Kolejny ciężki poranek w firmie informatycznej C. Programiści powoli włączają komputery, rozmawiając niemrawo o nowym systemie, który właśnie tworzą. Tempo pracy nie jest zbyt duże dopóki nie nastąpi moment, na który wszyscy czekają. Wspólna poranna kawa![1]

Managerowie firmy C już dawno zauważyli jak dużą rolę w życiu programisty odgrywa kawa, więc zaprosili do współpracy kilku producentów tego ożywczego napoju, którzy to oferują różne rodzaje kawy w zależności od potrzeb i nastroju pracowników firmy C. Aby móc jeszcze lepiej dbać o swoich pracowników w przyszłości (i upewnić się, że poza piciem kawy, czasem też programują), prezes firmy C potrzebuje prostego systemu do monitorowania wydarzeń związanych z kawą w których uczestniczą jego programiści.

Każde wydarzenie ma swojego twórcę, cel oraz treść. Na tą chwilę system powinien obsługiwać 3 podstawowe typy zdarzeń.

1. twórca: Programista A, cel: Programistka B, treść: Programista A zaprosił na kawę Programistkę B
2. twórca: Programista A, cel: Producent B, treść: Programista A wypił kawę Producenta B
3. twórca: Producent A, cel: Programista B, treść: Producent A dostarczył swoją kawę Programiście B

Możliwe jest także zdefiniowanie własnego zdarzenia (z dowolnymi: twórcą, celem i treścią).

Akcje powinny być wyświetlane zarządowi na liście, w kolejności ich pojawiania (od najnowszych do najstarszych). Aby sprawdzić, kto z pracowników pije najwięcej kawy, oraz jakie rodzaje cieszą się największą popularnością, właściciele firmy C, chcą mieć możliwość filtrowania listy wydarzeń ze względu na twórców i cel zdarzenia.

Zespół grafików firmy C zaprojektował już prosty interfejs tworzonego systemu (możesz go zobaczyć w pliku CoffeeActivityFeed.pdf).

Pomóż managerom firmy C w polepszaniu jakości życia ich programistów!

Wymagania funkcjonalne:

1. Aplikacja powinna umożliwiać dodawanie opisanych powyżej różnych rodzajów akcji.
2. Akcje powinny być wyświetlane na liście obok interfejsu służącego do ich dodawania (dodanie nowej akcji powinno powodować pojawienie się jej na liście bez konieczności przeładowywania strony w przeglądarce).
3. Akcje powinno dać się filtrować zarówno po twórcy jak i celu (jak wyżej, nałożenie danego filtra nie powinno wiązać się z koniecznością przeładowania strony).
4. Po przeładowaniu strony dotychczasowe akcje powinny być nadal widoczne, a filtry zresetowane.
5. Dostarczony system powinien mieć już stworzonych programistów i dostawców kawy, dodawanie, usuwanie i edycja programistów i producentów NIE powinny być możliwe (wskazówki: South, migracje)

Możliwe rozszerzenia:

1. Powiedzmy, że akcje użytkowników nie będą wyświetlane jedynie na liście oglądanej przez managerów, ale również na ich profilach na Facebooku (w dzisiejszych czasach wypada informować świat o wszystkim co się robi!). Być może wyświetlane tam informacje powinny mieć inny format.
2. Mówi się, że picie zbyt dużych ilości kawy może prowadzić do spadku zawartości magnezu w organizmie. Aby tego uniknąć właściciele firmy C myślą o podpisaniu kolejnej umowy,

1 W firmie, do której aplikujesz wszyscy jesteśmy uśmiechnięci i pełni zapału do pracy już od pierwszej minuty!

tym razem z producentami czekolady.

UWAGA: wspomniane wyżej rozszerzenia NIE powinny zostać przez Ciebie zaimplementowane! Firma C oczekuje jednak, że jeśli zostaną one uznane za potrzebne w przyszłości, ich dodanie do systemu będzie łatwe, a powstały kod pozostanie elegancki.

Wymagania techniczne:

1. Aplikacja napisana przy pomocy Django.
2. Przy tworzeniu części JavaScriptowej pomocne może okazać się JQuery.
3. PostgreSQL.

UWAGA: wybór wersji firma C pozostawia Tobie, zasada jest prosta – im nowsze tym lepsze.

Aspekty podlegające ocenie:

1. Testy jednostkowe. Firma C uważa, że kod bez napisanych testów jest równie dobry jak brak kodu. Pierwsze miejsce na tej liście nie jest przypadkowe (wskazówki: django-coverage i liczba 100).
2. Zgodność napisanego kodu z ogólnie przyjętymi standardami pisania kodu w pythonie (wskazówka: PEP8)
3. Ogólna jakość projektu. Podział na moduły i klasy. Rozszerzalność (wskazówki: OOP, wzorce projektowe).
4. Zwięzłość i czytelność (wskazówka: PEP20).

Aspekty NIE podlegające ocenie:

1. Wygląd aplikacji. Jeśli znasz CSS nie zawahaj się go użyć, ale jeśli ma zająć Ci to dużo czasu, firma C woli abyś spędził go na dopracowywaniu projektu backendu swojego systemu).

Punkty bonusowe:

1. Zaimplementuj listę wydarzeń jako niekończącą się listę rozszerzającą się wraz z przewijaniem. Sprawdź jak radzi sobie z tym np. Twitter, wyświetlając listę wypowiedzi danego użytkownika.

UWAGA: Punkty bonusowe zostaną dodane jedynie jeśli oceny podstawowych aspektów będą wystarczająco wysokie. Innymi słowy, firma C, podobnie jak w przypadku CSS, woli abyś odpowiednio dopracował podstawowe funkcjonalności systemu niż zajmował się problemem nieskończonej listy.

W razie jakichkolwiek pytań pisz na m.rychlik@clearcode.cc

Zadawanie pytań to nie wstyd i na pewno nie wpłynie ono negatywnie na wystawioną Twojemu programowi ocenę!

Powodzenia!