Завдання

Етапу Півфіналу

DEV Challenge 11

**Номінація Back-end developer**

**Категорія Pro**

## 1. Завдання

Написати систему моніторингу за змінами у текстах новин / оголошень на сайті, збереження історії змін.
**Вхідні дані**: Оберіть будь-який сайт, який містить список публікацій (з пейджинатором). Наприклад, це може бути ось цей сайт (або будь-який інший).
**Моніторинг:** Система має вміти сканувати публікації, на які ведуть посилання, переходити по сторінках у списку та зберігати контент (лише html, без файлів).
Потрібно передбачити регулярне сканування найсвіжіших сторінок, і трішки рідше - більш старих сторінок. У разі виявлення змін у тесті публікації (документа) система має зберігати нову версію.
Врахуйте, що сторінка може також і зникати, тому такі випадки також потрібно виявляти.
Запускати job сканування можна будь-яким зручним для вас способом:

- Автоматично за таймером (або scheduler-ом)
- Шляхом виклику методу API (регулярність в такому випадку буде забезпечуватись ззовні шляхом регулярного виклику методу)
- Інші способи, які ви оберете

**Очікуваний результат:** API, що надає доступ до списку всіх публікацій, до списку змінених або видалених публікацій та до різних версій однієї публікації.
Ми очікуємо побачити у зручному вигляді diff (різницю) між різними версіями додатку. Видача результату, наприклад, може бути шляхом виклику методу API з параметрами - версіями документу, у відповідь може бути колекція доданих і видалених частин, наприклад. Алгоритм і вигляд результату повністю обираєте ви - головне, опишіть ваш хід думок. Можна використовувати сторонні бібліотеки.

## 2. Формат представлення результатів

1. Рішення вивантажувати в особистому кабінеті на сайті в ОДНОМУ файлі-архіві з назвою у форматі **Ім'я_Прізвище.zip**.

2. Рішення має бути надано у вигляді клієнтської і сервісної частини, підніматись в контексті одного віртуального оточення Vagrant/Docker. Для старту вашого додатка має бути необхідним запустити єдину команда: vagrunt up або docker run або docker-compose up  (Середовище, в якому судді запускатимуть ваші рішення – це Mac OSX Yosemite (10.10.5), Debian Jessie або Windows 10, ці ОС будуть ставитись з нуля в

базових налаштуваннях. Тому крайнє важливо, аби ви перевірили, чи піднімається ваш контейнер чи віртуальна машина на цих платформах).

3. README файл, в якому обов'язково вказуйте методологію, яку ви обрали, та пояснення, чому на ваш погляд вона найкраще підходить. Також описуйте, як працює ваш API, речі, на які ви б хотіли звернути увагу (вітається, якщо у вас буде UML-діаграма для вашої системи).

4. SCALEME файл, в якому описано, як ваша система маштабується, які для цього необхідні конфігурації.

Зверніть увагу, що назва архіву - єдине місце, де ви вказуєте свої персональні дані. Назви файлів всередині архіву не мають містити вашого ім'я чи прізвища.
Розмір архіву з рішенням не має перевищувати 10 MB.

2. Організатори та судді залишають за собою право дискваліфікувати роботу учасника, якщо робота:

    2.1. містить будь-яку вказівку на ім'я, прізвище, електронну пошту, компанію, адресу чи інші персональні дані учасника;
    2.2. виконана у іншому форматі, ніж вказано у завданні;
    2.3. виконана за допомогою сторонніх осіб, а не учасником особисто.

## 3. Обмеження та критерії оцінювання

1. Судді звертатимуть увагу на:
    1.1. працездатність додатку;
    1.2. кількість та адекватність коментарів в коді;
    1.3. архітектурні рішення і їх правильне використання;
    1.4. UNIT-тести та інші інженерні практики.
2. Бали нараховуватимуться за наступними критеріями:
    2.1. Рішення задачі та робота функціоналу, що вказаний в задачі, тобто підйом без необхідності дороблювати недостаючі частини\бібліотеки\конфігурації і так далі, та коректна робота функціоналу, що вказано в завданні.
    2.2. Якісний дизайн, що можна розширювати (software design) – тут ми дивитимемось, яку методологію ви обрали, та на скільки добре ви обдумали майбутнє розширення вашої системи і розширення вашого API.
    2.3. Системна архітектура вашого додатку – тут ми читатимемо вашу документацію, як саме ваша система в майбутньому маштабуватиметься, як ви конфігурували оточення та які інструменти обрали.

1.

## 4. Експерти

**Олексій Милоцький**
CTO/Co-Founder @Mil's
Переможець та неодноразовий суддя Чемпіонату, чия команда Mil`s виросла до технічного партнера компанії “We_Challenge”. Займається розробкою архітектури Front-end та Back-end частин для високонавантажених Web рішень. Вагомим доказом його досвіду є сертифікація ZCE/ZFCE/ZFCA та ZF2 contributor.  У Web-розробці понад 10 років.

**Сергій Ілюхін**
.NET Backend Team Lead @Ciklum
Більше 10 років досвіду розробки веб-додатків (ASP.NET, MVC, WebApi, AngularJs, сервіси Azure). Підтримує розвиток сервісів відкритих даних, автор та розробник сервісу моніторингу публічної інформації органів місцевого самоврядування. Співзасновник спільноти Freelance Brovary, тренер на волонтерських Java курсах в Броварах.

## 5. Контакти

1. Рішення необхідно вивантажити у особистому кабінеті на сайті devchallenge.it за обраною номінацією **до 4 червня, 23:59 (EEST)**. Після вичерпання часу можливість вивантажити роботи на сайт буде автоматично заблокована.
2. Питання та уточнення щодо змісту завдання ви можете задати, заповнивши форму. Відповідь на своє запитання шукайте в документі впродовж 24 годин. Перш ніж надсилати запитання, перевірте документ. Можливо, на ваше запитання вже є відповідь:)
3. Судді ігноруватимуть питання, які не стосуються завдання Чемпіонату.
4. Організаційні запитання надсилайте на пошту *team@devchallenge.it*
5. **Оголошення фіналістів відбудеться 19 червня**.