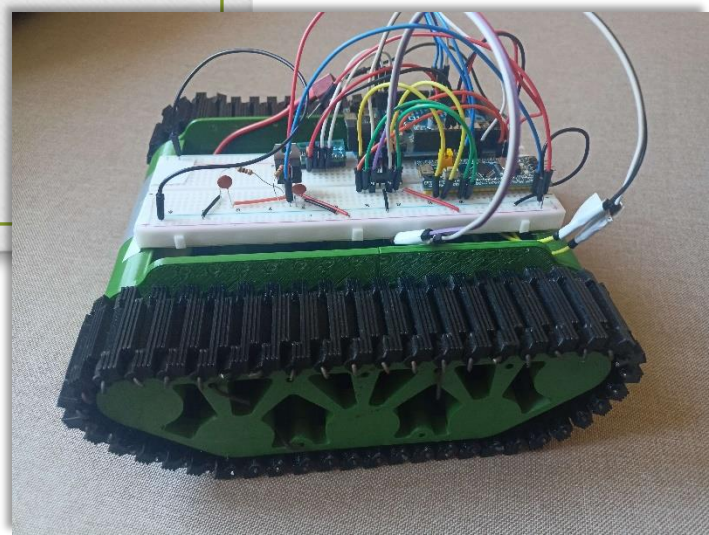
Robot mierzący przemieszczenie i przyspieszenie.

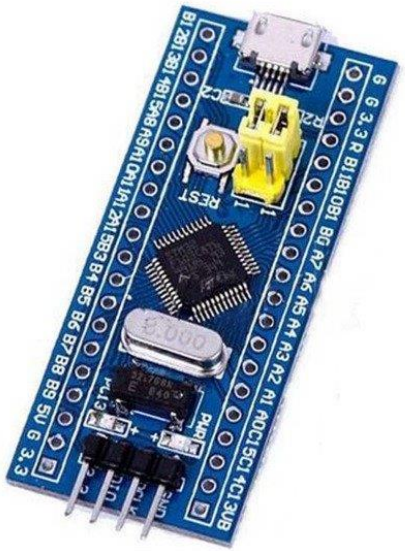
Michał Nizioł, Piotr Łętowski

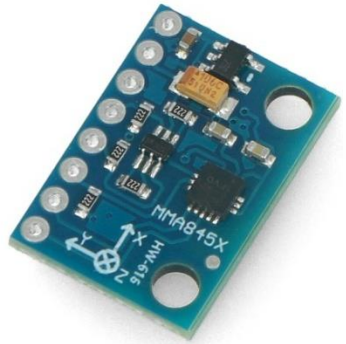
AGH



Założenia projektu

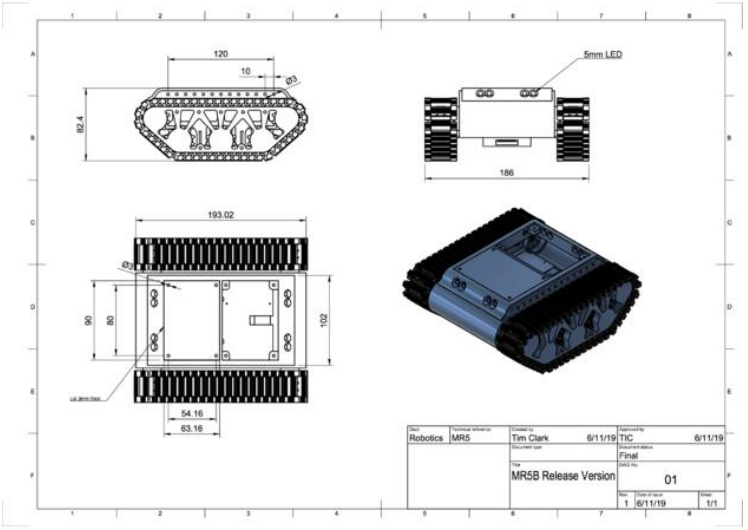
- Projekt jest robotem na podwoziu gąsienicowym sczytującym przyspieszenie z akcelerometru. Na podstawie odczytanego przyspieszenia obliczane jest przemieszczenie robota. Sterowanie robotem odbywa się za pomocą GUI znajdującym się na postawionym rest-api.





Spis użytych elementów

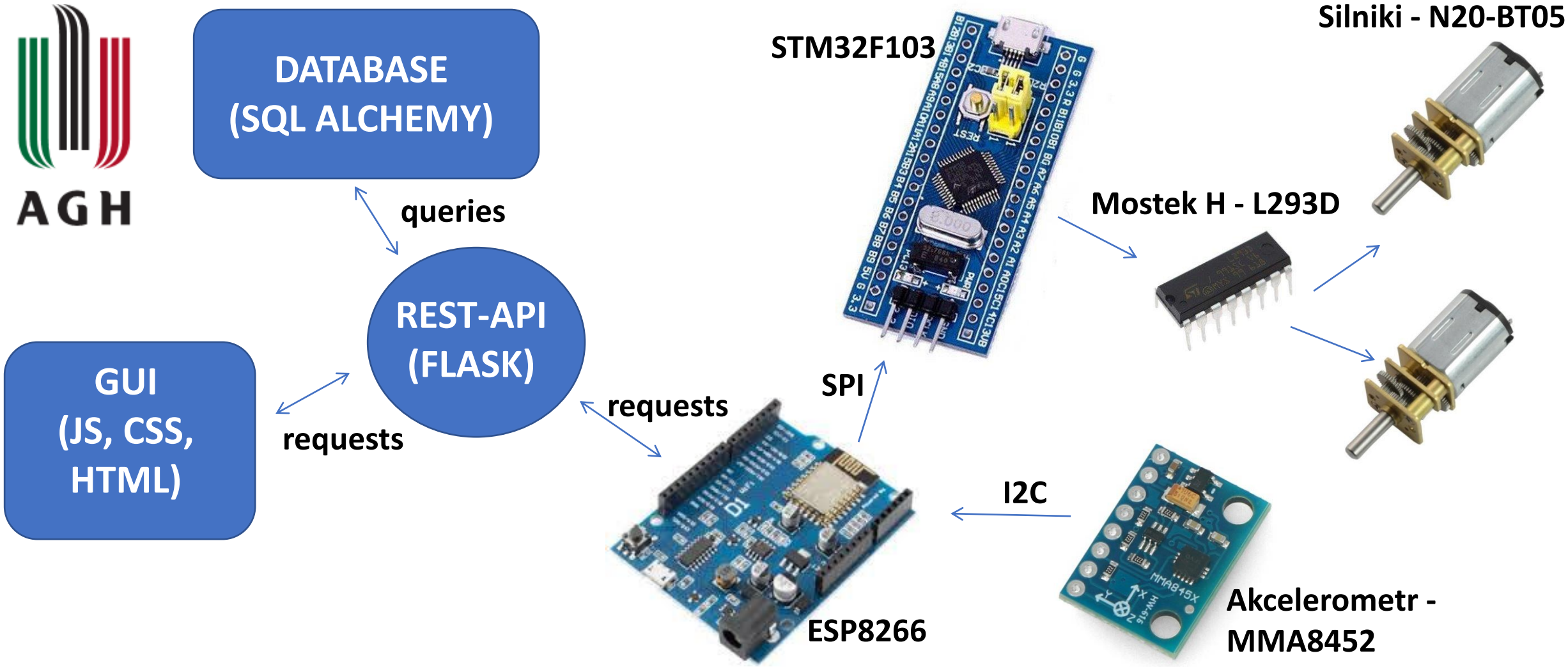
- Hardware - Mikrokontroler STM32F103C8T6 ARM Cortex M3
- 3-osiowy akcelerometr cyfrowy MMA8452
- ESP8266 z modułem WiFi
- Stabilizator napięcia LM317
- 2 silniki DC 12V N20-BT05 micro 50:1 625RPM.
- Podwozie gąsienicowe wykonane w technologii druku 3D ().



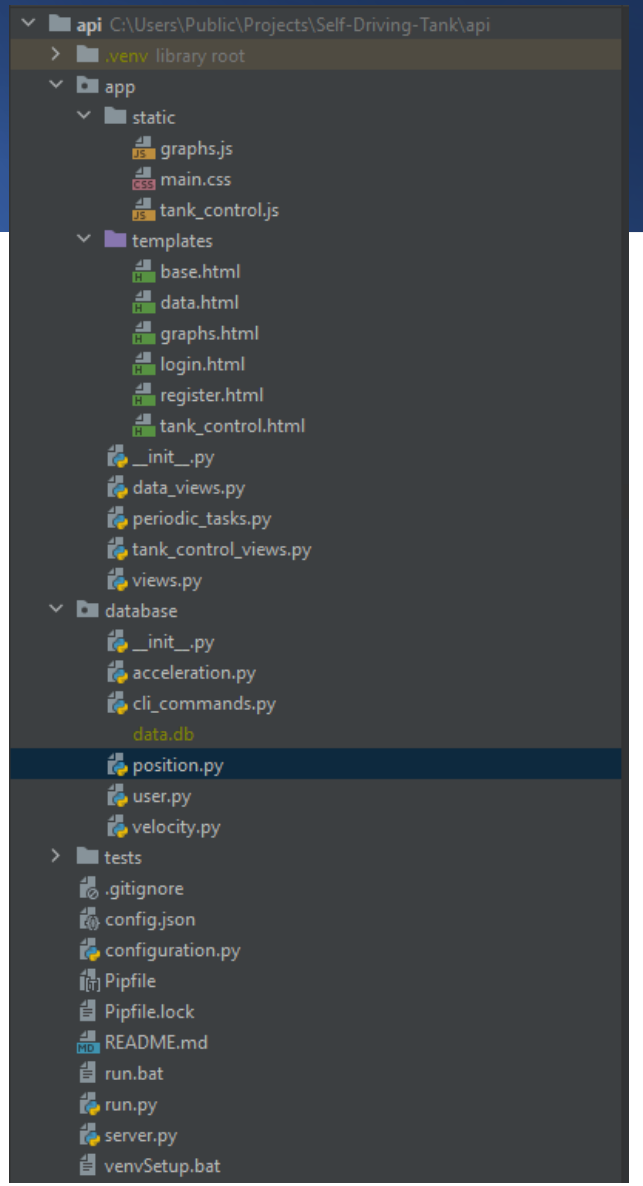
Opis techniczny

- Akcelerometr jest zamieszczony na pojeździe sterowanym przez rest-api jeżdżącym na gąsienicach. ESP8266 jest odpowiedzialne za przesyłanie danych przez WiFi do rest-api i odczytywanie danych z akcelerometru.
- Komunikacja z interfejsem następuje przez połączenie WiFi. Aby otrzymał on dane z akcelerometru, mikrokontroler musi wysłać odpowiedni request.
- Za sterowanie robotem odpowiedzialna jest STMka, która steruje silnikami przy pomocy Mostka H.
- Dane z akcelerometra (przyspieszenie na osi X, Y, Z) są przechowywane w bazie danych (SQL alchemy), tak jak i informacje o względnej pozycji robota oraz chwilowej prędkości na osi X i Y.
- Funkcja add_acc (POST metoda) służy do zapisania danych z akcelerometra do bazy danych. Oblicza ona pozycję oraz prędkość robota, na podstawie odpowiednich wzorów, danych teraźniejszych i poprzednich.

Działanie projektu - diagram



Software – struktura projektu



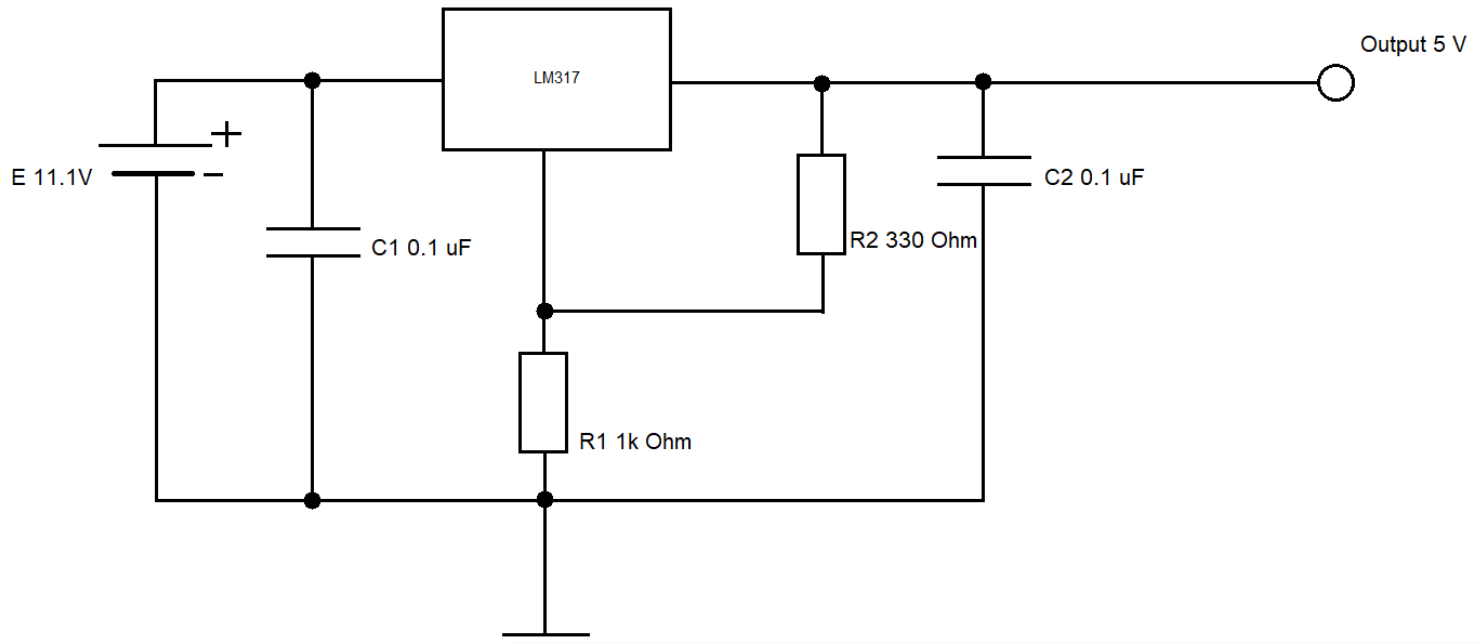
- Wszystkie wymagane pythonowskie biblioteki są zawarte w pliku Pipfile. Aby projekt działał poprawnie, należy najpierw uruchomić plik venvSetup.bat, który przy pomocy pipenv pobierze wszystkie potrzebne biblioteki i utworzy wirtualne środowisko (.venv) .
- W pliku configruation.py zawarta jest cała konfiguracja aplikacji. Serwis można uruchomić poprzez uruchomienia run.py lub run.bat.
- W folderze database znajdują się wszystkie skrypty odpowiedzialne za implementację schematów tabel bazy danych oraz komend pozwalających na jej obsługę, jak i sama baza danych (plik data.db).
- W folderze app znajduje się właściwa Flask'owa aplikacja i views, czyli zdefiniowane i obsłużone endpointy serwisu. W folderze templates znajdują się pliki .html, w static znajdują się pliki .css oraz .js.

Software – struktura projektu cd.

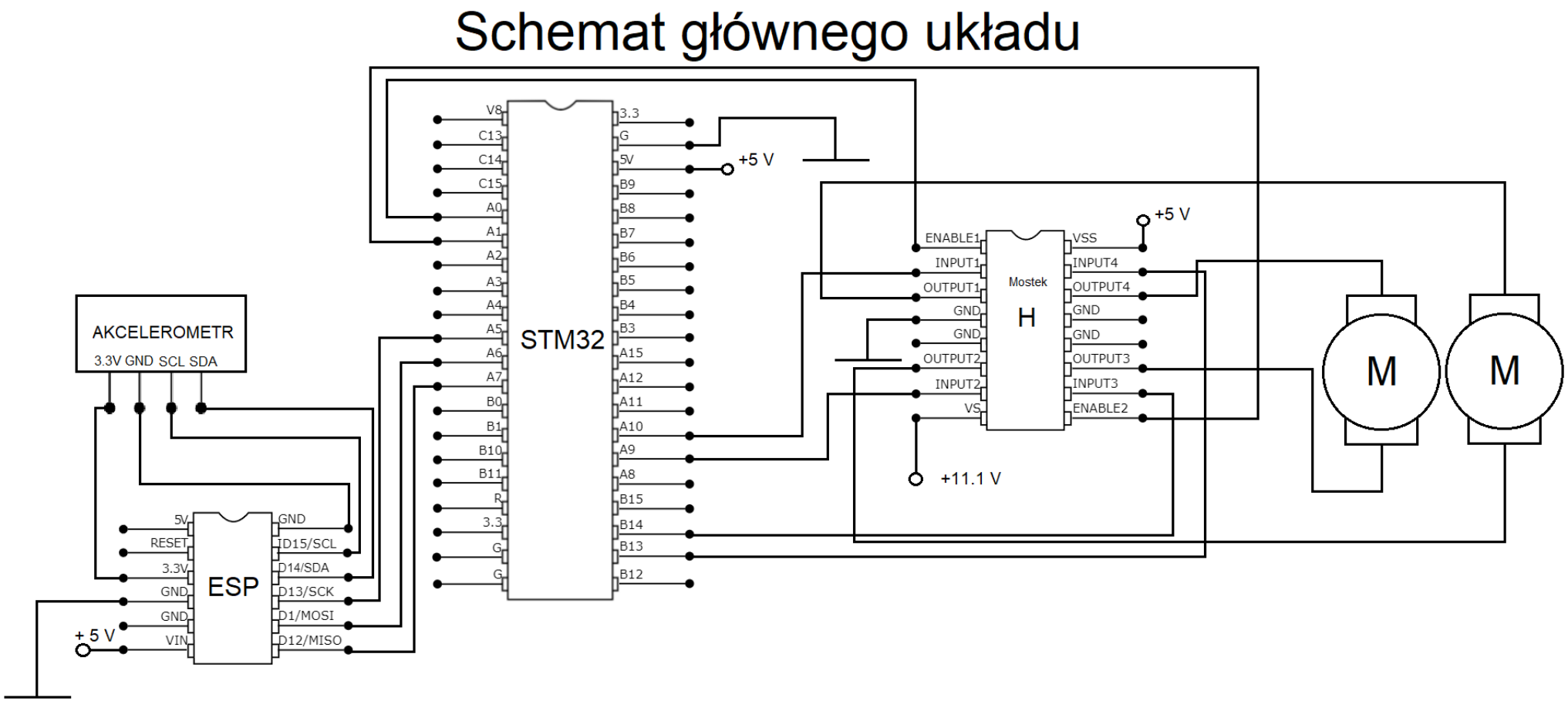
- W data_views.py znajdują się metody implementujące funkcjonalność CRUD (create, read, update, delete) dla danych zbieranych i przetwarzanych przez serwis.
- W tank_control_views.py znajdują się metody odpowiedzialne za sterowanie pojazdem.
- W esp znajduje się kod na mikrokontroler ESP-8266, napisany w C++ pobierający dane z akcelerometru przez I2C i wysyłający je do rest-api oraz przesyłający dane do STMki przez SPI.
- W stm znajduje się kod na STM32F103 odpowiedzialny za sterowanie silinkami, poprzez mostek h
- W periodic_task.py znajduje się symulacja otrzymywania danych z akcelerometru, wykonana za pomocą Schedulera, który wykonywał zadanie co określony odstęp czasu.
- Za pomocą metody get_state, dokonaliśmy symulacji kontrolek sterowania i jednoczesnego przetwarzania danych.

Schematy połączeń elektrycznych

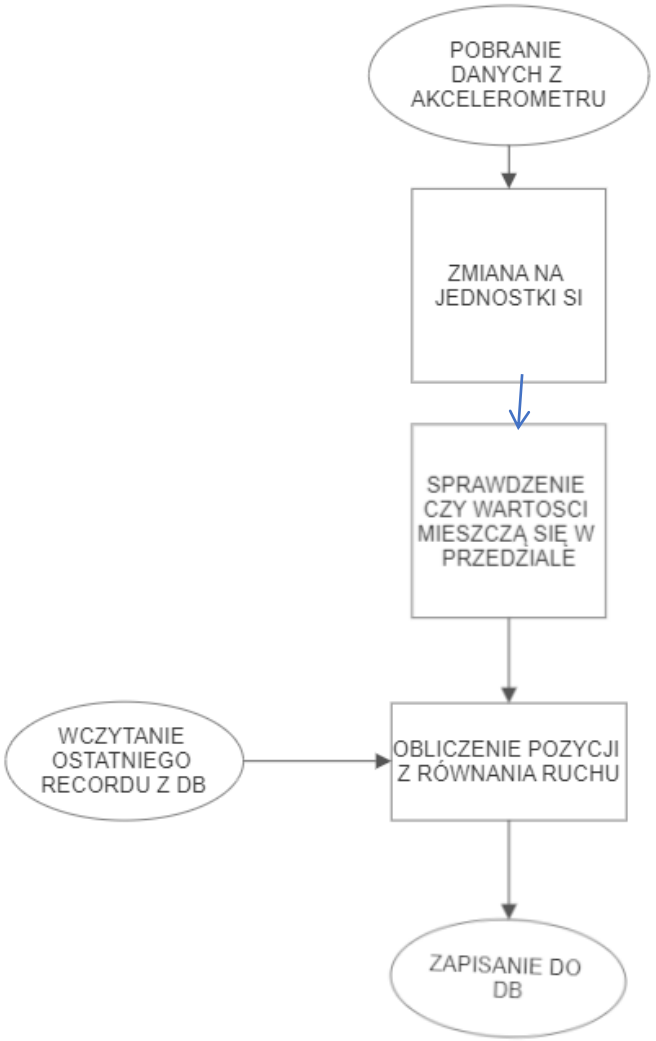
Schemat stabilizatora napięcia



Schematy połączeń elektrycznych

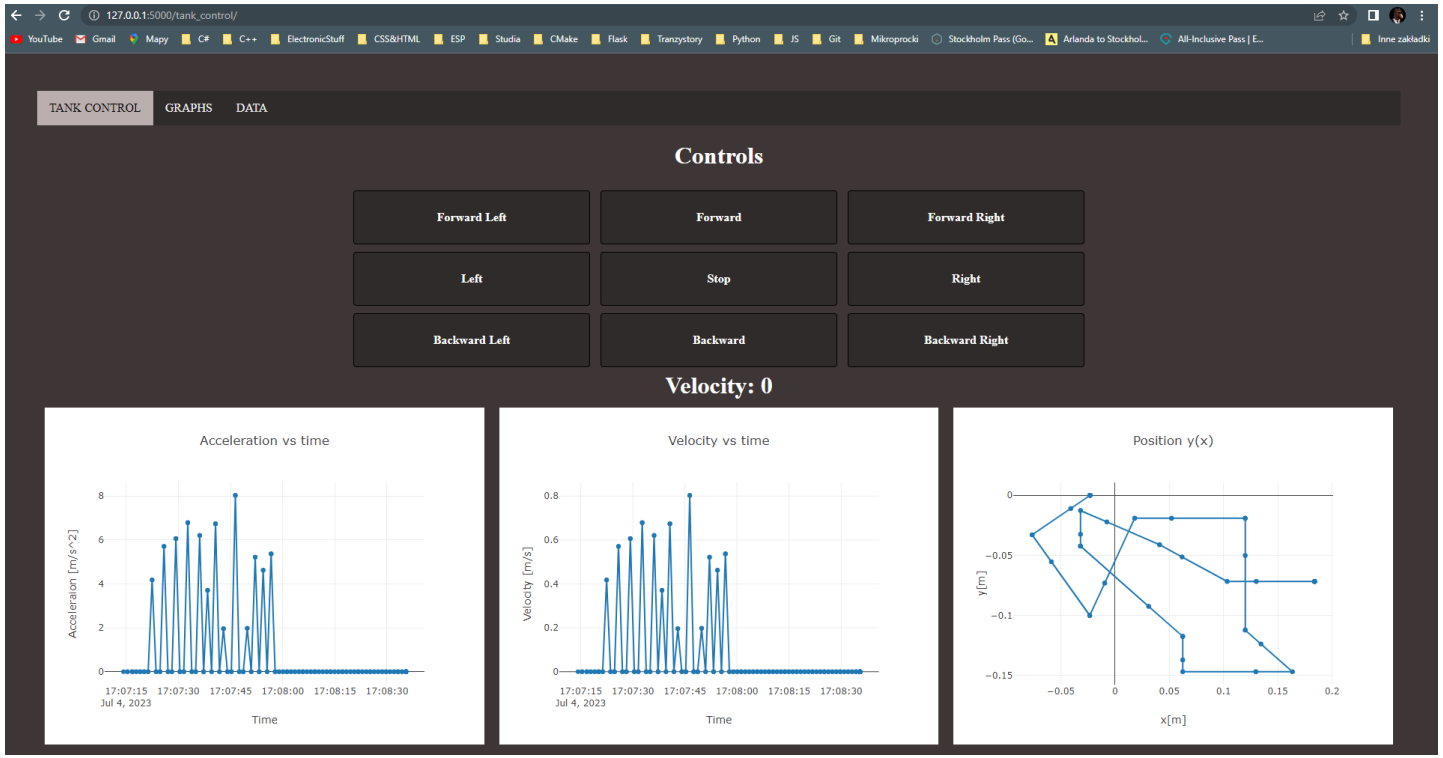


Schemat blokowy algorytmów



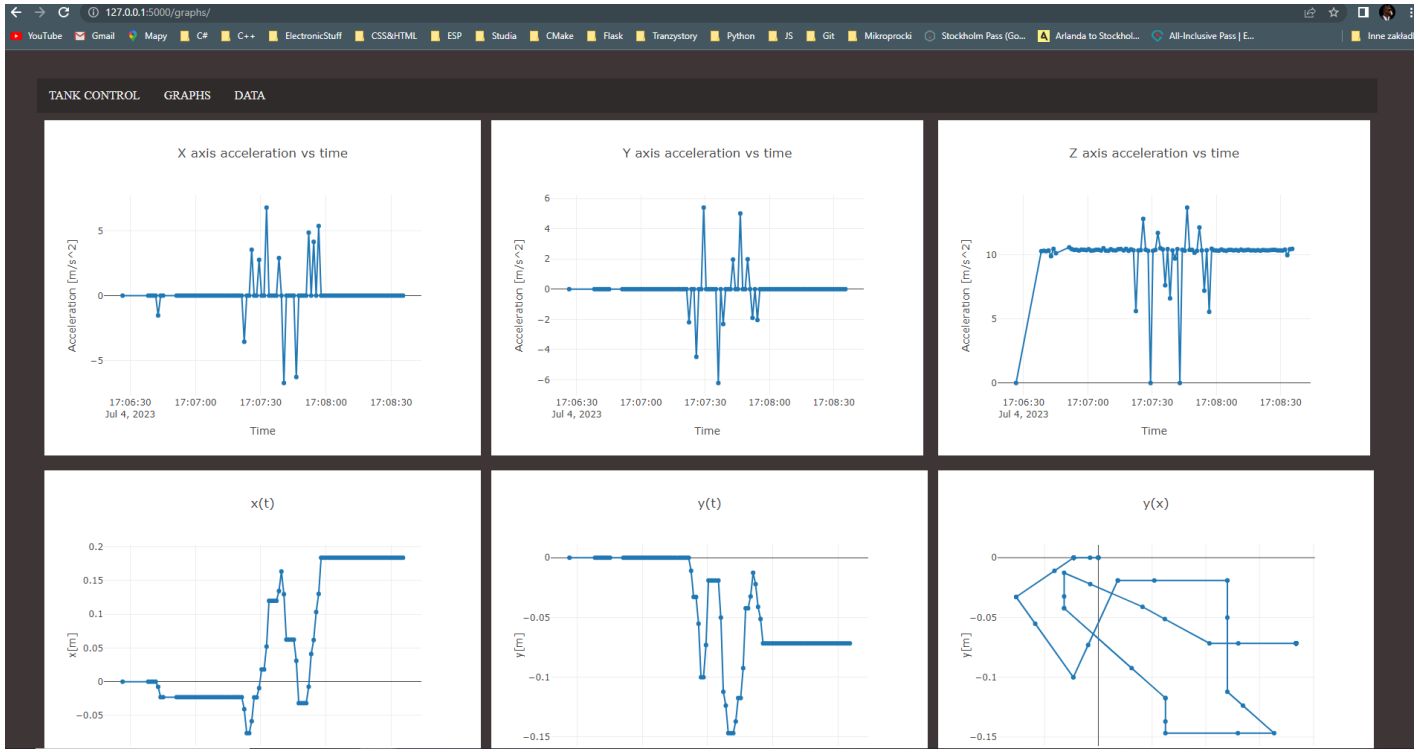
Algorytm oblicza szybkość i pozycję na podstawie odczytanego przyspieszenia oraz danych z poprzednich odczytów, które są pobierane z bazy danych. Do obliczania zostało wykorzystane równanie ruchu.
Jeśli dane odczytane z akcelerometru nie mieszczą się w ustalonym zakresie są ignorowane. W momencie zerowania przyspieszenia zerowane jest też prędkość chwilowa, żeby uniknąć błędów związanych z zatrzymaniem pojazdu.

Widok GUI - /tank_control/



Kontrolkami sterujemy kierunkiem poruszania się robota. Poniżej znajdują się wykresy rysowane w czasie rzeczywistym kolejno od lewej: Wypadkowego przyspieszenia od czasu, wypadkowej prędkości od czas oraz pozycji robota y(x)

Widok GUI - /graphs/



W zakładce graph wyświetlane są wykresy wszystkich danych zebranych do tej pory. (przyspieszenia dla wszystkich osi od czasu, pozycja x i y od czasu oraz x(y).

Opis uzyskanych rezultatów i przeprowadzonych testów

- Podczas testowania i pomiarów zauważono niedokładności w obliczonym przemieszczeniu do około 10% względem rzeczywistego przemieszczenia.
- Zauważono również opóźnienia w odczycie akcelerometru w czasie rzeczywistym wynoszące około 1-2s.
- Algorytmy na symulowanych danych działają poprawnie, więc winę ponosi hardware
- Zbyt małe próbkowanie danych z akcelerometru powoduje duże niedokładności w obliczeniach

Dziękujemy za uwagę