

Systemy telemedyczne w semestrze zimowym 21' - projekt

Temat nr 8: Model systemu IoT gromadzącego podstawowe dane o zdrowiu (temperatura, ciśnienie, tętno, SpO2) wykorzystujący bazę danych typu time series oraz prezentację danych w Web.

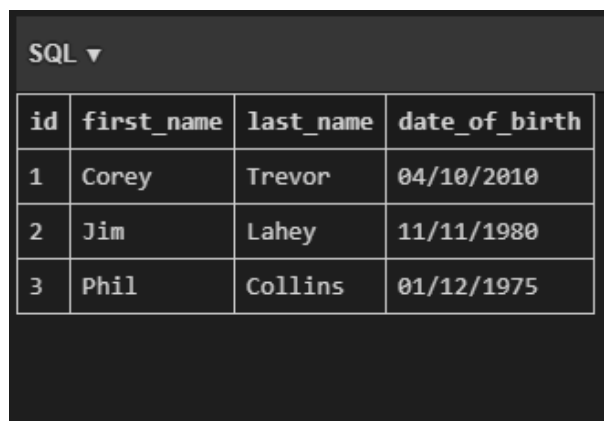
Skład zespołu H:

Kacper Kubicki

Opis projektu:

Projekt został wykonany w środowisku Python.

W pliku *create-tables.py* zdefiniowano sposób tworzenia pustej bazy danych SQL z wykorzystaniem biblioteki SQLite, składającej się z dwóch tabel – *patients* i *results*. Tabela *patients* (rys. 1) składa się z następujących kolumn: PRIMARY KEY (*patient_id*), imię, nazwisko i data urodzenia. Natomiast tabela *results* (rys. 2): id, temperatura, ciśnienie, SpO2, tętno, *patient_id*, timestamp. Funkcja *create_table()* odpowiada za tworzenie tabeli według powyższego pomysłu, a funkcja *insert_patient()* odpowiada za umieszczenie danych osobowych pacjentów w pierwszej tabeli *patients*. Wszystkie działania na bazie danych są wykonywane w oparciu o komendy zawarte w pliku *dbcommands.py*. W celu wyczyszczenia tabeli *results* z danych, należy uruchomić plik *clear-results-table.py*. W pliku *config.py* utworzona została ścieżka dostępu do bazy danych (DB_PATH).



SQL ▼			
id	first_name	last_name	date_of_birth
1	Corey	Trevor	04/10/2010
2	Jim	Lahey	11/11/1980
3	Phil	Collins	01/12/1975

Rysunek 1. Tabela „patients”

SQL ▾							
< 1 / 3 > 1 - 50 of 116							
id	temperature	pressure_systolic	pressure_diastolic	SpO2	heart_rate	patient_id	timestamp
1	36.720731504807205	123	80	61	91	1	1643145905
2	36.575291494491495	118	78	60	89	1	1643145965
3	36.95865144193874	119	83	59	91	1	1643146025
4	36.798573766365564	117	81	61	90	1	1643146085
5	36.47129164252471	125	81	60	89	1	1643146145
6	36.66503838809224	118	81	60	89	1	1643146205
7	36.73556120407759	120	82	60	91	1	1643146265
8	36.314367001873144	125	81	60	90	1	1643146325
9	36.77944586043073	116	81	60	90	1	1643146385
10	36.66085915638028	124	79	60	91	1	1643146445
11	36.63230234441849	121	79	59	91	1	1643146505
12	36.66501661011889	122	78	59	90	1	1643146565
13	36.671083964765245	125	79	60	90	1	1643146625
14	36.62588234408778	122	79	61	89	1	1643146685
15	36.6438957403415	119	82	60	90	1	1643146745
16	36.45004991312268	121	77	61	92	1	1643146805
17	36.643200836144295	118	79	61	90	1	1643146865
18	36.172262526021164	120	84	62	89	1	1643146925
19	36.43041638950308	124	79	60	89	1	1643146985
20	36.439094077784624	117	80	60	91	1	1643147045
21	36.63685304952942	120	80	59	89	1	1643147105

Rysunek 2. Tabela „results”

W pliku *emulator.py* utworzono funkcję *create_random_data()*, która odpowiada za generowanie parametrów życiowych (temperatura, ciśnienie, tętno, SpO2). Zmiana wartości parametrów zależna jest od losowej wartości z przedziału $<0;0,15>$ oraz wartości współczynnika przypisanego do danego parametru “na sztywno”. Dane o parametrach zostały wprowadzone do tabeli *results* w bazie danych SQL poprzez funkcję *insert_result()*. Rekordy w tabeli *results* są przypisane do konkretnego pacjenta w zależności od klucza *patient_id*. Z którego każdy posiada dodatkową informację o czasie badania (timestamp) podany w sekundach.

W pliku *server.py* utworzone zostały funkcje *get_patients()* oraz *get_results()*. Funkcja *get_patients()* służy do sczytywania danych osobowych pacjentów, natomiast *get_results()* służy do sczytywania parametrów. Dane z obu funkcji żądanie GET wysyła na serwer. Dane o parametrach zostały podzielone na 5 zmiennych (np. YTemperature), z których każda zawiera informacje o czasie (godzina i minuta) oraz o wartości emulowanego parametru. Utworzone zostały również wykresy przy wykorzystaniu biblioteki *plotly*. W pliku *config.py* utworzona została ścieżka do utworzonych wykresów (FILES_PATH).

Za działanie aplikacji Webowej odpowiedzialny jest plik *index.html*. Po wybraniu pacjenta (predefiniowanego w funkcji *insert_patients()* w pliku *create-tables.py*) z rozwijającej się listy (rys. 3) pojawiają się informacje takie jak imię, nazwisko, data urodzenia oraz wykresy zmiany badanych parametrów (temperatura, ciśnienie, tętno, SpO2) w funkcji czasu (rys. 4).

-- select a patient --

-- select a patient --
Corey Trevor
Jim Lahey
Phil Collins

Get results

Rysunek 3. Rozwijana lista wyboru pacjenta



Rysunek 4. Informacje o wybranym pacjencie

Środowisko oraz biblioteki potrzebne do uruchomienia projektu:

Python
Sqlite3
numpy
Flask
Flask_restful
Plotly

W celu odpowiedniego działania aplikacji wymagane jest utworzenie folderu *files* w folderze z rozpakowanymi plikami (rys. 5).

W pliku *config.py* należy wprowadzić ścieżkę do pliku *patients.db* oraz folderu *files* (rys. 6).

C:\Users\Kacper\Documents\git\telm21z		Przeszukaj: telm21z	
Nazwa	Data modyfikacji	Typ	Rozmiar
__pycache__	25.01.2022 22:27	Folder plików	
files	25.01.2022 22:24	Folder plików	
clear-results-table	25.01.2022 00:12	Python File	1 KB
config	25.01.2022 19:36	Python File	1 KB
create-tables	25.01.2022 22:22	Python File	2 KB
dbcommands	25.01.2022 22:27	Python File	2 KB
emulator	26.01.2022 20:05	Python File	2 KB
index	25.01.2022 23:48	Chrome HTML Do...	4 KB
patients	26.01.2022 19:56	ANSYS 2019 R3 .d...	20 KB
server	26.01.2022 19:40	Python File	4 KB

Rysunek 5. Folder z projektem po rozpakowaniu i utworzeniu folderu „files”

```

config.py
config.py > ...
1 DB_PATH = r"C:\Users\Kacper\Documents\git\telm21z\patients.db"
2 FILES_PATH = r"C:\Users\Kacper\Documents\git\telm21z\files"

```

Rysunek 6. Przykładowe ścieżki do pliku „patients.db” oraz folderu „files” w pliku „config.py”

Instrukcja użytkownika systemu:

1. Uruchomienie *create-tables.py*.

```
C:\Users\Kacper\Documents\git\telm21z>python create-tables.py
```

2. Wywołanie *emulator.py* z argumentami odpowiadającymi *patient_id* oraz ilości wyników, które mają zostać utworzone dla danego pacjenta. W przykładzie dla pacjenta id=1 utworzonych zostało 30 rekordów z danymi.

```
C:\Users\Kacper\Documents\git\telm21z>python emulator.py 1 30
```

3. Uruchomienie pliku *server.py*, działającego do momentu wyłączenia.

```
C:\Users\Kacper\Documents\git\telm21z>python server.py
```

4. Podczas działania *server.py* należy otworzyć plik *index.html* poprzez dwukrotnie kliknięcie lub przeciągnięcie pliku z folderu do okna przeglądarki.

<< Dokumenty > git > telm21z		Przeszukaj: telm21z	
Nazwa	Data modyfikacji	Typ	Rozmiar
__pycache__	25.01.2022 22:27	Folder plików	
files	25.01.2022 22:24	Folder plików	
clear-results-table	25.01.2022 00:12	Python File	1 KB
config	25.01.2022 19:36	Python File	1 KB
create-tables	25.01.2022 22:22	Python File	2 KB
dbcommands	25.01.2022 22:27	Python File	2 KB
emulator	26.01.2022 20:05	Python File	2 KB
index	25.01.2022 23:48	Chrome HTML Do...	4 KB
patients	26.01.2022 19:56	ANSYS 2019 R3 .d...	20 KB
server	26.01.2022 19:40	Python File	4 KB