Maciej Walerczuk
Andrzej Dawidziuk

Zadanie 2 - Aplikacja wspierająca zapis studenta na projekt - dokumentacja końcowa

Technologie

Python3, Django (razem z systemem template'ów), HTML5, coverage, mommy, webtest

Dokumentacja użytkownika

1. Aby uruchomić serwer należy postępować zgodnie z krokami opisanymi w pliku README.md.
2. Uruchomiony serwer działa na porcie 8000.
3. Po wpisaniu odpowiedniego url-a w pole adresu w przeglądarce (np: http://localhost:8000) zostanie pobrana strona logowania.
4. Aby zarejestrować nowego użytkownika należy przejść na podstronę REGISTER

Zaimplementowana funkcjonalność

Udało się zaimplementować całą funkcjonalność opisaną w dokumentacji wstępnej oprócz następujących zmian:

1. Użytkownik uwierzytelniany jest za pomocą nazwy użytkownika i hasła, a nie maila i hasła
2. W interfejsie wyświetlana jest nazwa użytkownika, a nie jego imię i nazwisko

Pokrycie kodu testami (z użyciem Coverage.py)

Name	Stmts	Miss	Cover
projects_helper/apps/__init__.py	0	0	100%
projects_helper/apps/common/__init__.py	0	0	100%
projects_helper/apps/common/admin.py	11	0	100%
projects_helper/apps/common/forms.py	17	0	100%
projects_helper/apps/common/models.py	93	8	91%
projects_helper/apps/common/urls.py	4	0	100%
projects_helper/apps/common/views.py	49	20	59%
projects_helper/apps/lecturers/__init__.py	2	0	100%
projects_helper/apps/lecturers/forms.py	9	0	100%
projects_helper/apps/lecturers/urls.py	3	0	100%
projects_helper/apps/lecturers/views.py	74	12	84%
projects_helper/apps/students/__init__.py	2	0	100%
projects_helper/apps/students/urls.py	3	0	100%
projects_helper/apps/students/views.py	86	9	90%

TOTAL	353	49	86%

Wszystkie testy wykonywane są za pomocą polecenia “manage.py test”.
Opis jak wygenerować (i otworzyć) pełen raport coverage znajduje się w pliku README.md

Opis napotkanych problemów

Django

Żaden z nas nie miał wcześniej do czyniania z Django, co czyniło ten projekt dosyć ciekawym. Okazało się, że Django jest bardzo przyjaznym, łatwym do nauczenia frameworkiem. Wewnętrzny system template’ów Django pozwolił na stworzenie strony bez użycia JavaScriptu. Dodatkowo, dzięki integracji z bootstrapem3, niemalże znikła potrzeba pisania własnych CSSów.

Moduł autoryzacji

Problematyczna była integracja modułu autoryzacji Django z naszym specyficznym użytkownikiem (czyli dwoma rodzajami użytkowników Student i Lecturer). Problem został rozwiązany poprzez relacje 1 do 1 Użytkownik - Student oraz 1 do 1 Użytkownik - Lecturer oraz pole w Użytkowniku wskazujące z kim mamy do czynienia (L - lecturer, S - student). Doprowadziło to do sytuacji, w której każdorazowe odwołanie do studenta bądź wykładowcy wymaga zrobienia złączenia, w perspektywie wielu użytkowników może to prowadzić do problemów z wydajnością. Wydaje się, że najlepszą opcją byłoby wrzucenie wszystkich atrybutów studenta i wykładowcy do jednej encji (Użytkownik)

Podział systemu na moduły (Django apps)

W związku ze specyficzną budową “apps” w Django zdecydowaliśmy się stworzyć trzy moduły:

- common: moduł zawierający wszystkie modele oraz widoki do rejestracji i logowania użytkownika
- students: moduł zawierający widoki dostępne tylko dla studenta
- lecturers: moduł zawierający widoki dostępne tylko dla prowadzącego projekt (wykładowcy)

Rozwiązany został w ten sposób problem cyklicznych zależności pomiędzy modułami, ponieważ moduł common jest niezależny, natomiast moduły students i lecturers korzystają wyłącznie z common.
Taki podział pozwala również bardzo łatwo zarządać modelami, które znajdują się w jednym miejscu.

Czas poświęcony na projekt

Około 20 godzin na osobę - ze względu na brak praktyki w używaniu Django ciężko mierzyć czas pisania kodu.

Załączniki

- coverage: szczegółowy raport z testów (należy wygenerować według opisu w README.md)
- README.md: opis instalacji, plik znajduję się w głównym folderze projektu