

Pokemony 8

Tematyka

Pokemony to stworzenia, które trenerzy pokemonów łapia, z którymi podróżują, pracują, bawią się i za pomocą których rozgrywają walki z innymi trenerami. Niestety, niektórzy trenerzy wykorzystują je do popełniania przestępstw. Takich trenerów ścigają policjantki wykorzystujące pokemony.



Otrzymujesz od działu IT policji zlecenie stworzenia prototypu aplikacji do organizowania zgłoszeń o przestępstwach z wykorzystaniem pokemonów. Niedokończoną wersję tej aplikacji WWW z wstępnie wypełnioną bazą danych znajdziesz pod adresem <http://www.mimuw.edu.pl/~ciebie/wwwegz2018Lpopr-start.zip>.

Dane

Otrzymujesz niedokończoną aplikację WWW napisaną w Pythonie 3.6, Django 2.1.1 i Django Channels 2.1.3. Zawiera ona wypełnioną bazę danych z poniższym modelem.

Model Incident

| id (primary key) | reg_date | pokemon | suspect | description | assigned (foreign key) |
|------------------|------------|---------|---------|---|------------------------|
| 1 | 2018-09-05 | Spearow | | A boy was attacked on the road from Pallet Town to Viridian City by a flock of Spearows. | 1 |
| 2 | 2018-09-05 | Meowth | Jessie | Meowth and a trainer tried to rob a pokeball shop in Cerulean City. | 1 |
| 3 | 2018-09-11 | Koffing | | A trainer used Koffing's Smokescreen in Poke Center in Saffron City and attempted to steal pokemons. Might be connected to recent Team Rocket movement. | NULL |
| ... | ... | ... | ... | ... | ... |

Model ten zawiera informacje o incydentach zarejestrowanych w systemie. Data zgłoszenia `reg_date` nie musi być unikalna. Pola `suspect` i `assigned` są opcjonalne i gdy są puste są odpowiednio pustym stringiem i `NULLEM`. Pole `assigned` wskazuje na odpowiedni obiekt z modelu `django.contrib.auth.models.User`.

Początkowo aplikacja WWW posiada wypełnioną bazę danych, niefunkcjonalną stronę główną, niedokończoną aplikację RESTową i niedokończoną aplikację WebSocketową.

Zadanie na 3

Na stronie głównej umieść sekcję "Incydenty" z tabelką z wszystkimi zgłoszonymi incydentami. Powinna ona zawierać dla każdego incyduentu:

- datę zgłoszenia (w dowolnym formacie)
- nazwę rodzaju pokemona związanego z incyduentem
- nazwę podejrzanego trenera (lub "-" jeżeli takiego nie ma w bazie danych)
- pierwsze 80 znaków opisu zgłoszenia zakończone trzema kropkami (...) lub całość opisu bez kropek jeżeli posiada do 80 znaków
- nazwę użytkownika przydzielonej policjantki (lub "-" jeżeli nikt nie został przydzielony)

Oprócz tego na stronie głównej umieść na górze jakiś nagłówek oraz na dole stopkę z Twoim imieniem i nazwiskiem.

Data zgłoszenia powinna być linkiem prowadzącym do strony incyduentu, na której powinny być wymienione wszystkie powyższe informacje, tylko z pełnym opisem zgłoszenia.

Tabelka na stronie głównej powinna mieć obramowanie będące linią ciągłą grubości 2 pikseli, a wiersze powinny być od siebie oddzielone liniami przerywanymi (- - -) o grubości 1 piksela. Kolumny nie powinny być od siebie oddzielone liniami. Obramowanie tabelki powinno być oddalone o 10 pikseli od prawego i lewego brzegu strony.

Zawrzyj treści na stronie głównej i podstronach trenerów w odpowiednich semantycznych tagach HTML. Na stronie głównej użyj co najmniej tagów `header`, `section` i `footer`. Zadbaj, by każda strona była napisana w poprawnym HTML 5 i w poprawnym CSS. Nie wyłączaj zabezpieczenia przed atakami CSRF przy korzystaniu z mechanizmu sesji na serwerze.

Dodatki na wyższą ocenę

(+0.5) Dodaj funkcjonalność logowania i wylogowywania się użytkowników. Uczyń serwis dostępny tylko dla zalogowanych użytkowników, tj.:

- Niezalogowani użytkownicy powinni na stronie głównej widzieć formularz typu POST do zalogowania.
- Niezalogowani użytkownicy nie mogą mieć wyświetlonej sekcji "Incydenty", a dostęp do podstron incydentów powinien się kończyć błędem HTTP 403 "Forbidden".
- Zalogowani użytkownicy powinni widzieć dane tak, jak oryginalnie.
- Zalogowanym użytkownikom nie powinien wyświetlać się formularz do logowania, a na stronie głównej i podstronach incydentów powinna im się wyświetlać ich nazwa użytkownika oraz link do wylogowania się.

W przypadku porażki przy logowaniu lub wylogowywaniu się poinformuj o tym użytkownika.

(+0.5) Umieść przycisk do przydzielania wyświetlanego incydentu obecnie zalogowanemu użytkownikowi. Przycisk ten powinien się znaleźć na podstronach incydentów, które nie mają przydzielonej policjantki (tj. w bazie danych pole `assigned` jest równe `NULL`). Powinien się on wyświetlać tylko zalogowanym użytkownikom (np. zalogowanym poprzez panel administracyjny). Przydzielenie to powinno odbywać się za pomocą wysłania zapytania AJAXowego typu POST. W przypadku błędu wypisz użytkownikowi "Error", w przeciwnym razie zaktualizuj informację o przydzielonym użytkowniku i schowaj przycisk. Stwórz unit testy (zwykle, nie Selenium) uruchamiane poprzez `python manage.py test`, które przetestują obsługę zapytania AJAXowego w przypadku sukcesu, porażki z powodu już przydzielonego użytkownika oraz porażki z powodu braku zalogowania.

(+0.5) Dokończ implementację statycznej strony do wyświetlania przydzielonych zdarzeń znajdującej się pod adresem `/static/incidents.html`. Zalogowany użytkownik powinien po załadowaniu strony za pomocą RESTowego zapytania otrzymać zakodowane w formacie JSON informacje o przydzielonych mu incydentach. Informacje te powinny zostać wyświetlone w postaci tabelki z tymi samymi danymi, co na stronie głównej (ale z dowolnym wyglądem tabelki).

Stwórz na tej podstronie bezstanowe logowanie użytkownika nie wylogowujące go po odświeżeniu strony, tj. mechanizm, w którym użytkownik wpisuje swoją nazwę użytkownika i hasło, klika na przycisk do logowania, otrzymuje informację o byciu zalogowanym, a dane identyfikujące go są wysyłane z każdym zapytaniem RESTowym i sprawdzane po stronie serwera. Bezstanowość oznacza tu brak zapamiętywania informacji o zalogowanym użytkowniku po stronie serwera. W przypadku błędu lub złych danych do logowania wypisz użytkownikowi "Error". Wciąż możesz (a nawet potrzebujesz) użyć stanu po stronie przeglądarki użytkownika w postaci np. `localStorage`.

(+0.5) Dokończ implementację strony wyświetlającej informacje o pojawieniu się nowych incydentów znajdującą się pod adresem `/dashboard/` korzystając z WebSocketów. Po każdorazowym dodaniu incydentu do bazy danych z panelu administracyjnego Django (lub opcjonalnie też innych źródeł) powinna zostać wysłana informacja o nowym incydencie do wszystkich połączonych z tą stroną użytkowników. Powinna ona zostać wyświetlona w postaci dopisania do listy zdarzeń daty zgłoszenia incydentu będącej linkiem do jego podstrony. Istniejące przed nawiązaniem połączenia zdarzenia nie powinny zostać wyświetlone.

Do wykonywania kodu po każdorazowym dodaniu incydentu do bazy danych możesz użyć np. `post_save` z mechanizmu sygnałów Django. Do rozgłaszania WebSocketowym konsumentom wiadomości z kodu poza konsumentem będzie potrzebna Ci funkcja `channels.layers.get_channel_layer`. W przypadku błędu lub przerwania połączenia wypisz użytkownikowi "Error". Nie musisz zabezpieczać strony przed niezalogowanymi użytkownikami.

Reguły

Niedozwolone są jakiejkolwiek formy komunikacji. Można pracować na własnym sprzęcie i korzystać z swoich danych na studentsie, swoim laptopie, lub w postaci papierowej.

Można korzystać z następujących stron internetowych:

- <https://www.djangoproject.com/>
- <http://jquery.com/>
- <http://getbootstrap.com/>
- <http://www.w3schools.com/>, <http://www.w3.org/>
- https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API
- <https://virtualenv.pypa.io/en/latest/>
- <https://docs.python.org/3/>
- <https://stackoverflow.com/> (bez postowania!)
- <https://channels.readthedocs.io/en/latest/>
- <https://redis.io/>, <https://github.com/MicrosoftArchive/redis>, lub inny port
- <http://bulbapedia.bulbagarden.net/>
- Oficjalne dokumentacje używanych bibliotek i narzędzi
- <https://www.google.pl/> (w sensownym zakresie, jeśli ktoś nie wie, co to jest sensowny zakres, to dla niego sensowny zakres ogranicza się do poprzednich stron)

W szczególności nie jest dozwolone korzystanie z tutoriali innych niż znajdujące się powyżej. Oficjalne dokumentacje użytych bibliotek są ok.

Można serwować jQuery bądź Bootstrap z zewnętrznych serwerów. Do rozwiązania warto dodać README.txt opisujący co się zrobiło i dlaczego. Oprócz funkcjonalności będzie oceniany dobór metody i jakość realizacji.

Oddawanie

```
zip -r egz_www_2018_popr.zip katalog_z_egzaminem
```

Jeśli egz_www_2018_popr.zip ma więcej niż 3MB, to coś za dużo zzipowałeś. Umieść spakowany plik tak, aby był dostępny pod adresem:

```
http://students.mimuw.edu.pl/~twoj_login/egz_www_2018_popr.zip
```

```
cp egz_www_2018_popr.zip ~/public_html/  
chmod a+r ~/public_html/egz_www_2018_popr.zip  
chmod a+x ~ ~/public_html
```

Jeżeli używasz bibliotek Pythonowych innych niż django, channels i channels-redis, załącz o tym informację aktualizując requirements.txt np. w następujący sposób:

```
pip freeze > katalog_z_egzaminem/requirements.txt
```

O godzinie 13:00 skrypt pobierze Twoje rozwiązanie z katalogu domowego i wyśle Ci maila. Dopiero po odebraniu maila wyjdź z sali.