nałożenia opłaty za potencjalne zniszczenie książki. Każda książka może przynależeć do jednego lub wielu gatunków oraz mieć jednego bądź więcej autorów - stan ten odzwierciedlają tabele *genre* i *author* wraz z ich encjami asocjacyjnymi *book_ author* orac *book_ genre*.

Część *Functionalities* reprezentuje czynności, których mogą dokonać osoby. *loan* reprezentuje wypożyczenie książki o określonym *id* przez jednego z bibliotekarzy określonemu członkowi. *loan_ date* jest datą wypożycznia, *due_ date* datą do której można książkę oddać bez ponoszenia kosztów, *return_ date* jest datą oddania - jeśli jest ona nullem to znaczy, że książka nie została jeszcze zwrócona, a w przypadku gdy jest większa od *due_ date* to powinna zostać nałożona opłata zależna od ilości dni opóźnienia. *payment* reprezentuję płatność nałożoną na członka przez bibliotekarza lub automatycznie - w przypadku automatycznym referencjonowane jest *id_ librarian* równe 1. Jeśli *payment_ date* jest null'em to znaczy, że opłata nie została uiszczona i członek nie może korzystać z biblioteki.

3 Projekt logiczny

3.1 DDL

Definicje SQL wszystkich tabel, kluczy, widoków i triggerów znajdują się w folderze `src/sql` projektu, są one ponumerowane w kolejności, w której powinny być uruchomione w bazie danych.

3.2 Normalizacja

Większość tabel jest znormalizowana do formy 3NF, jedyna tabela do której mogę mieć zastrzeżenia to *address* gdyż *city* wynika teoretycznie z *postal_ code*, ale mam wątpliwości co do praktyczności dalszej normalizacji.

3.3 Widoki, triggery

W ramach ułatwienia pracy z bazą danych zaprojektowałem kilka widoków i trigerrów. Spośród widoków można wyróżnić:

- *member_ all_ data* - zawiera wszystkie dane członka wraz z danymi osobowymi i adresem
- *librarian_ all_ data* - zawiera wszystkie dane pracownika „
- *authors_ of_ book* - zawiera dane autorów książek o określonych id
- *genres_ of_ book* - zawiara nazwy gatunków „
- *lent_ book* - zawiera aktualnie wypożyczone książki
- *book_ with_ loan_ status* - zawiera dane książki wraz z aktualnym statusem wypożyczenia „
- *member_ with_ fee* - zawiera członków posiadających nie uiszczone opłaty

Z triggerów natomiast:

- *delayed_ book_ return_ trigger* - odpowiada za nałożenie na członka opłaty za opóźnienie w oddaniu książki
- *impose_ membership_ payment_ trigger* - nakłada na członka opłatę za rejestrację

4 Projekt funkcjonalny

Aplikacja jest stroną internetową z prostym, nie wymagającym wyjaśniania interfejsem, choć jest trochę toporna, gdyż wymaga czasami zapamiętania kilku id wymaganych w formularzu - w sytuacji realnej byłoby nieco łatwiej, bo można by np. przeczytać id prosto z okładki książki, a pracownik swoje znałby na pamięć.

5 Obsługa aplikacji

5.1 Uruchomienie

Aby uruchomić aplikację w środowisku linux należy posiadać zainstalowany lokalnie język Python oraz wywołać następujące komendy z folderu głównego aplikacji:

- `python -m venv env`
- `source env/bin/activate`
- `pip install -r src/requirements.txt`
- `python main.py`

Pierwsze dwie komendy utworzą wirtualne środowisko dzięki, któremu odpowiednie paczki oprogramowania nie będą instalować się globalnie, kolejna komenda zainstaluje wskazane paczki oprogramowania, a ostatnia uruchomi aplikację, która będzie działać domyślnie pod adresem `http://localhost:5000`.

5.2 Struktura katalogów

W pliku `main.py` znajduje się główna część kodu w której zdefiniowane endpointy odpowiadające każdej zaimplementowanej funkcjonalności. Wewnątrz folderu `src` znajdują się takie foldery jak

- `templates` - folder zawierający szablony stron zapisane w języku silnika templatkowego używanego domyślnie przez Flask'a - Jinja;
- `static` - folder z plikami statycznymi tj. pliki css;
- `sql` - folder zawierający pliki z definicjami tabel itp. oraz folder `mock` zawierający przykładowe dane wygenerowane głównie przy użyciu serwisu `www.mockaroo.com`, skrypty tam umieszczone są ponumerowane zgodnie z kolejnością uruchamiania w bazie danych
- `forms` - folder zawierający formularze zdefiniowane przy użyciu biblioteki WTForms
- `models` - folder zawierający klasy z metodami odpowiedzialnymi za komunikację z bazą danych. W początkowym etapie projektu myślałem, że zrobię coś w stylu ORM - wtedy nazwa `models` miałaby nieco właściwsze znaczenie, tutaj natomiast "model"jest raczej zbieraniną podobnych semantycznie metod - analogicznie do diagramu ERD