Wyniki etapu IV: Implementacja i testy

Polwita

Projektowanie oprogramowania

Skład zespołu: Prowadzący: mgr Urszula Staszak

Grzegorz Dzikowski

Implementacja

# Struktura kodu

Frontend:

Obraz zawierający tekst

Opis wygenerowany automatycznie

Backend:

Obraz zawierający tekst

Opis wygenerowany automatycznie

# Generacja kodu

Użyłem programu <https://pypi.org/project/sqlacodegen/> do wygenerowania klas ORM w backendzie Pythona. Program łączy się z bazą danych i generuje struktury tabel



Obraz zawierający tekst

Opis wygenerowany automatycznie

# Wzorce projektowe

# Dokumentacja kodu

Przy pomocy narzędzia <https://www.sphinx-doc.org/en/master/> wygenerowałem dokumentację do klasy bazodanowej

Obraz zawierający tekst

Opis wygenerowany automatycznie

Testowanie

# Testy systemowe – przypadki/procedury testowe

[Document also those test cases which were automated.]

## <Nazwa przypadku użycia>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TC ID | Idea testowa | Kroki testu | Dane testowe | Oczekiwane wyniki |
|  |  |  |  |  |

## <Nazwa przypadku użycia>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TC ID | Idea testowa | Kroki testu | Dane testowe | Oczekiwane wyniki |
|  |  |  |  |  |

…

# Testy systemowe automatyczne

[Document which test cases were automated, how (the name of the tool, a screen-shot). Option.]

# Testy jednostkowe

[A list of methods for which unit test exist. Samples of 1-2 test methods.]

# Ocena jakości kodu

[Info if any tool (which one) was used for assessment of source quality. The measures plus interpretation. Option.]