**Django - szkolenie 9**

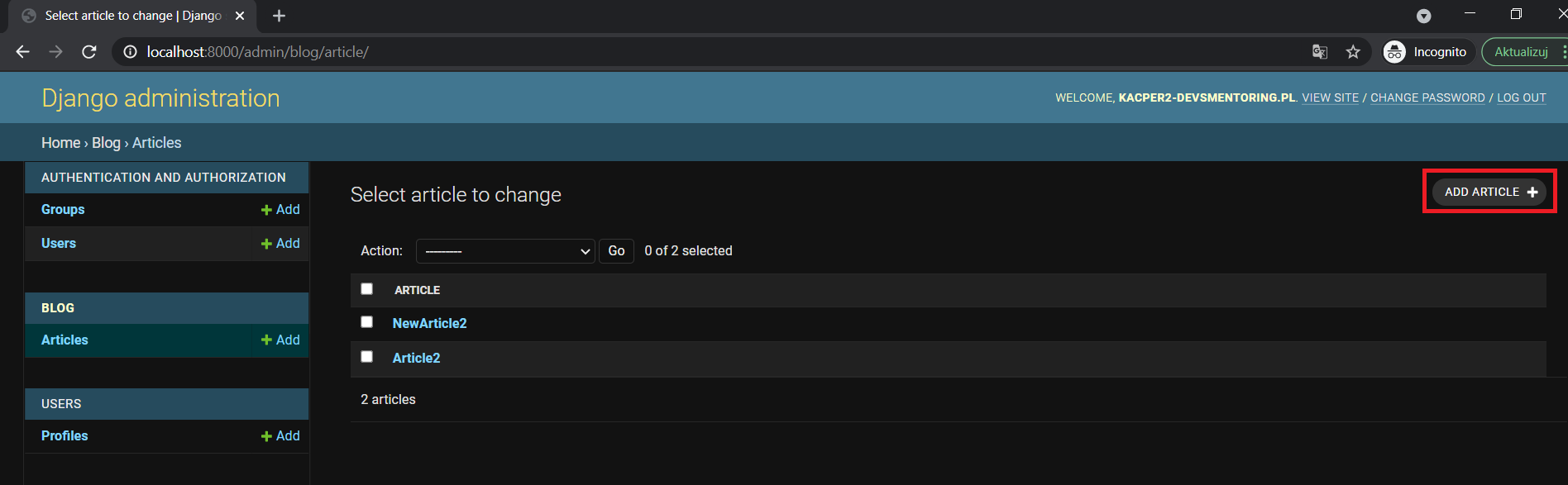
**Paginacja**

**Wstęp**

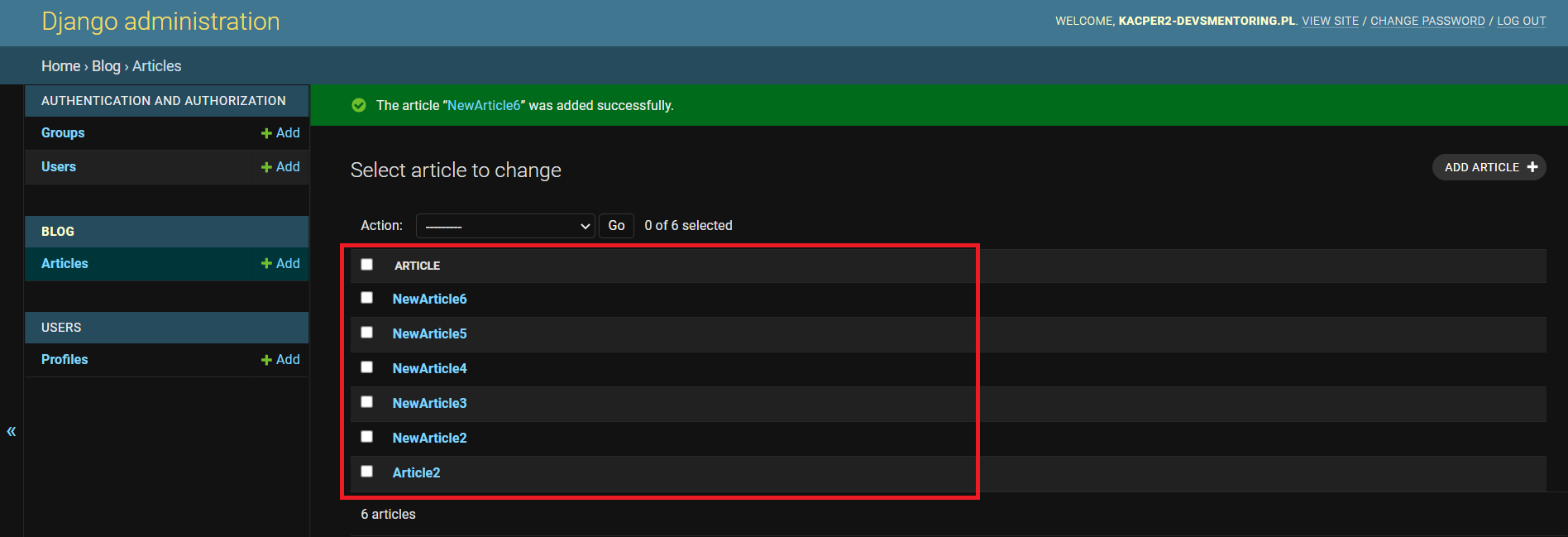
W szkoleniu tym wprowadzimy jedne z ostatnich poprawek do Naszej aplikacji. Celem będzie dodanie paginacji dla większej ilości udostępnionych artykułów na blogu. Za kilka chwil dodam do Naszego serwisu 4 nowe posty i jak się przekonasz, wszystkie zostaną umieszczone na jednej stronie. Naszym celem będzie rozdzielenie ich między kilka różnych podstron - tak, aby w ramach pojedynczego endpointa znajdowała się z góry określona ilość publikacji. Zrezygnowanie z przechowywania wszystkich artykułów w jednym miejscu zapewni Nam wiele korzyści. Otóż przede wszystkim strona nie będzie musiała za jednym razem renderować widoku wszystkich artykułów, dzięki czemu zoptymalizujemy jej czas ładowania. Dodatkowo, przyznaj, że totalnie nieintuicyjne byłoby trzymanie wszystkich postów na jednej podstronie. Gdyby ilość naszych publikacji wzrosła do kilkudziesięciu, to powstałaby ich jedna długa lista, co mocno zaburzyłoby UX klientów.

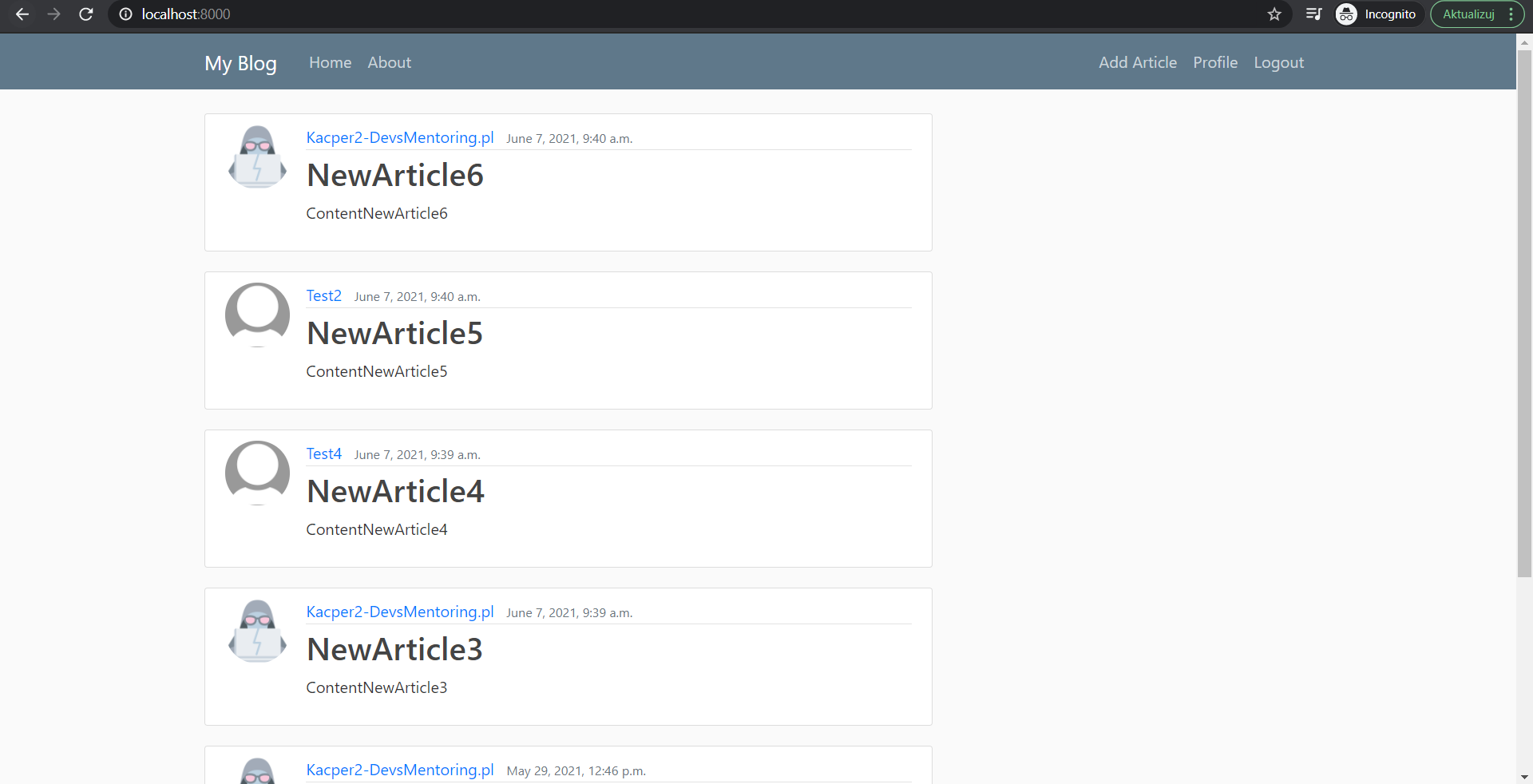
**Dodanie nowych artykułów**

Aby móc rozdzielić artykuły na kilka różnych stron, musimy najpierw wypełnić Naszą bazę danych nowymi publikacjami. Przejdźmy zatem do panelu administratora (localhost:8000/admin), zalogujmy się na konto i dodajmy nowe artykuły do tabeli Articles. Na etapie dodawania miałem już utworzone 2 artykuły i dodałem 4 kolejne.



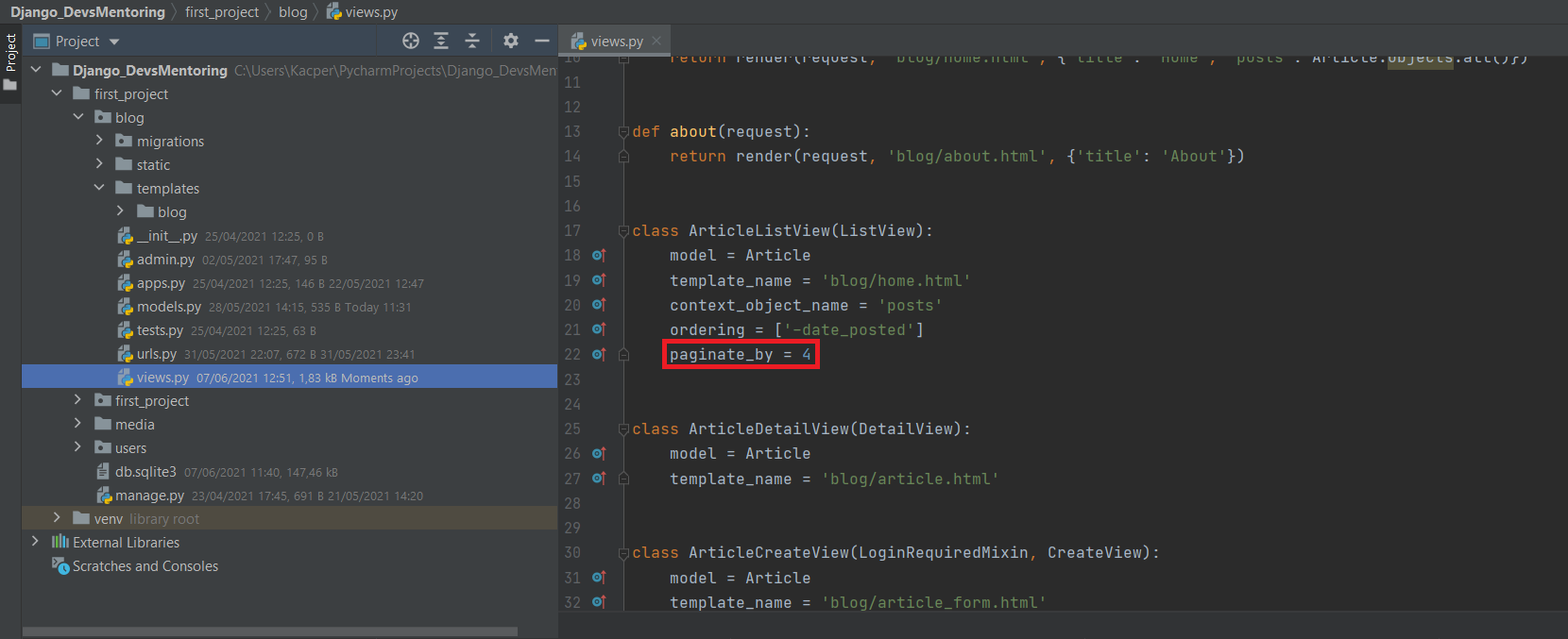
Po utworzeniu otrzymałem następujący efekt:



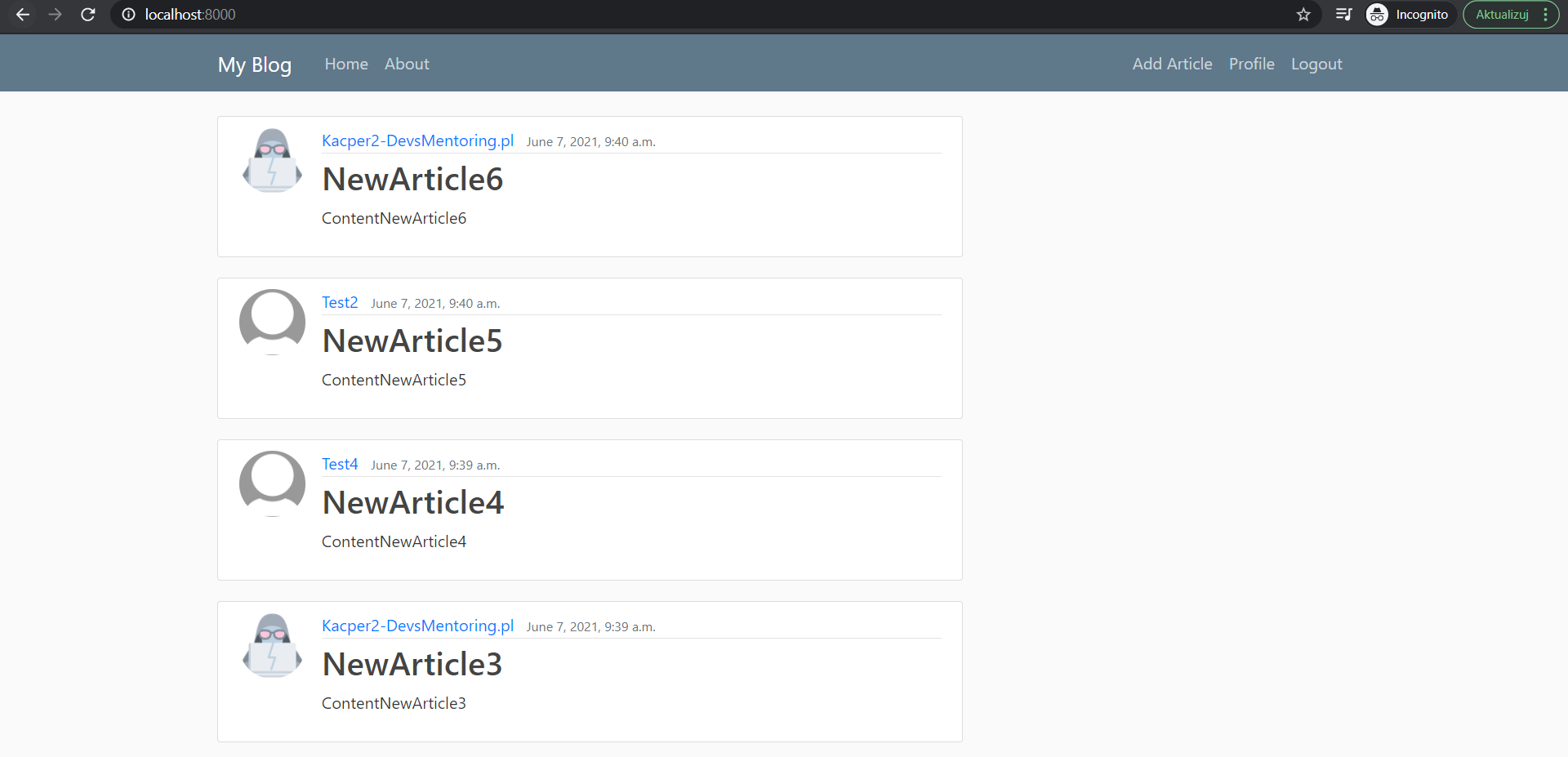


**Pierwsza zmiana w kodzie**

Zacznijmy od dodania parametru informującego o tym, iż generowany widok będzie podzielony na różne strony. Paginować będziemy class-based widok ArticleListView znajdujący się w pliku blog/views.py. Dodajmy zatem nowy atrybut paginate\_by, w którym określimy, ile maksymalnie artykułów ma znajdować się na pojedynczej stronie. Mając w tym momencie utworzone 7 artykułów, będę chciał podzielić je na 2 strony - tak więc na pojedynczy widoku znajdować się będą maksymalnie 4 posty.

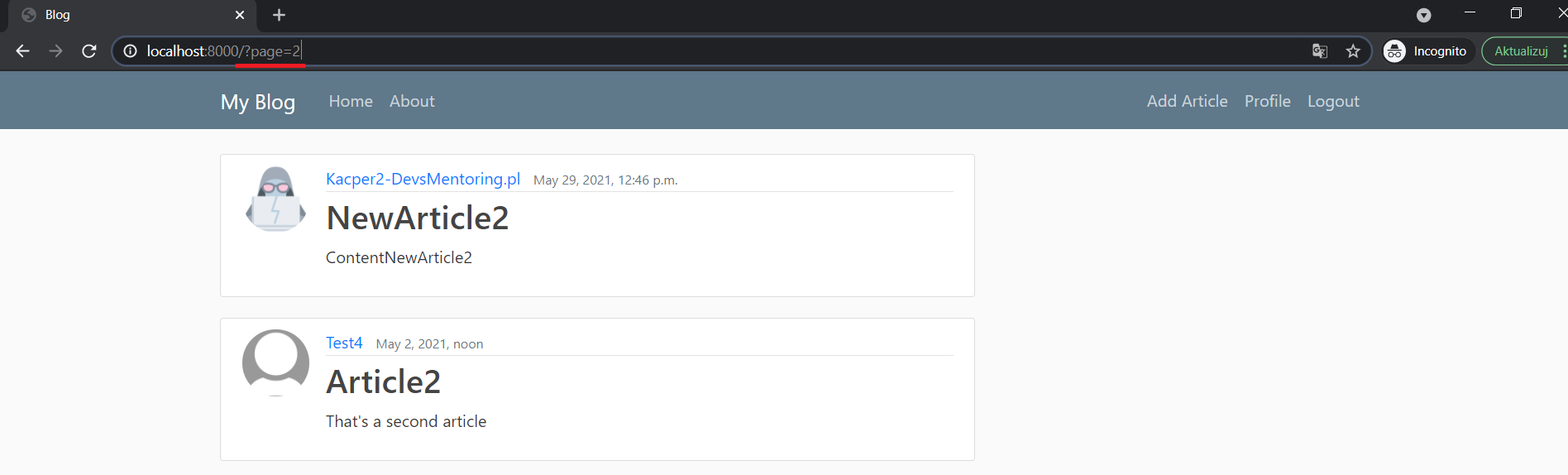


Teraz, gdy przejdziemy na stronę główną, powinniśmy ujrzeć tylko 4 z 7 dodanych artykułów.



Będziemy na pewno musieli dodać wyświetlanie numerów dostępnych zakładek na stronie głównej, ale na ten moment możemy na pewno zauważyć, że dodany atrybut paginate\_by = 4 faktycznie wprowadził zmiany w serwisie. Pytanie teraz, jak dostać się do pozostałych stron z artykułami? Otóż konieczne będzie dodanie parametru /?page=nr\_strony do ścieżki localhost:8000. W ten sposób będziemy mogli w wygodny sposób nawigować między konkretnymi stronami.

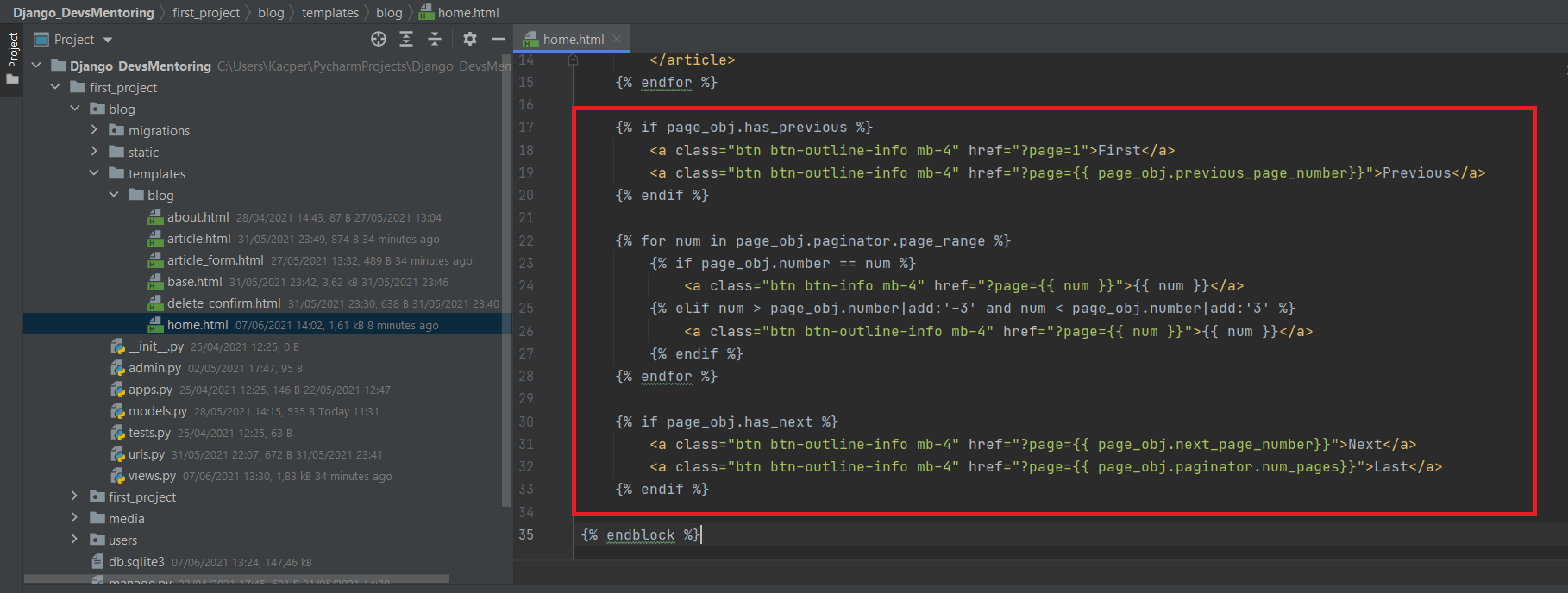
Spójrz, jak wygląda strona nr 2 z dwoma pozostałymi artykułami (zwróć również uwagę na podkreślony parametr ?page=2 umieszczony wewnątrz adresu URL):



**Dodanie numeracji na stronie**

Kolejnym krokiem będzie dodanie możliwości wyświetlania numeracji dla dostępnych stron. Zmianę tę wprowadzimy wewnątrz pliku blog/templates/blog/home.html. W ramach numeracji stron będziemy również chcieli zawrzeć odnośniki: prev/first oraz next/last, które przekierowywać nas będa kolejno do wcześniejszej/pierwszej lub następnej/ostatniej strony z artykułami. Oczywiście konieczne będzie także zaznaczenie kolorem numeru strony, na której aktualnie się znajdujemy.

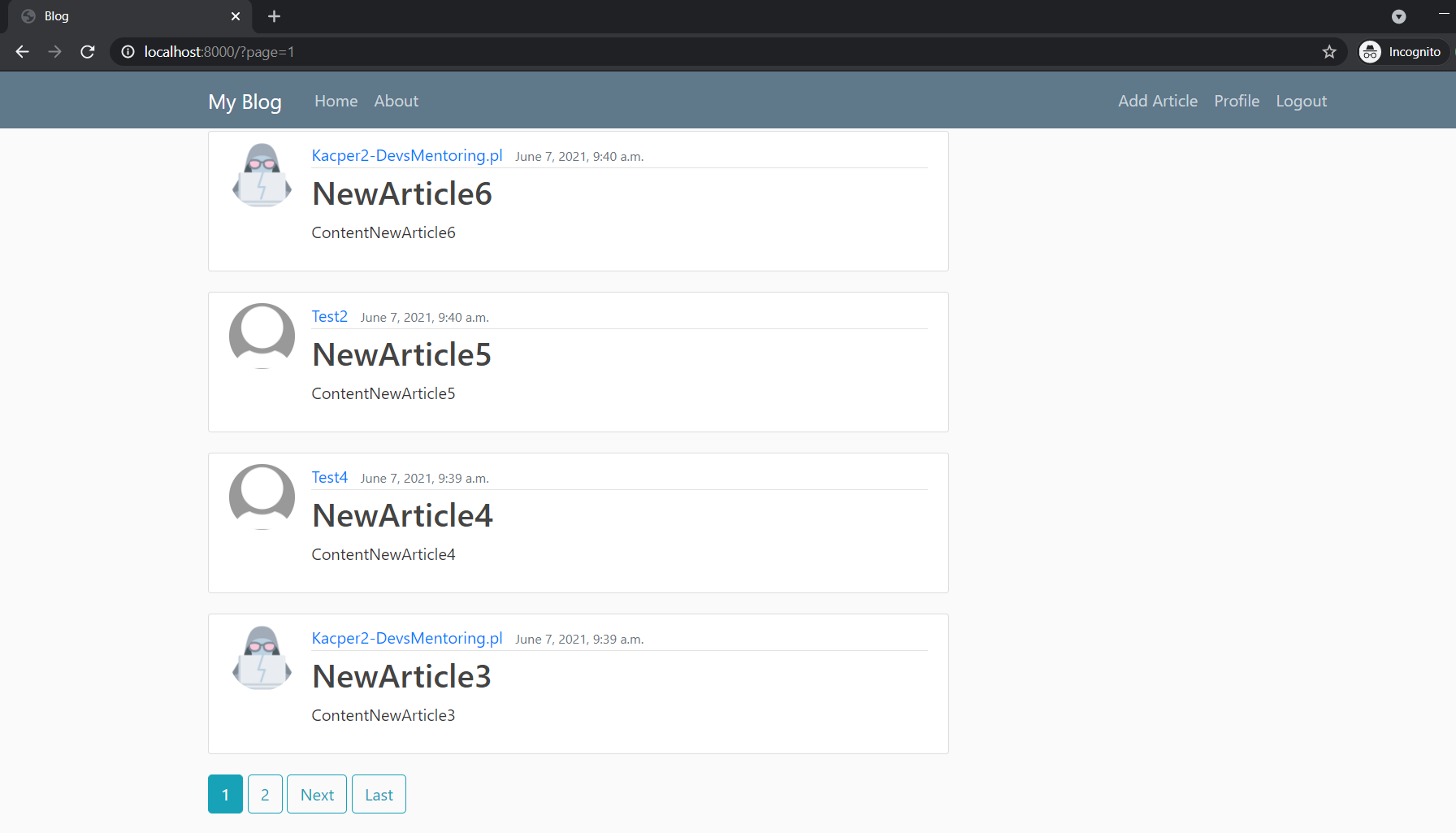
Przejdźmy zatem do zaimplementowania tych założeń w kodzie:



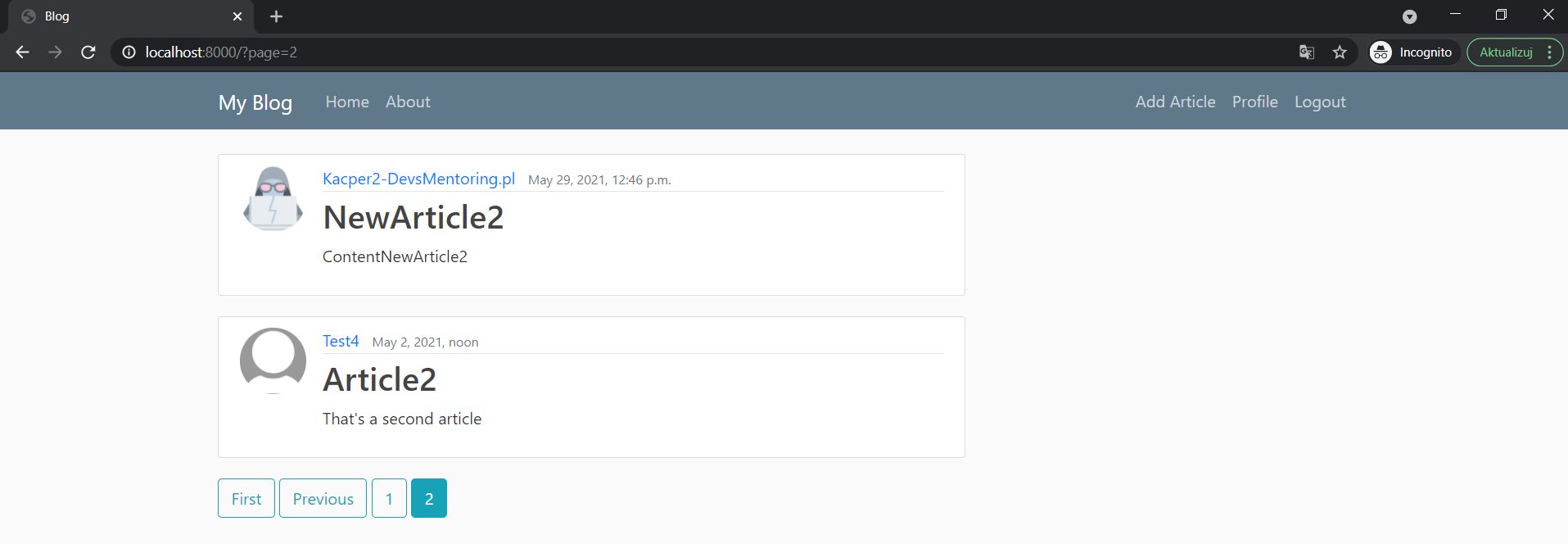
*Plik blog/templates/blog/home.html*

Myślę, że powyższy kod nie wymaga wielu wyjaśnień. Jest dość czytelny i samokomentujący się. Chciałbym jedynie zwrócić uwagę na zapis: num > page\_obj.number|add:’-3’ and num < page\_obj.number|add:’-3’. Umożliwia on nam filtrowanie i wybranie do wyświetlenie takich numerów stron, które są o 3 większe i o 3 mniejsze niż numer strony, na które się obecnie znajdujemy. Umożliwi to nam wyświetlenie numerów trzech sąsiadujących stron względem tej, na której się znajdujemy, a nie wszystkich jednocześnie. Tak więc znajdując się, np. na stronie numer 4, wyświetlone zostaną numery stron: 2, 3, 4, 5, 6, 7.

W efekcie otrzymamy:



*Pierwsza strona artykułów*



*Druga strona artykułów*