Dokumentacja

Spis treści:

1. Spis użytych technologii
2. Lista plików i opis ich zawartości
3. Kolejność i sposób uruchamiania plików
4. Schemat projektu bazy danych
5. Zależności funkcyjne
6. EKNF
7. Najtrudniejsze podczas realizacji projektu
8. Spis użytych technologii
   1. Python w wersji 3.11.4.
   2. Biblioteki w pythonie, między innymi sqlalchemy. Bardziej szczegółowy spis użytych bibliotek w pliku requirements.txt
   3. Baza danych MariaDB, na serwerze zajęciowym
   4. ERD editor dla wizualizacji zależności między tabelami
9. Lista plików i opis ich zawartości

Pimp my wheels

|

| - data (folder z danymi dla projektu)

| \

| | - parameters (folder ze podstawowymi stałymi)

| | \

| | | - dates.json

| | |

| | | - employees.json

| | |

| | | - services\_parts.json

| |

| | - raw (folder z imionami i nazwiskami osób)

| | \

| | | - female\_first\_names.csv (imiona żeńskie)

| | |

| | | - female\_surnames.csv (nazwiska żeńskie)

| | |

| | | - male\_first\_names.csv (imiona męskie)

| | |

| | | - male\_surnames.csv (nazwiska męskie)

| |

| | - brands.csv (marki, modele i ceny samochodów)

| |

| | - equipment.csv(nazwy, typy i ceny rynkowe części samochodów)

| |

| | - female\_surnames.csv(????)

| |

| | - male\_surnames.csv(????)

| |

| | - names.csv(????)

|

| - src(folder dla podstawowej generacji bazy)

| \

| | - emulation (folder dla emulacji akcji podejmowanych przez osób)

| | \

| | | - \_\_init\_\_.py

| | |

| | | - customer\_decision\_maker.py (akcji podejmowane przez klientów)

| | |

| | | - emulation.py (emulacja jednego dnia pracy)

| | |

| | | - equipment\_generator.py (generator narzędzi)

| | |

| | | - workshop\_decision\_maker.py (akcji dotyczące napraw w warsztatu)

| | |

| | | - workshop\_emulator.py (akcji dotyczące sprzedaży/kupna warsztatu)

| |

| | - generators

| | \

| | | - \_\_init\_\_.py

| | |

| | | - brand\_geneerator.py (generator aut dla tabel)

| | |

| | | - equipment\_generator.py (generator narzędzi dla tabel)

| | |

| | | - personal\_data\_generator.py (generator danych personalnych)

| |

| | - models (folder architektur i zależności w bazie)

| | \

| | | - \_\_init\_\_.py

| | |

| | | - base.py (classa bazowa dla pozostałych tabel w sqlalchemhy)

| | |

| | | - customer.py (tabela klientów)

| | |

| | | - employee.py (tabela pracowników)

| | |

| | | - equipment.py (tabela katalogu narzędzi)

| | |

| | | - inventory.py (tabela urzytych narzędzi)

| | |

| | | - service.py (tabela napraw)

| | |

| | | - transaction.py (tabela transakcji bankowych)

| | |

| | | - vehicle.py (tabela kupionych i sprzedanych pojazdach)

| | |

| | | - workshop.py (tabela warsztatów)

| |

| | - person

| |

| | - utils

| | \

| | | - \_\_init\_\_.py

| | |

| | | - names.py (wczytywanie danych z data/raw)

| |

| | - \_\_init\_\_.py

|

| - venv (wirtualne środowisko pythona)

|

| - Dokumentacja.docx (dokumentacja)

|

| - main.py (plik generujący całą bazę)

|

| - PimpMyWheelsSchema.vuerd.json (wizualizacja zależności i tabel w bazie)

|

| - requirements.txt (spis użytych bibliotek)

1. Kolejność i sposób uruchamiania plików
   1. Na początek należy zainstalować python w wersji 3.11.4. Można to

zrobić na oficjalnej stronie:

<https://www.python.org/downloads/release/python-3114/>

Wraz z pythonem instaluje się pip (administrator bibliotek dla pythona). Można sprawdzić czy się zainstalował za pomocą wpisania polecenia:

pip list

w cmd. Jeżeli pojawia się lista zaistalowanych bibliotek, to wszystko działa, jeżeli nie, można zainstalować administrator anaconda:

<https://www.anaconda.com/download>

3.2 Należy zainstalować wszystkie biblioteki dla projektu. Można to łatwo zrobić za pomocą komendy:

pip install -r requirements.txt

Znajdując się w głównym folderze PimpMyWheels

3.3 Instalujemy Visual Studio Code. Można to zrobić z oficjalnej strony:

<https://code.visualstudio.com/download>

3.4 W Visual Studio Code instalujemy ERD Editor.

3.5 Dalej można zapoznać się z ogólną architekturą bazy w pliku PimpMyWheelsSchema.vuerd.json.

3.6 Ostatnim krokiem wchodzimy do pliku Raport.Rmd i generujemy pdf raporta za pomocą Knittr

3.7 Teraz można otworzyć Raport.pdf

1. Schemat projektu bazy danych i zależności funkcyjne

Wizualizacja schematu projektu bazy danych jest w pliku: PimpMyWheelsSchema.vuerd.json

Poniżej przedstawimy opis każdej tabeli:

* 1. Customers

id(Int, unsigned, not null, autoincrement , primary key)

name(String(25))

surname(String(25))

email(String(60))

phone\_number(String(12))

birth\_date(Date)

address(String(200))

account\_creation\_date(Date)

account\_deletion\_date(Date)

last\_active(Date)

* 1. Employees
  2. Equipment
  3. Inventory
  4. Services
  5. Transactions
  6. Vehicles
  7. Workshops

1. EKNF
2. Najtrudniejsze podczas realizacji projektu