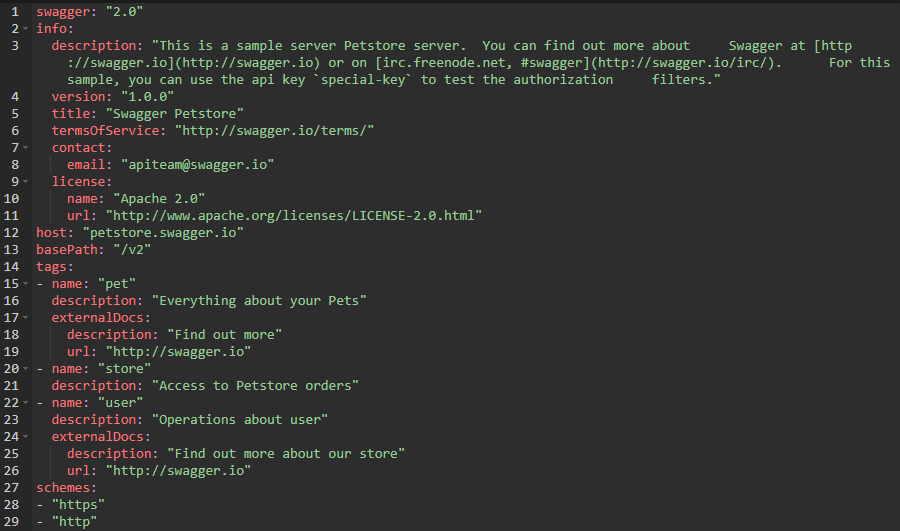
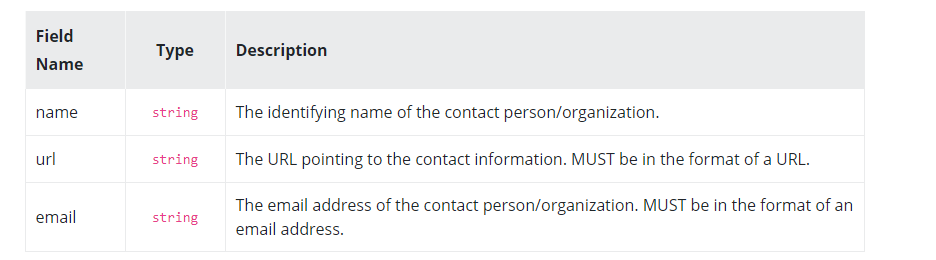
Struktura pliku yaml

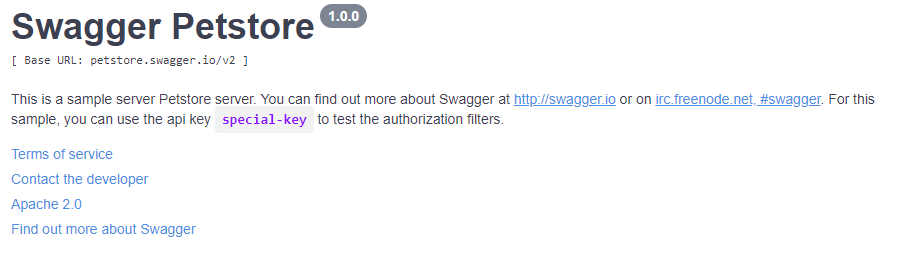
Swagger jest narzędziem do projektowania, prototypowania i dokumentowania API w języku yaml lub json. Swagger w projekcie jest naszym wyznacznikiem do budowy frontendu, backendu, oraz tworzenia bazy danych i dzięki takiemu podejściu co do budowy aplikacji może się wypowiedzieć każdy z członków projektu przed jego rozpoczęciem.   
  
Na poprzednich zajęciach było za zadanie przejść przez proces instalacyjny oraz zapoznać się z działaniem swagger-editora i składni yaml. Dzisiaj trochę więcej o swaggerze i dokładniejszym opisem działania.

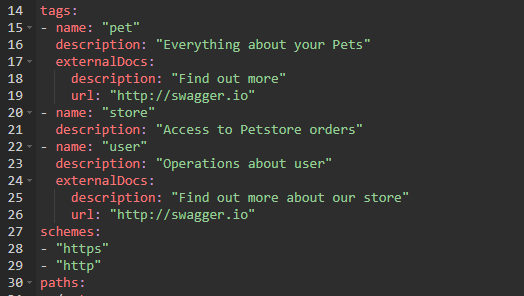


Bazując na przykładzie ze swagger-editora w linijce 1 znajduje się wersja edytora. W drugiej mamy info, gdzie znajdują się kolejno opis projektu, wersja, tytuł, regulamin, adres kontaktowy, licencja. W specyfikacji możemy znaleźć więcej opcji do ustawienia, dla przykładu pole kontakt może mieć również inne pola:

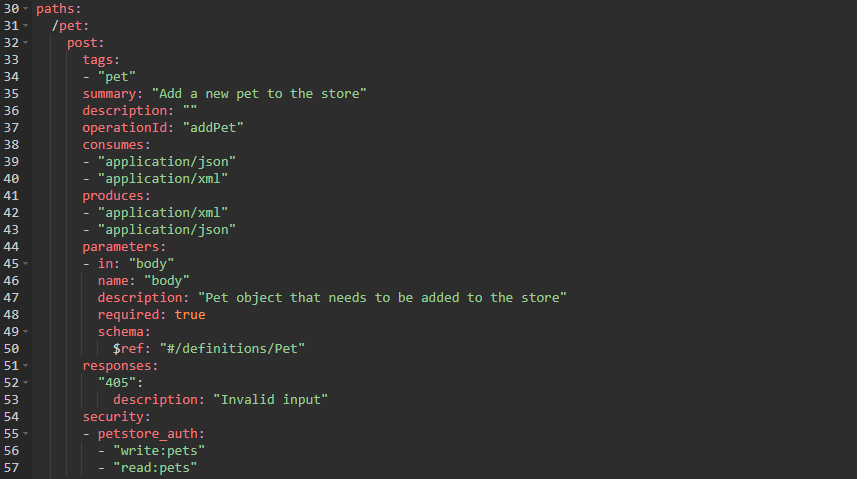
  
 Link: <https://swagger.io/specification/#contact-object>

Warto zauważyć że prawa strona automatycznie reaguje na wszystkie zmiany i opis z yamla został przedstawiony w eleganckiej formie na górze strony.



Kolejnym elementem który przyda się nam są opisy operacji. Możemy wykorzystać do tego tagi. Zaraz po ustawieniach projektu można zdefiniować tagi które będą używane w dalszej części.  
  


Link: https://swagger.io/docs/specification/2-0/grouping-operations-with-tags/

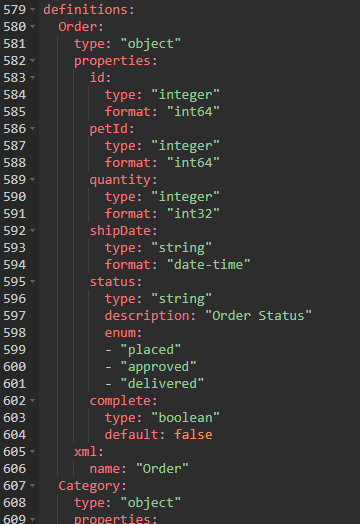
Kolejnym ważnym elementem struktury API są ścieżki:  
  


Pierwszą metodą CRUD w przykładzie jest Post służący do dodawania nowych elementów do bazy.  
Inne dostępne metody to

GET – pobieranie pozycji z bazy  
PUT – modyfikacja/aktualizacja   
DELETE – usuwanie danych  
Oczywiście często metoda PUT czy GET posiada id.

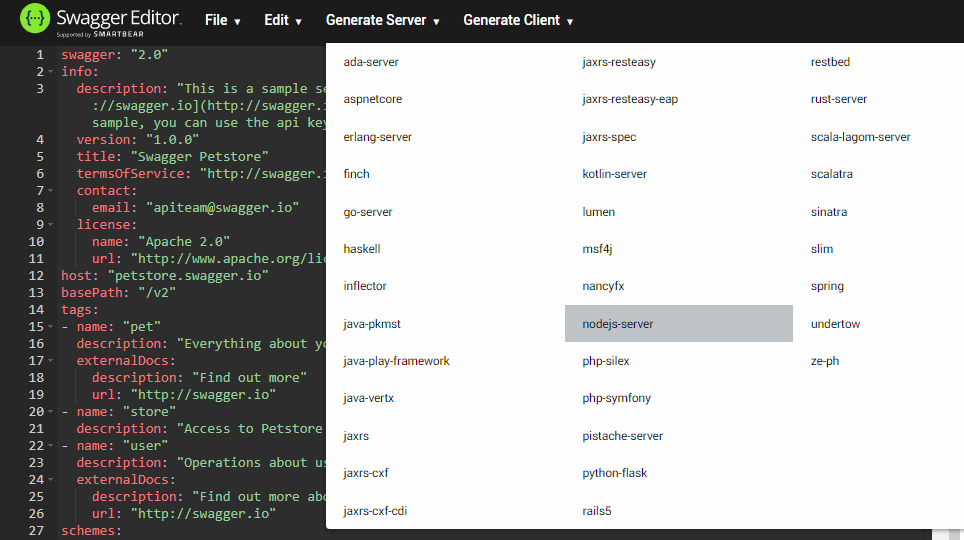
W linijce 33 przypisujemy tag który czytelnie pokaże nasze metody z opisem po prawej stronie oraz kolejne pole summary opisze daną metodę. Następnie możemy ustawić opis metody. Kolejne pola costumes i produces są to typy mime: https://swagger.io/docs/specification/2-0/mime-types/.  
Kolejnym ważnym elementem jest określenie parametrów, tutaj możemy zdefiniować typ prosty, albo skorzystać ze schematu poprzez $ref:ścieżka do definicji. Tutaj możemy również dodać przykłady <https://swagger.io/docs/specification/describing-responses/>. Wszystkie schematy należy umieszczać na dole pliku yaml. W linijce 51 możemy dodać odpowiedzi http. Najbardziej popularne to:  
200 – wszystko ok, 404 – nie znaleziono strony, 503 – usługa niedostępna, 500 – wewnętrzny błąd serwera. Od linijki 55 można definiować ustawienia bezpieczeństwa, w tym wypadku dostępny jest odczyt i zapis.

Definicje schematów:



W przypadku przykładu PetStore definicje zaczynają się od 579 linijki. Obiekt Order zawiera pola potrzebne do skonstruowania poprawnego schematu czyli id własne, id zwierzaka i Inne potrzebne w zamówieniu pola. Taka struktura będzie wzorem do stworzenia bazy danych.

Dzięki swagger-editor możemy później wygenerować szkielet wraz z kontrolerami.



W utworzonym w ten sposób szkielecie musimy doinstalować paczki za pomocą menagera npm, czyli komendą npm install, a następnie npm start. W tak utworzonym szkielecie mamy dostęp do wygenerowanych automatycznie kontrolerów w folderze „controllers” oraz mamy dostęp do pliku yaml w folderze api. Wygodnie jest na tym etapie korzystać ze Swagger-editora ponieważ interfejs od razu waliduje i generuje interfejs z prawej strony. Szkielet aplikacji przyda się później.