**ЗМІСТ**

[**ВСТУП** 7](#_Toc498857816)

[**1.** **ЗАДАЧА РОЗБОРКИ СИСТЕМИ A/B ТЕСТУВАННЯ** 8](#_Toc498857817)

[1.1. Мета розробки системи A/B тестування 8](#_Toc498857818)

[1.2. Застосування системи A/B тестування 9](#_Toc498857819)

[1.3. Опис підсистем системи A/B тестування 9](#_Toc498857820)

[**1.3.1.** **Web-застосунок системи A/B тестування.** 10](#_Toc498857821)

[**1.3.2.** **API системи A/B тестування.** 11](#_Toc498857822)

[**1.3.3.** **База даних MS SQL.** 11](#_Toc498857823)

[**2.** **ОГЛЯД ІСНУЮЧИХ ПРОГРАМНИХ РІШЕНЬ** 12](#_Toc498857824)

[2.1. Огляд роботи аналогів, їх переваг та недоліків 12](#_Toc498857825)

[3. МЕТОДИ РЕАЛІЗАЦІЇ ПРОГРАМНОЇ СИСТЕМИ 16](#_Toc498857826)

[3.1 Скрапінг HTML сторінки 16](#_Toc498857827)

[3.2 Методи маркетингових дослідженнь 17](#_Toc498857828)

[4. ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ СИСТЕМИ 22](#_Toc498857829)

[4.1 Архітектура системи 22](#_Toc498857830)

[**4.1.1.** **Етапи роботи системи.** 23](#_Toc498857831)

[**4.1.2.** **Опис Web-інтерфейсу системи A/B тестування.** 23](#_Toc498857832)

[**4.1.3.** **Архітектура API системи A/B тестування.** 24](#_Toc498857833)

[**4.1.4.** **Опис бази даних системи A/B тестування.** 25](#_Toc498857834)

[4.2 Програмні засоби розроблення системи 30](#_Toc498857835)

[**4.1.5.** **ASP.NET Web API.** 31](#_Toc498857836)

[**4.1.6.** **MS SQL.** 33](#_Toc498857837)

[**4.1.7.** **Angular.** 34](#_Toc498857838)

[**4.1.8.** **JQuery.** 35](#_Toc498857839)

[5. МЕТОДИКА РОБОТИ КОРИСТУВАЧА З СИСТЕМОЮ 36](#_Toc498857840)

[5.1. Системні вимоги та інсталяція програмного забезпечення 36](#_Toc498857841)

[**5.1.1.** **Апаратні та програмні вимоги.** 36](#_Toc498857842)

[**5.1.2.** **Інсталяція програмного забезпечення.** 37](#_Toc498857843)

[5.2. Сценарії роботи користувача з системою 42](#_Toc498857844)

[ВИСНОВКИ 48](#_Toc498857845)

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 49](#_Toc498857846)

# **ВСТУП**

Системи A/B тестування є потужним і надійним інструментом для маркетингового дослідження. Як відомо, в бізнесі не буває статичних станів. Підприємство повинно постійно розвиватися, щоб відповідати поточній ринковій ситуації, потребам клієнтів і власників. Щоб прибутковість проекту хоча б не падала, підприємцю необхідно постійно розширювати асортимент, збільшувати охоплення аудиторії за допомогою реклами та публікації корисного контенту, покращувати поведінкові метрики сайту і коефіцієнт конверсії.

Одним з інструментів розвитку веб-проектів є A/B-тестування. Цей метод дозволяє вимірювати переваги аудиторії і впливати на ключові показники ефективності сайту, включаючи конверсію, час перебування користувачів на сторінці, середню суму замовлення, показник відмов і інші метрики.

A/B-тестування — метод маркетингового дослідження, суть якого полягає в тому, що контрольна група елементів порівнюється з набором тестових груп, в яких один або декілька показників були змінені, для того, щоб з'ясувати, які зі змін покращують цільовий показник. Хороший приклад — це дослідження впливу колірної схеми, розташування та розміру елементів інтерфейсу на конверсію сайту.

Метод часто використовується при оптимізації веб-сторінок відповідно до заданої мети. Тестуються дві дуже схожі сторінки (сторінка А і сторінка В), які відрізняються лише одним елементом або декількома елементами (тоді метод називають A/B/N-тестування). Сторінки А і В показуються користувачам почергово в рівних пропорціях, при цьому відвідувачі, як правило, не знають про це. По закінченню певного часу або при досягненні певного статистично значимого числа показів порівнюються числові показники мети і визначається найкращий варіант сторінки.

# **ЗАДАЧА РОЗБОРКИ СИСТЕМИ A/B ТЕСТУВАННЯ**

Метою цієї роботи є побудова системи, що має можливість створювати задачі для проведення A/B тестування, а також на основі налаштувань збирати статистичні дані з web-сторінок та відображати результати у зрозумілому для користувача вигляді.

* 1. Мета розробки системи A/B тестування

Метою розробки є створення системи A/B тестування, яка буде акумулювати інформацію з сайтів для подальшого збереження та можливості експорту даних з системи агрегації до інших систем. Система має упорядковану структуру та об’єднує зібрані дані на одному ресурсі. Це дасть можливість користувачам системи отримати дані, які зберігаються на сайтах та використовувати зібрану інформацію з метою наповнення свого сайту контентом.

Система має використовувати:

* серверну технологію ASP.NET Web API [1];
* базу даних MS SQL;
* Angular [2] - Typescript фреймворк у якості клієнтської технології.

Задачі, які повинна виконувати вищеописана система:

* створення нової задачі для даної сторінки користувача;
* розпізнавання HTML [3] елементів сторінки для зручного вибору елементів сторінки;
* впровадження механізму заміни HTML елементів на сторінку користувача;
* механізм збору статистичних даних зі сторінки користувача;
* зручне відображення результатів A/B тестування.
  1. Застосування системи A/B тестування

Важливо також виділити основні задачі програмного продукту та визначити, в яких цілях він буде використовуватися користувачами.

Цільовою аудиторією системи є власники сайтів, у яких є потреба заповнювати свій сайт інформацією згідно тематики сайту. Вони можуть агрегувати дані як зі своїх дочірніх систем, так і з інших систем, контент яких знаходиться у вільному доступі та не є об'єктом авторського права.

Після того, як користувач визначиться з сайтом для обробки, він створює задачу і моя система її виконує. Крім того він зможе імпортувати дані до своєї системи.

Прикладом застосування є використання системи у статистичних центрах для визначення наступної інформації:

* середня ціна на товари в інтернеті;
* вартість путівок у туристичні місця;
* заробітна плата різних професій в різних регіонах;
* ціни на будівельні матеріали;
* визначення потреби роботодавців у працівниках;
  1. Опис підсистем системи A/B тестування

Таку систему можна розділити на дві незалежні підсистеми:

* система налаштування та створення задачі A/B тестування, яка дозволяє переглядати результати проведеного дослідження;
* система аналізу та збору статистичних даних, з можливістю динамічної підміни елементів.

У першій підсистемі вхідними даними є сторінка користувача, на якій будуть проводитися дослідження. Після проведення налаштувань вихідними даними, які будуть вхідними в другій підсистемі, повинні бути селектор елементу, який не буде змінюватись протягом досліджень, та два варіанти HTML коду цього елементу.

Як сказано вище, вхідними даними для другої підсистеми будуть вихідні дані першої. За допомогою Javascript [4] файлу, який користувач має розмістити на вказаній сторінці будуть проводитися дослідження. Після впровадження користувачем скрипту система повинна почати роботу. При відкриті сторінки користувача, будь-яким відвідувачем сайту, повинна відбутись заміна HTML елементу. Відображення одного елементу повинно відбутися для першої половини відвідувачів, та іншого елементу для другої половини. Таким чином, 50% користувачів будуть бачити один елемент, а інші 50% - інший.

Кожне відвідування сторінки та кожне натискання досліджуваного елементу повинні реєструватися системою та зберігатися для відображення в першій підсистемі для можливості подальшого статистичного аналізу.

* + 1. **Web-застосунок системи A/B тестування.**

Web-застосунок розроблений з використанням технології ASP.NET MVC [6], а також бібліотек JavaScript, які забезпечують взаємодію користувача з системою, роблять інтерфейс більш інтуїтивно зрозумілим, а також дозволяють розробнику організовувати взаємодію серверної та клієнтської частин. Він призначений для створення завдань агрегації, передбачення можливих посилань на сторінки за заданими селекторами, перегляду вже створених завдань, фільтрації за заданим ключовим словом, а також отримання результатів агрегації.



Рисунок 1.1 – Схема взаємодії компонентів системи агрегації контенту

* + 1. **API системи A/B тестування.**

Консольний застосунок розроблений для того, щоб перевіряти чи з’явились у базі даних нові задачі; у випадку якщо з’явились, то за заданими параметрами він починає збирати дані та зберігати створені структури у базу даних. Консольний застосунок працює автономно і не залежить від роботи Web-застосунку.

* + 1. **База даних MS SQL.**

Система A/B тестування використовує документо-орієнтовану базу даних MongoDB, яка дозволяє гнучко створювати структури залежно від того, яку інформацію треба агрегувати. Документо-орієнтована система керування базами даних — система керування базами даних, спеціально призначена для зберігання ієрархічних структур даних (документів). В основі документо-орієнтованих СКБД лежать документні сховища (колекції) котрі мають структуру дерева. Структура дерева починається з кореневого вузла і може містити кілька внутрішніх вузлів.

# **ОГЛЯД ІСНУЮЧИХ ПРОГРАМНИХ РІШЕНЬ**

В останні роки з’явилась достатня кількість програмного забезпечення, яка частково розв’язує задачу A/B тестування. Серед них такі програмні засоби:

* WebHarvy;
* import.io;
* Google Web Scraper.

Головним недоліком усіх програмних продуктів є неможливість обходу сайту за заданими посиланнями.

* 1. Огляд роботи аналогів, їх переваг та недоліків

Першим популярним інструментом для A/B тестування є Convert. Сервіс дозволяє наступні види тестування: А / В, спліт-тестування, багатофакторне тестування.

Сервіс надає непоганий онлайн-редактор, в якому легко без спеціальних знань змінювати елементи сторінки для тестування. Окремо налаштовується тестування під мобільні пристрої.

Варіантів для тестування може бути безліч. Кожному можна привласнити унікальне ім'я і задати свої налаштування. При цьому, є функція копіювання попереднього варіанту для більш швидкого його редагування.

Наступним популярним інструментом є LPgenerator. Інструменти даного тестування представлені, як частина функціоналу по створенню посадкової сторінки. Сервіс дозволяє лише A / B-тестування.

Присутній онлайн-редактор, з великою кількістю шаблонів для генерації посадкової сторінки з нуля. Однак, власну сторінку редагувати в ньому не можна.

Одним з найсильніших за функціоналом сервісів для тестування є Optimizely. Підтримує А / В тестування і багатофакторний аналіз.

При створенні проекту тестування пропонується вибрати одну з трьох платформ на якій буде проводитися аналіз.

Зручний онлайн-редактор сторінки дає можливість змінювати варіанти сайту для тестування прямо в вікні сервісу.

Так само є можливість налаштування цілей тестування.

Після налаштування цілей і запуску кампанії, реалізоване відстеження тестування в режимі реального часу.

Досить потужним сервісом для тестування є AB TASTY [7]. Доступні наступні види тестування: тестування захищених сторінок, тестування перенаправлення, багатосторінкові тестування, тунелі.

Зручна настройка сегментації і персоналізаціі. Є онлайн редактор, як графічний, так і HTML, CSS, JavaScript. Легке налаштування кампанії і наочне відстеження тестування і досягнення цілей в реальному часі. Сервіс дозволяє відслідковувати дійсно величезну кількість показників.

Наступний сервіс Unbounce - по суті є західним аналогом LPgenerator. Являє собою потужний інструмент для створення landing сторінок, який включає в себе А/В тестування. Однак, через це власний, вже готовий сайт, протестувати вийде. Є графічний редактор з величезною базою редагованих шаблонів посадкових сторінок.

Сервіс VWO має весь функціонал, який може знадобитися для тестування сайту. Види тестування: А / В, спліт-тестування, багатофакторне тестування.

Налаштування цілей, дає можливість автоматизувати тестування безлічі варіантів до досягнення певних показників. Кампанію можна налаштувати як для десктопних комп’ютерів, так і для версії на будь-якому з мобільних пристроїв. Зручний онлайн-редактор і швидке налаштування величезної кількості варіантів.

Після запуску кампанії відразу можна почати відстежувати тестування в наочних звітах.

Так само сервіс пропонує «Ідеї» - це готові набори визначені встановлення і цілі для тестування. При чому, сервіс аналізує ваш сайт і видає вам ідеї, які найбільш підійдуть саме для вашого сайту.

Наступним сервісом є сайт Clickthroo, який дозволяє тільки А/В тестування.

Є онлайн редактор і набори шаблонів. Можливо відстеження тестування в реальному часі, настройка гео-таргетингу і відстеження всієї аналітики.

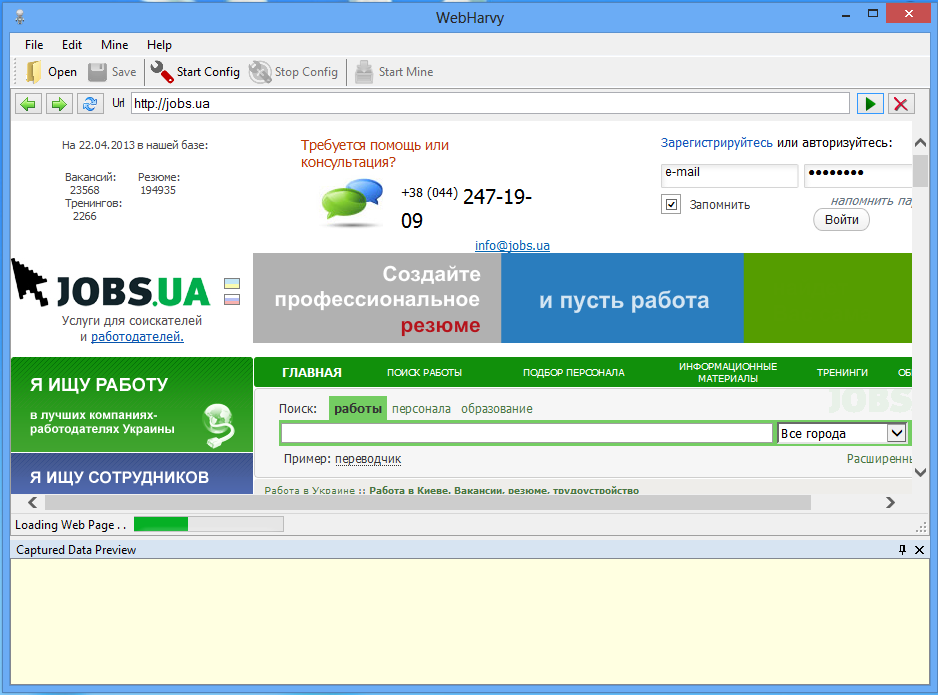


Рисунок 2.1 – Діалогове вікно налаштувань для скрапінгу даних WebHarvy

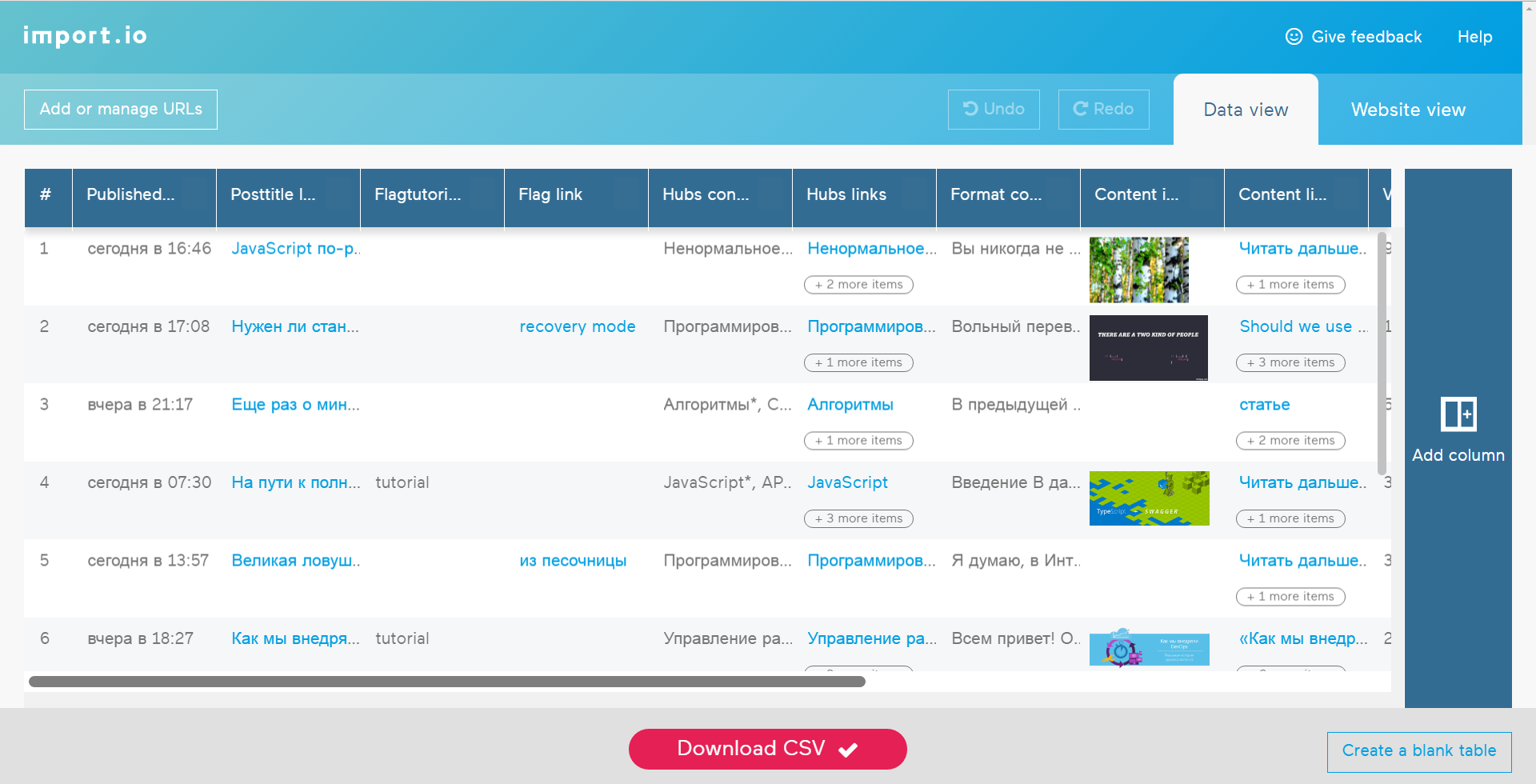


Рисунок 2.2 – Діалогове вікно налаштувань import.io

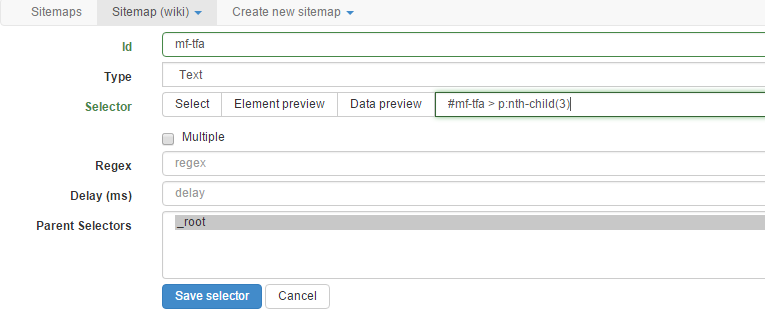


Рисунок 2.2 – Діалогове вікно створення sitemap

1. МЕТОДИ РЕАЛІЗАЦІЇ ПРОГРАМНОЇ СИСТЕМИ

У процесі розробки системи агрегації контенту використовувались наступні методи: методи скрапінгу HTML сторінки, методи маркетингового дослідження, зокрема метод A/B тестування.

* 1. Скрапінг HTML сторінки

Скрапінг являє собою метод вилучення інформації з веб-сайтів. Як правило, це досягається за рахунок або безпосередньо реалізації протоколу передачі гіпертексту (на якому базується Web) або за допомогою веб-браузера.

Скрапінг тісно пов'язаний з веб-індексацією, яка індексує інформацію в Інтернеті за допомогою бота і є універсальною методикою, прийнятою більшістю пошукових систем. На противагу цьому, скрапінг більше фокусується на перетворенні неструктурованих даних в мережі, як правило, в форматі HTML, в структуровані дані, які можуть зберігатися і аналізуватися в центральній локальній базі даних або таблиці. Скрапінг також пов'язаний з веб-автоматизацією, яка імітує людський перегляд інформації в інтернеті за допомогою спеціалізованого програмного забезпечення.

Існують наступні методи скрапінгу:

1. Копіювання та вставка інформації користувачем: іноді навіть найкраща технологія скрапінгу не може замінити людину, особливо, коли на сайті явно встановлений бар’єр для запобігання автоматизованого скрапінгу.
2. Регулярні вирази: простий, але ефективний підхід для отримання інформації з веб-сторінок за допомогою регулярних виразів.
3. HTTP програмування: статичні і динамічні веб-сторінки можуть бути отримані шляхом HTTP-запитів на віддалений веб-сервер.
4. HTML парсинг: багато веб-сайтів мають великі колекції сторінок, створених динамічно з використанням бази даних, як структурованого джерела. Цей метод визначає структуру сторінки та створює об’єкти, виходячи з цієї структури.
5. DOM (Document Object Model) аналіз: вбудовуючись у повноцінний веб-браузер, наприклад, Internet Explorer або Mozilla, програма може отримати динамічний контент, згенерований на стороні клієнта. Ці елементи управління браузера також аналізують веб-сторінки в DOM дереві, на основі яких програма може отримати частини сторінок.
6. Веб-скрапінг: є багато інструментів, які можуть бути використані для веб-скрапінгу. Ці інструменти надають інтерфейс для того, щоб точно визначити структуру сторінки, а також зберегти вибрані структури у базі даних.
7. Аналіз веб-сторінки з використанням комп’ютерного зору або комп’ютерного бачення: є програмне забезпечення яке працює за допомогою машинного навчання і комп'ютерного зору, яке намагається ідентифікувати і отримувати інформацію з веб-сторінок шляхом інтерпретації сторінок візуально так, як це зробив би користувач.
   1. Методи маркетингових досліджень

Найважливішими методами проведення маркетингових досліджень є: спостереження, опитування, експеримент і визначена група кількісних методів.

Спостереження - це метод збирання первинної маркетингової інформації про об'єкт, що досліджується. Спостереження здійснюється за обраними групами людей, ситуаціями.

До проведення спостережень застосовують різноманітні підходи: пряме і побічне спостереження, відкрите і приховане, структуроване і неструктуроване, за допомогою людини або механічних засобів.

Пряме спостереження передбачає безпосереднє спостереження за об'єктом, наприклад за покупцем у магазині. При проведенні побічного спостереження вивчаються результати певної поведінки, а не сама поведінка.

Відкрите спостереження передбачає, що люди знають про те, що за ними спостерігають. Необхідно пам'ятати, що присутність спостерігачів певною мірою впливає на поведінку людей, за якими спостерігають, тому необхідно прагнути звести його до мінімуму.

Найбільш прийнятим в таких умовах є приховане спостереження, коли досліджуваний об'єкт не знає, що за ним спостерігають.

При проведенні структурованого спостереження наперед визначається, що підлягає спостереженню. Всі інші види спостереження ігноруються. Структуроване спостереження використовується для перевірки результатів, отриманих іншими методами, та їх уточнення.

Коли проводиться неструктуроване спостереження, спостерігач фіксує в досліджуваному епізоді всі види поведінки. Такий тип спостереження часто використовується при проведенні розвідувальних досліджень.

Наприклад, компанія, яка випускає будівельний інструмент, може направити своїх співробітників для вивчення направлень і частоти використання певного інструменту при будівництві будинків. Результати спостережень використовуються при вдосконаленні цього інструменту.

Щоб отримати необхідну й об'єктивну інформацію і не пропустити будь-яких важливих факторів, необхідно наперед сумлінно розробити план спостережень. У ньому необхідно передбачити терміни, засоби збирання інформації, відповідальних осіб.

Найбільш важливими етапами спостережень є:

* визначення цілі;
* визначення суб'єкта і об'єкта;
* вибір методів спостереження;
* підготовка технічних документів (тиражування карток, інструкцій для споживачів, підготовка письмових приладів, технічного обладнання).

У процесі реалізації плану здійснюється проведення спостережень, збирання даних, накопичення інформації. В першу чергу необхідно правильно підібрати спостерігачів. Найважливіші вимоги, що висуваються до спостерігача - добросовісність, увага, терплячість.

Фіксація результатів спостереження здійснюється у вигляді:

* короткочасного запису, що проводиться по "гарячих слідах";
* реєстрації інформації;
* щоденника, до якого систематично вносяться всі необхідні відомості;
* фото-, відео-, звукозаписів.

Контроль спостереження можна здійснювати одним з методів нижче:

* проведення розмови з учасниками ситуації;
* звернення до документів, пов'язаних з певною подією;
* повторення спостереження.

Звіт про спостереження містить:

* документацію про час, місце і обставини проведення спостереження;
* інформацію про роль спостерігача і ефективність обраних методів;
* характеристику осіб, за якими проводиться спостереження;
* власні інтерпретації спостерігача;
* оцінку надійності отриманих результатів.

Одним з найважливіших методів проведення маркетингових досліджень, поряд з експериментом або в його рамках, залишається опитування населення і спеціалістів. Шляхом опитування здійснюється систематичне збирання інформації про осіб, яких опитують. Опитування здійснюється за допомогою особистих контактів, телефону, або поштою.

Особисте опитування проводиться в процесі бесіди безпосередньо з опитуваним і дає змогу отримати найбільш обширну і достовірну інформацію. Персональне інтерв'ю має свої плюси і мінуси.

Переваги: більш репрезентативна вибірка, більшу кількість питань може бути задано, можлива оперативна перевірка. Недоліки: дорого, помилки, викликані особистістю інтерв'юера або обманом зі сторони опитуваного.

Нерідко метод опитування використовується для отримування результатів впливу реклами. Інтерв'юер, використовуючи тести на впізнання реклами, розглядаючи разом з досліджуваним спеціалістом раніше бачений ним журнал, опитує, які рекламні оголошення є знайомими. Більш достовірні дані про ефективність реклами дають тести на запам'ятовування: людина, яка заявила, що вона уже знайома з представленою їй на даний момент торгівельною маркою, або іншим елементом реклами, повинна розповісти про її зміст більш детально. Так можна оцінити якість запам'ятовування, міру її достовірності. Але і такі тести ще не дають підстав для висновків про готовність до укладання угоди, до покупки товару.

Ще один крок вперед порівняно з тестами - опитування на тему запропонованої реклами. Можливі прості питання типу: "Чи подобається вам ця реклама?", "Чи цікава вона для вас?", "Чи довіряєте ви їй?". Можливі й побічні методи. Наприклад, людину наперед просять вибрати із ряду предметів той, який їй найбільше подобається. Зміна вибору на користь про рекламованого предмета - критерій оцінки ефективності реклами.

Для оцінки дієвості реклами використовуються і проективні методи. Один з них - тест на словесні асоціації: якщо ті, або інші слова з проекту рекламного повідомлення, що надається у випадковій послідовності серед інших слів, викликають у людей невірні або небажані асоціації, то вони виключаються із реклами. Тести на сприйняття допомагають визначити ступінь зрозумілості, легкість зчитування основних змістових блоків реклами.

Відомі також тести, опитування, що мають своєю ціллю аналіз іміджу - ставлення населення до певного продукту або підприємства, що здійснює опитування. Прагнення підприємств і реальність у цьому випадку найбільш різко розходяться. Підприємства часто бувають шоковані тим, що думає населення насправді, тому такі тести і опитування найбільш повчальні.

Детально і диференційовано оцінити якість і ефективність рекламного повідомлення дає можливість анкетне опитування, до якого можна залучити як безпосередньо споживачів, так і спеціалістів- експертів.

Опитування по телефону відносно просте, дешеве і не займає багато часу.

Недоліки - певна частина населення може не мати телефону.

Найбільш повну і добротну інформацію можна отримати експериментальним шляхом. Причому, процес експерименту проводиться ніби в три етапи, рівнозначними самостійними видами опитування: експлораторне (розвідувальне), дескриптивне (описувальне), каузальне (експериментальне) дослідження.

Зазначимо, що різниця між категоріями досліджень має відносний характер, а самі категорії можна розцінити як ступені деякого єдиного процесу. Причому, кожний вид досліджень відповідає певному типу маркетингових проблем.

Отже, розглянувши методи маркетингового дослідження можна сказати, що A/B тестування задовольняє багатьом потребам сучасного бізнесу. Воно є одним з методів спостереження, при якому досліджувана група людей не знає про проведення дослідження. A/B-тест повинен мати певний результат, який піддається вимірюванню, наприклад, кількість продажів, число людей, реєстрації, кількість натискань на той чи інший елемент. Багато компаній використовують «експериментальний» підхід до прийняття маркетингових рішень. Це стає все більш поширеною практикою, і досвід росте в цій галузі. Є багато A/B-тестувань тематичних досліджень, які показують, що практика тестування стає все більш популярною також у малому і середньому бізнесі.

1. ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ СИСТЕМИ

Система A/B тестування розроблена в середовищі Visual Studio 2017 з використанням технологій ASP.NET Web API та Angular.

* 1. Архітектура системи

Система складається з двох основних компонентів: компонент, що виконує аналіз даних, та веб-інтерфейс. Взаємодія цих компонентів та основних модулів представлена на наступній діаграмі (рис. 4.1).

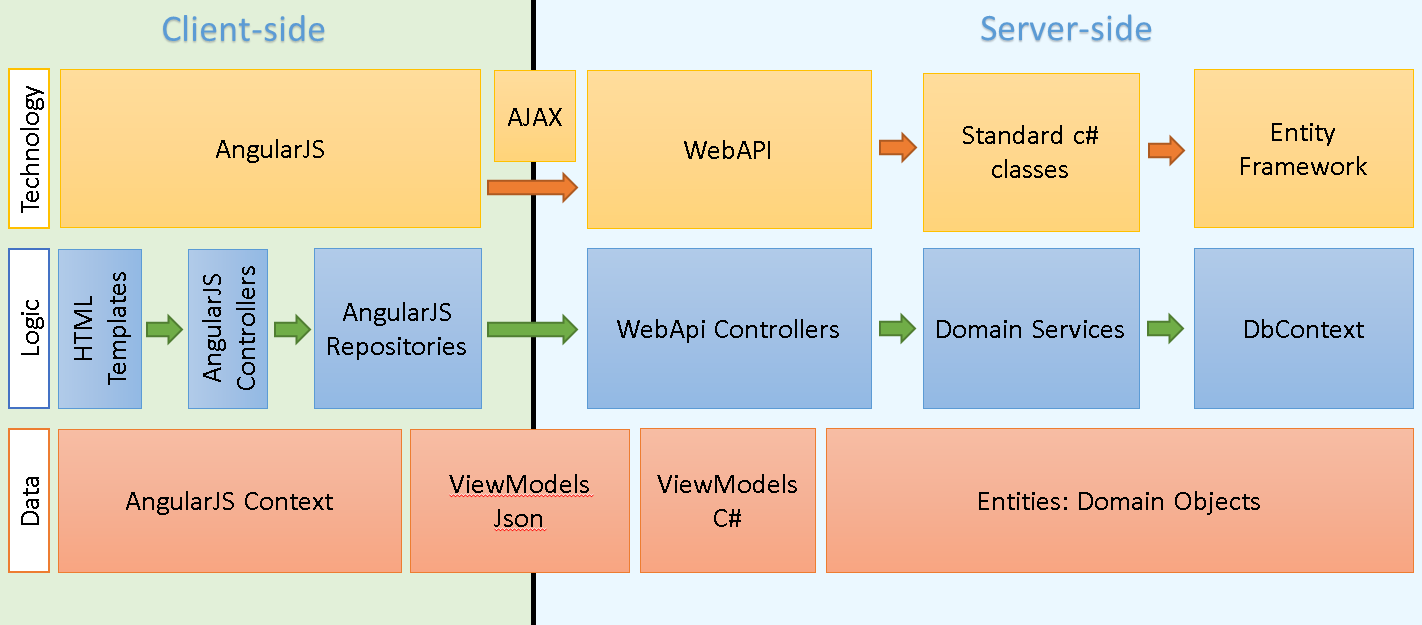


Рисунок 4.1 Архітектура та взаємодія основних модулів системи

Обидва модулі є незалежними один від одного, але й в той самий час кожен з них виконує функцію, яка відповідає за надання даних для іншого модуля.

Система включає низку модулів, які складають основний функціонал:

* “HomeController” — основний модуль системи. Відповідає за створення та налаштування задач A/B тестування.
* “AccountContoller” — модуль, що відповідає за збереження даних користувачів системи, надає можливість реєстрації та авторизації.
* “ABTestContoller” — модуль, що відповідає за механізм відображення А та В елементів на сторінці клієнта, а також сбір та аналіз дій відвідувачів сайту кліента.
* “ScrappingController” — модуль, який дозволяє отримати сторінку користувача, та відобразити її у первинному вигляді.
* “ABContext” — модуль, що надає можливість працювати з БД.
  + 1. **Етапи роботи системи.**

На рисунку 4.2 зображені функції користувача системи та функції самої системи.

Під час побудови задачі A/B тестування відбувається наступний процес:

Користувач надає сторінку, для якої будуть відбуватися створення налаштування задачі A/B тестування.

Система аналізує сторінку, та надає можливість створити нову задачу.

Після налаштувань, користувач зберігає задачу. Система створює всі необхідні початкові дані для початка A/B тестування.

Система генерує файл, який користувач повинен розмістити на досліджуваній сторінці.

Після того, як користувач вставив файл на свою сторінку система постійно відслідковує дії відвідувача сайту, та зберігає ці дані для подальшого відображення клієнту.

* + 1. **Опис Web-інтерфейсу системи A/B тестування.**

Web-інтерфейс використовує бібліотеку KnockoutJs [15] та ajax технології для того, щоб користувач працював з інтерфейсом без перерв на перезавантаження сторінок при надходженні нового запиту на роботу. Для роботи з DOM використано JavaScript бібліотеку jQuery [16]. Задачами даного модуля є:

* формування завдання агрегації даних;
* розміщення завдання до черги у базу даних;
* повернення результатів агрегації.

За допомогою даного модулю користувач має можливість створити нове завдання на парсинг, перезапустити вже завершене та отримати результати парсингу.



Рисунок 4.2 – Діаграма прецедентів

* + 1. **Архітектура API системи A/B тестування.**

Консольний додаток системи агрегації складається з двох модулів: модуль скрапінгу та модуль роботи с базою даних. Модуль скрапінгу, використовуючи дані з БД, оброблює та передає структуровані дані в модуль роботи с базою даних, а той в свою чергу забезпечує роботу з БД (збереження, видалення, отримання даних).

* + 1. **Опис бази даних системи A/B тестування.**

Базою даних для системи агрегації контенту з різних тематичних сайтів була обрана СКБД MongoDb, оскільки вона дозволяє створювати документи динамічної структури в одній колекції.

Документо-орієнтована система керування базами даних — система керування базами даних, спеціально призначена для зберігання ієрархічних структур даних (документів). В основі документо-орієнтованих СКБД лежать документні сховища (колекції) котрі мають структуру дерева. Структура дерева починається з кореневого вузла і може містити кілька внутрішніх вузлів.

База даних системи агрегації контенту з різних тематичних сайтів складається з трьох колекцій: Tasks, Data, Users (рисунок 4.4).



Рисунок 4.3 – Архітектура консольного додатку

На рисунку 4.5 представлений приклад даних записаний до колекції “Data”.

Ця колекція складається з декількох обов’язкових і необмеженої кількості динамічних полів (кількість полів залежить від структури, яку вводить користувач). Обовязковими є:

* «\_id» – ідентифікатор документа;
* «TaskId» – ідентифікатор завдання, до якого прив’язані дані у розпарсеному документі;
* <Url> - посилання на сторінку, з якої отримані данні;
* всі інші поля є динамічними і залежать від вводу користувача.

Всі дані, які були отримані з усіх сайтів, зберігаються в даній колекції, що дає додаткові можливості створення статистики по всім даним.

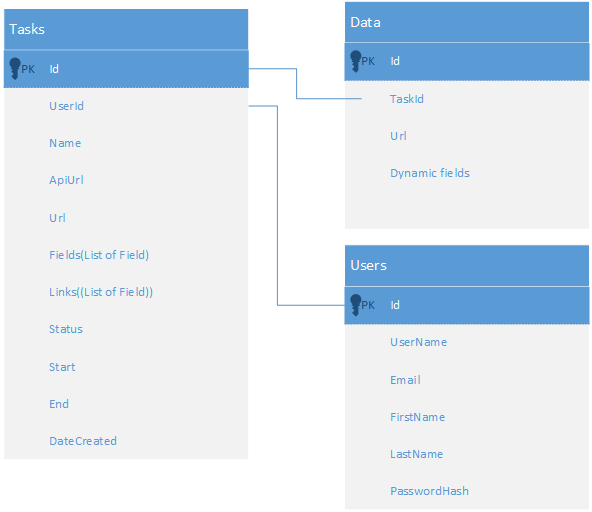


Рисунок 4.4 – Схема бази даних

На рисунку 4.6 представлена ключова колекція, яка має назву «Tasks».

Ця колекція зберігає в собі завдання на роботу по агрегації даних з конкретного сайту. Вона складається з наступних полів:

* «\_id» – ідентифікатор документа;
* «UserId» – ідентифікатор користувача, який створив завдання;
* «Name» – назва завдання;
* «ApiUrl» – поле, яке містить посилання на сторінку де можливо переглянути розпарсені дані;
* «Url» – стартове посилання;
* «Fields» – складне поле, яке містить масив об’єктів. Ці об’єкти описують поле, яке буде додано до колекції «Data». Один об’єкт складається з таких простих полів: Name – назва поля, яке з’явиться у колекції «Data», Selector – jquery селектор для текстового блоку на сторінці, з якого будуть вибрані дані та поміщені до колекції «Data»;
* «LinkSelectors» – складне поле, яке містить об’єкти селекторів для посилань. Завдяки цим об’єктам скрапер переходить по конкретним посиланням;
* «Status» – поле, яке містить статус завдання;
* «Start» – поле, яке містить дату і час старту завдання;
* «End» – поле, яке містить дату і час кінця завдання;
* «DateCreated» – поле, яке містить дату і час створення завдання.

На рисунку 4.7 представлена колекція, яка має назву «Users».

Ця колекція зберігає інформацію про користувача для його аутентифікації. Вона складається з таких полів:

* «\_id» – ідентифікатор користувача;
* «UserName» – логін користувача, який вводиться при вході у застосунок;
* «SecurityStamp» – згенерований код захисту, яке повинен змінюватись при будь-якій зміні даних користавача;
* «Email» – поштова адреса користувача;
* «EmailConfirmed» – поле, яке свідчить яке свідчить про факт підтвердження поштової адреса;
* «PhoneNumber» – номер телефону;
* «PhoneNumberConfirmed» – поле, яке свідчить про факт підтвердження номер телефону;
* «TwoFactorEnabled» – визначає чи увімкнена двофакторна аутентифікація;
* «LockoutEndDateUtc» – значення дати й часу, коли разблокування закінчиться; в будь-який час в минулому вважається не заблокований;
* «LockoutEnabled» – поле, яке свідчить чи заблокований користувач;
* «AccessFailedCount» – лічильник невдачних спроб логіну;
* «Roles» – список ролей, які виконує користувач;
* «PasswordHash» – пароль у захешованому вигляді;
* «Logins» – список, який містить інформацію про логіни користувача;
* «Claims» – список заявок;
* «FirstName» – ім’я користувача;
* «LastName» – прізвище користувача.

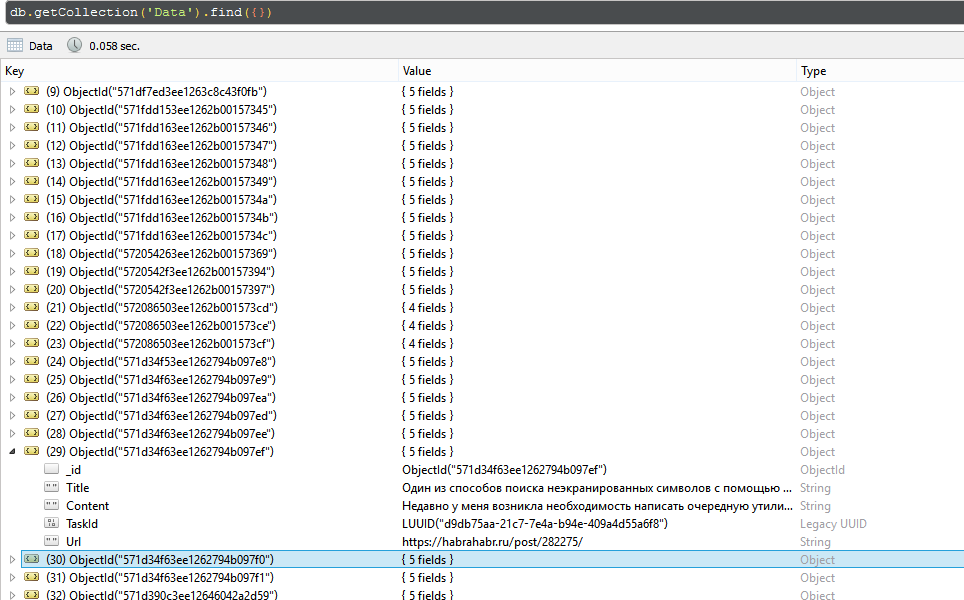


Рисунок 4.5 – Дані у колекції “Data”

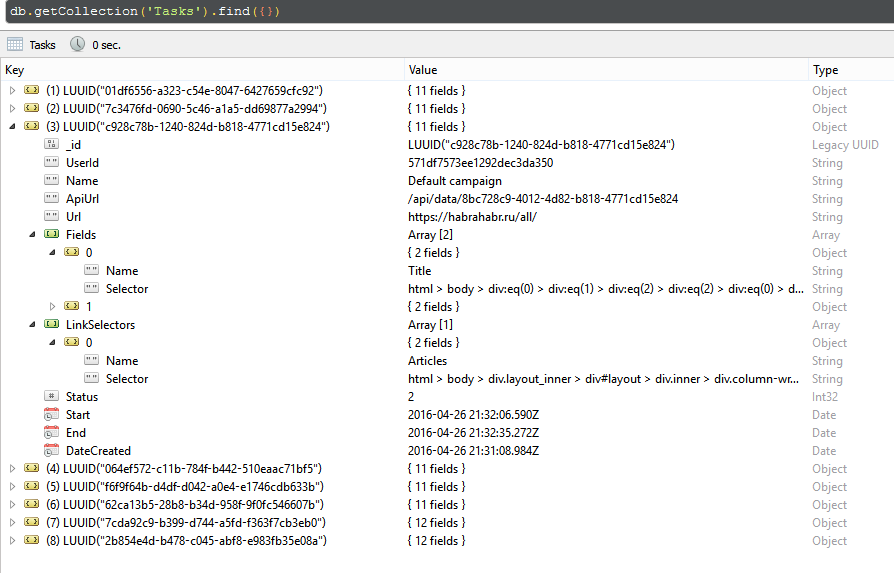
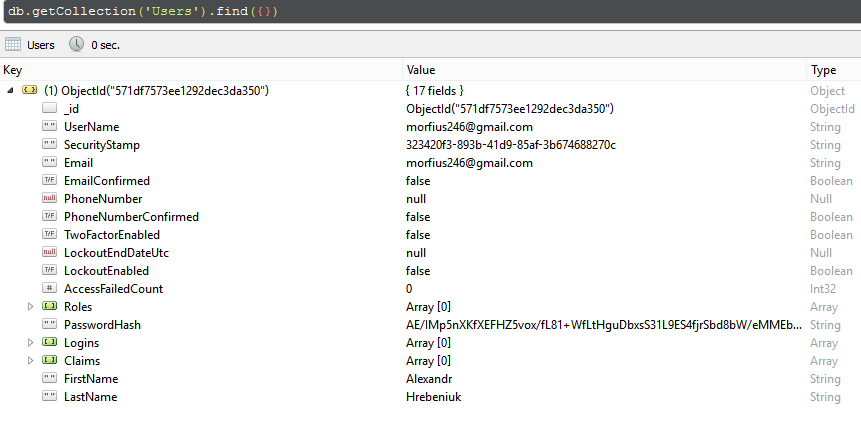


Рисунок 4.6 – Дані у колекції “Tasks”

 Рисунок 4.7 – Дані у колекції «Users»

* 1. Програмні засоби розроблення системи

Для написання програмного продукту використано наступні методи та засоби розробки.

Платформа ASP.NET Core собою представляє технологію від компанії Microsoft, призначену для створення різного роду веб-додатків: від невеликих веб-сайтів до великих веб-порталів і веб-сервісів.

З одного боку, ASP.NET Core є продовженням розвитку платформи ASP.NET. Але з іншого боку, це не просто черговий реліз. Вихід ASP.NET Core фактично означає революцію всієї платформи, її якісна зміна.

ASP.NET Core є кросплатформенним opensource-проектом. Весь код проекту доступний на GitHub, і тому кожен може бачити реалізацію всього функціоналу цієї платформи і робити до неї зміни, якщо це буде потрібно.

Додаток ASP.NET Core може працювати з двома виконуваними середовищами: .NET Core і з повною версією фреймворка .NET [9].

.NET Core представляє модульну крос-платформену виконуючу середу, яка спрощує розгортання програми на мові C# [10]. .NET Core програми можна запускати не тільки на ОС Windows, але і на Linux і MacOS.

Завдяки модульності всі необхідні компоненти веб-додатки можуть завантажуватися як окремі модулі через пакетний менеджер Nuget.

ASP.NET Core - більш оптимізована для використання в хмарних сервісах і має вбудовану підтримку впровадження залежностей.

Angular [11] — JavaScript-фреймворк з відкритим програмним кодом, який розробляє Google. Призначений для розробки односторінкових додатків, що складаються з однієї HTML сторінки з CSS і JavaScript. Його мета — розширення браузерних застосунків на основі шаблону Модель-вид-контролер (MVC), а також спрощення їх тестування та розробки.

Фреймворк працює зі сторінкою HTML, що містить додаткові атрибути і пов'язує області вводу або виводу сторінки з моделлю, яка є звичайними змінними JavaScript. Значення цих змінних задаються вручну або отримуються зі статичних або динамічних JSON-даних.

Angular спроектований з переконанням, що декларативне програмування найкраще пасує для побудови інтерфейсів користувача та опису програмних компонентів, в той час як імперативне програмування пасує для опису бізнес-логіки. Фреймворк адаптує та розширює традиційний HTML, щоб забезпечити двосторонню прив'язку даних для динамічного контенту, що дозволяє автоматично синхронізувати модель та вид. У результаті AngularJS зменшує роль DOM-маніпуляцій з метою підвищення продуктивності і спрощення тестування.

* + 1. **ASP.NET Web API.**

ASP.NET — технологія створення веб-застосунків і веб-сервісів від компанії Майкрософт. Вона є складовою частиною платформи Microsoft.NET і розвитком старішої технології Microsoft ASP. У цей час останньою версією цієї технології є ASP.NET 5.

ASP.NET зовні багато в чому зберігає схожість із старішою технологією ASP, що дозволяє розробникам відносно легко перейти на ASP.NET. У той же час внутрішній устрій ASP.NET істотно відрізняється від ASP, оскільки вона заснована на платформі. NET і, отже, використовує всі нові можливості, що надаються цією платформою.

У якості мови програмування для реалізації системи було взято мову С#, що має підтримку усіх необхідних технік програмування. Мова С# є універсальною та високорівневою мовою програмування, придатною для створення програм будь-якої складності. Вона є однією з найбільш перспективних у світі мов програмування, яка на практиці підтвердила свою надійність та масштабованість. Ці причини є основними факторами, що визначили вибір цієї мови.

Таким чином, в огляді методів реалізації було визначено та обґрунтовано основні методи та технології, які будуть використані при розробці програмного продукту: модульне програмування на мові С# з використанням об’єктно-орієнтованої техніки, а також застосування знань з інформаційних систем.

ASP.NET MVC Framework — фреймворк для створення веб-застосунків, який реалізує шаблон Model-view-controller. Цей фреймворк доданий Microsoft в ASP.NET.

Шаблон архітектури Model-View-Controller (MVC) розділяє додаток на три основні компоненти: модель, вигляд і контролер. Платформа ASP.NET MVC являє собою альтернативу схемі веб-форм ASP.NET при створенні веб-додатків. Платформа ASP.NET MVC є платформою відображення з широкими можливостями тестування і, подібно до додатків на основі веб-форм, інтегрована з існуючими функціями ASP.NET, наприклад з головними сторінками і перевіркою достовірності на основі членства. Платформа MVC визначається в збірці System.Web.Mvc.

MVC являє собою стандартний шаблон розробки, знайомий багатьом фахівцям. Деякі типи веб-додатків мають переваги при створенні на платформі MVC. Для інших може бути доцільно використання традиційної схеми програми ASP.NET, заснованої на веб-формах і зворотного передачі. У деяких випадках можливе поєднання двох підходів: застосування однієї схеми не виключає використання іншого. Шаблон розробки MVC представлений на рисунку 4.8.

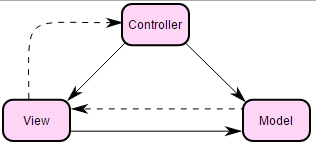


Рисунок 4.8 – Шаблон розробки MVC

* + 1. **MS SQL.**

MongoDB [18] — документо-орієнтована система керування базами даних (СКБД) з відкритим вихідним кодом, яка не потребує опису схеми таблиць. MongoDB займає нішу між швидкими і масштабованими системами, що оперують даними у форматі ключ/значення, і реляційними СКБД, функціональними і зручними у формуванні запитів.

Код MongoDB написаний на мовою C++ і поширюється в рамках ліцензії AGPLv3.

MongoDB підтримує зберігання документів в JSON-подібному форматі, має досить гнучку мову для формування запитів, може створювати індекси для різних збережених атрибутів, ефективно забезпечує зберігання великих бінарних об'єктів, підтримує журналювання операцій зі зміни і додавання даних в БД, може працювати відповідно до парадигми Map/Reduce, підтримує реплікацію і побудову відмовостійких конфігурацій. У MongoDB є вбудовані засоби із забезпечення шардінга (розподіл набору даних по серверах на основі певного ключа), комбінуючи який реплікацією даних можна побудувати горизонтально масштабований кластер зберігання, в якому відсутня єдина точка відмови (збій будь-якого вузла не позначається на роботі БД), підтримується автоматичне відновлення після збою і перенесення навантаження з вузла, який вийшов з ладу. Розширення кластера або перетворення одного сервера в кластер проводиться без зупинки роботи БД простим додаванням нових машин.

Основні можливості MongoDB Server (рисунок 4.9):

─ документо-орієнтоване сховище (проста та потужна JSON-подібна схема даних);

─ досить гнучка мова для формування запитів;

─ динамічні запити;

─ повна підтримка індексів;

─ профілювання запитів;

─ швидкі оновлення ”на місці”;

─ ефективне зберігання двійкових даних великих обсягів;

─ журналювання операцій, що модифікують дані в БД;

─ підтримка відмовостійкості і масштабованості: асинхронна реплікація, набір реплік і шардінг;

─ може працювати відповідно до парадигми MapReduce [19];

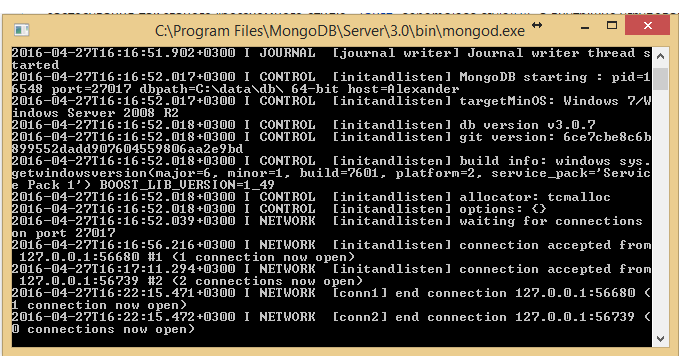


Рисунок 4.9 – Стартове вікно MongoDB Server

* + 1. **Angular.**

KnockoutJS являє собою автономну JavaScript реалізацію патерну Model-View-ViewModel з власними шаблонами. Основними принципи:

* чіткий поділ між даними застосунку, шаблонами, та даними для відображення;
* наявність чітко визначеного шару спеціалізованого коду для управління відносинами між шаблонами.

Останній використовує вбудоване управління подіями мови Javascript.

Ці можливості KnockoutJS оптимізують та спрощують програмування складних відносини між шаблонами.

KnockoutJS розроблений та підтримується Стівом Сандерсоном, працівником Microsoft. Автор підкреслює, що це особистий проект з відкритим кодом, і не є продуктом Microsoft.

* + 1. **JQuery.**

jQuery – популярна JavaScript-бібліотека з відкритим вихідним кодом. Вона була представлена у січні 2006 року у BarCamp NYC Джоном Ресіґом (John Resig). Згідно з дослідженнями організації W3Techs, JQuery використовується понад половиною від мільйона найвідвідуваніших сайтів. jQuery є найпопулярнішою бібліотекою JavaScript, яка посилено використовується на сьогоднішній день.

jQuery є вільним програмним забезпеченням під ліцензією MIT (до вересня 2012 було подвійне ліцензування під MIT та GNU General Public License другої версії).

Синтаксис jQuery розроблений, щоб зробити орієнтування у навігації зручнішим завдяки вибору елементів DOM, створенню анімації, обробки подій, і розробки AJAX-застосунків. jQuery також надає можливості для розробників, для створення плагінів у верхній частині бібліотеки JavaScript. Використовуючи ці об'єкти, розробники можуть створювати абстракції для низькорівневої взаємодії та створювати анімацію для ефектів високого рівня. Це сприяє створенню потужних і динамічних веб-сторінок.

1. МЕТОДИКА РОБОТИ КОРИСТУВАЧА З СИСТЕМОЮ

Для того, щоб почати працювати з системою агрегації контенту з різних тематичних сайтів потрібно ознайомитись з системними вимогами, а також виконати ряд операцій, щоб встановити її на сервері.

* 1. Системні вимоги та інсталяція програмного забезпечення

Перед початком інсталяції програмного забезпечення треба впевнитись, що комп’ютер відповідає заданим системним вимогам. Вимоги формувались у відповідності до середньостатистичних, тобто таких, які відповідають можливостям більшості підприємств.

* + 1. **Апаратні та програмні вимоги.**

Апаратні вимоги для коректної роботи програмного забезпечення для комп’ютера:

* процесор: Pentium з тактовою частотою не менше 1Ггц або аналогічний (рекомендується Pentium з тактовою частотою 2 Ггц або аналогічний);
* ОЗУ: не менше 256 МБ (рекомендується 1024 МБ);
* екран: 800 x 600, 256 кольорів (рекомендується 1280 x 768, high color (32-розрядний колір));
* Програмні вимоги для коректної роботи програмного забезпечення для комп’ютера:
* встановлена ОС Windows 7 / Windows 8 / Windows 10;
* .NET Framework 4.0 або вище;
* встановлена база даних MongoDb.

Система коректно функціонує на комп'ютерах з операційними системами:

* Windows XP;
* Windows Server 2003;
* Windows Vista;
* Windows Server 2008;
* Windows 7;
* Windows 8;
* Windows Server 2012;
* Windows 10.
  + 1. **Інсталяція програмного забезпечення.**

Приступаючи до роботи з програмою, користувачу насамперед необхідно включити «Служби IIS» (Internet Information Services). Для цього необхідно зробити наступні дії:

* закрийте всі програми, які працюють в даний час;
* відкрийте панель керування та двічі клацніть «Установка й видалення програм»;
* на лівій панелі клацніть «Додавання та видалення компонентів Windows». Запуститься майстер компонентів Windows;
* оберіть «Служби IIS» [20] і натисніть кнопку «Далі». Якщо прапорець вже встановлений, то служби IIS вже встановлені.

Також необхідно встановити базу даних MongoDb:

* завантажуємо архів mongodb з сайту (рисунок 5.1);
* розпаковуємо (наприклад, «C:\Program Files\MongoDB»);
* після встановлення заходимо в папку «C:\Program Files\MongoDB\Server\3.0\bin» і запускаємо файл «mongod.exe». Отримуємо результат представлений на рисунку 5.2.

Після установки «Служб IIS» необхідно створити Web-сайт:

1. Відкрийте «Диспетчер IIS» (рисунок 5.3).
2. На панелі «Підключення» клацніть правою кнопкою миші вузол «Сайти» в дереві, потім виберіть команду «Додати веб-сайт» (рисунок 5.4);
3. У вікні зображеному на рисунку 5.5 необхідно зробити такі дії:
   1. У діалоговому вікні «Додавання веб-сайту» в полі «Ім'я веб-сайту» введіть зрозуміле ім'я веб-сайту.
   2. Клацніть пункт «Вибрати», якщо слід вибрати пул додатків, відмінний від пулу, зазначеного в полі «Пул додатків». У діалоговому вікні «Вибір пулу додатків» виберіть пул додатків в списку «Пул додатків», а потім натисніть кнопку «OK».
   3. У полі «Фізичний шлях» введіть фізичний шлях до папки веб-сайту або натисніть кнопку огляду (...), щоб вибрати папку у файловій системі. Якщо фізичний шлях, вказаний на кроці 5, веде до загального мережного ресурсу, клацніть Підключитися як, щоб вказати облікові дані, що мають дозвіл на доступ до ресурсу з цього шляху. Якщо не використовуються певні облікові дані, виберіть параметр Користувач веб-додатки (наскрізна перевірка справжності) у діалоговому вікні «Підключитися як».
   4. У списку «Тип» виберіть протокол для веб-сайту.
   5. Значення за замовчуванням в полі IP-адреса - Значення не привласнені. Якщо необхідно вказати статичний IP-адресу для веб-сайту, введіть IP-адресу в полі IP-адресу.
   6. У текстовому полі «Порт» введіть номер порту.
   7. Додатково можна ввести ім'я заголовка вузла для веб-сайту в полі «Тема вузла».
   8. Якщо не слід вносити зміни в параметри сайту і слід зробити його негайно доступним, встановіть прапорець Негайний запуск сайту.
4. Натисніть кнопку ОК.
5. Після створення Web-сайту необхідно запустити консольний додаток.

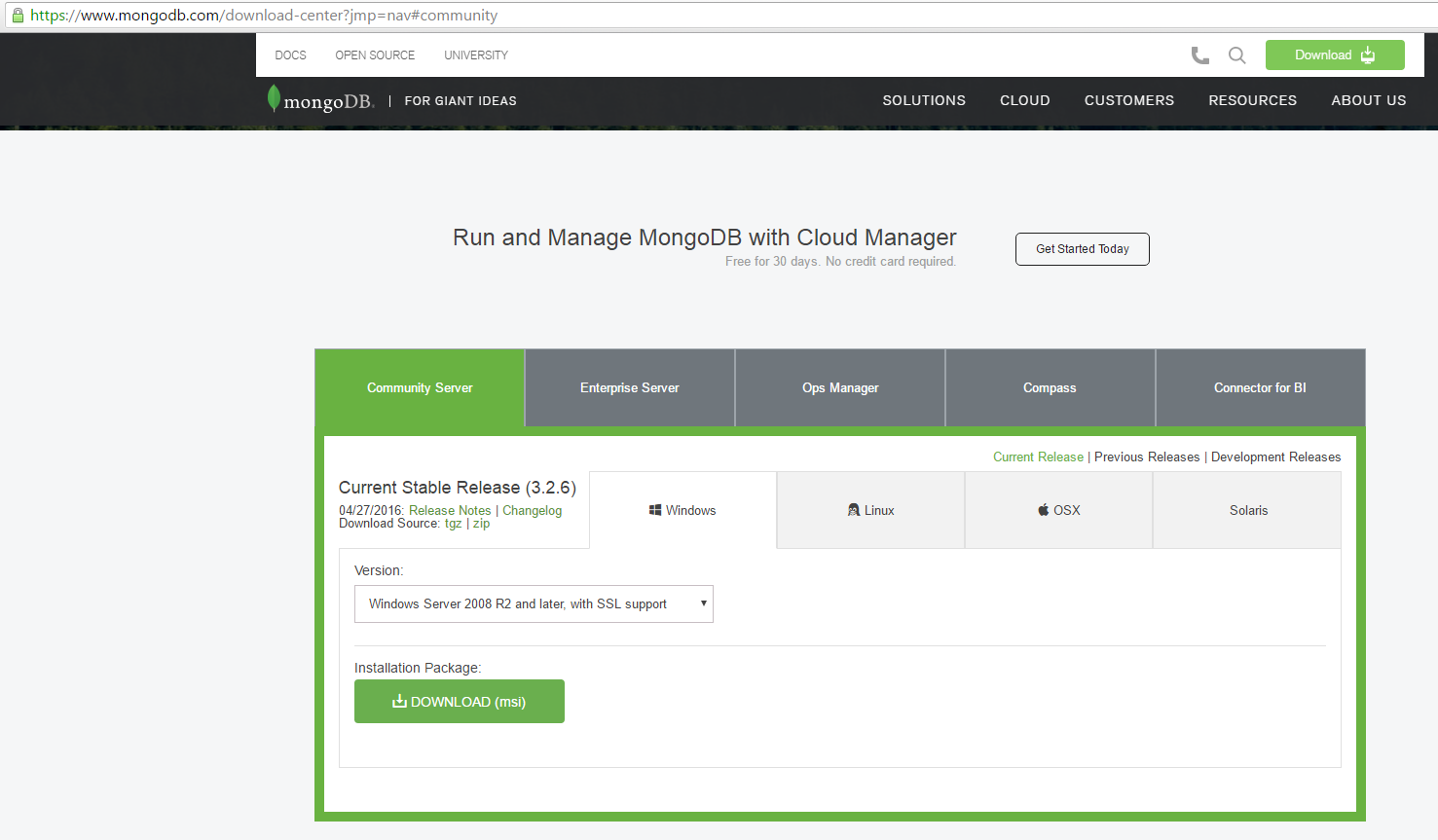


Рисунок 5.1 – Завантаження MongoDb з сайту

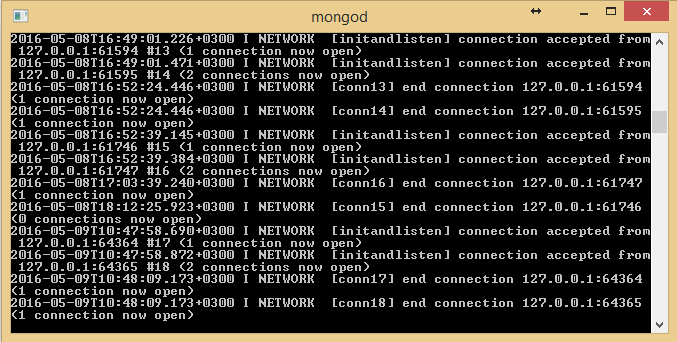


Рисунок 5.2 – Запуск MongoDb

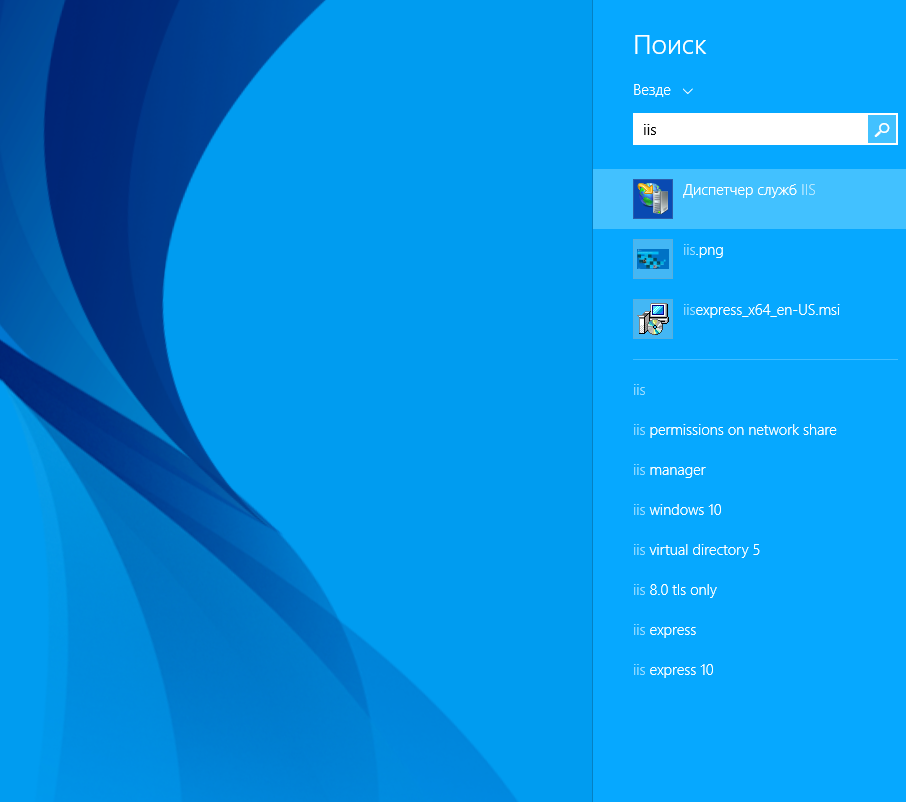


Рисунок 5.3 – Відкриття «Служб IIS»

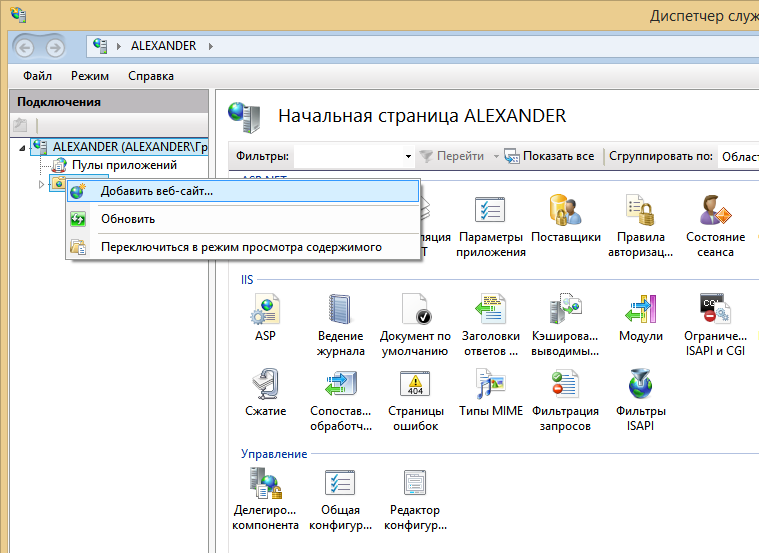


Рисунок 5.4 – Додавання нового сайту

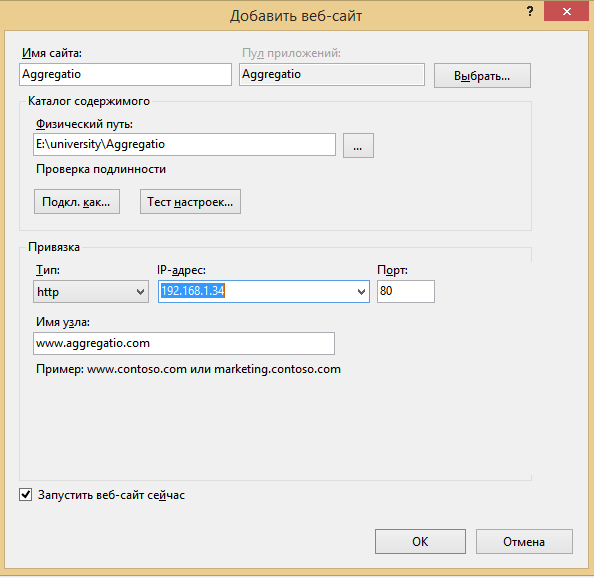


Рисунок 5.5 – Введення даних нового сайту

* 1. Сценарії роботи користувача з системою

Користувач взаємодіє з системою агрегації контенту з різних тематичних сайтів через Web-інтерфейс.

Можливості розробленої системи:

* створення завдання на автоматичну екстракцію даних з завданого Web-ресурсу;
* перегляд списку завдань;
* перегляд отриманих даних;
* видалення завдання з усіма отриманими даними;
* перезапуск завдання;
* доступ до отриманих даних через API.

Сценарій роботи користувача представлено у вигляді демонстрації перерахованих вище можливостей системи.

Створення нового завдання включає в себе такі кроки:

1. Введення назви завдання (рисунок 5.6).

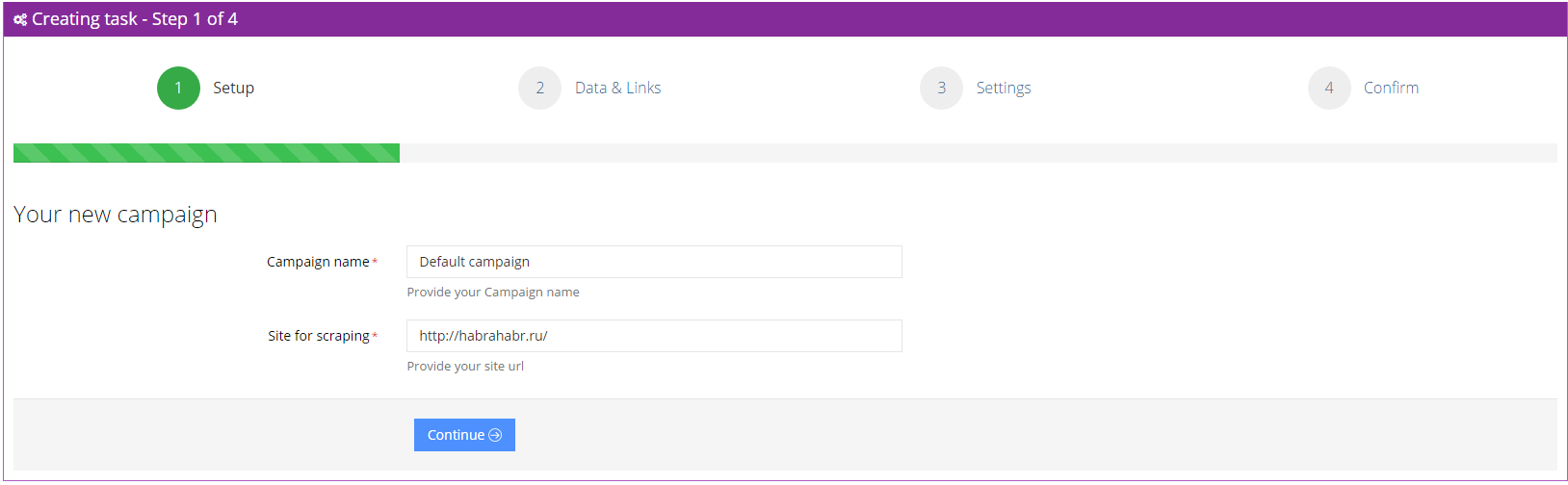


Рисунок 5.6 – Сторінка налаштування конфігурації – перший етап

1. Формування структури даних та введення необхідних селекторів (рисунок 5.7).

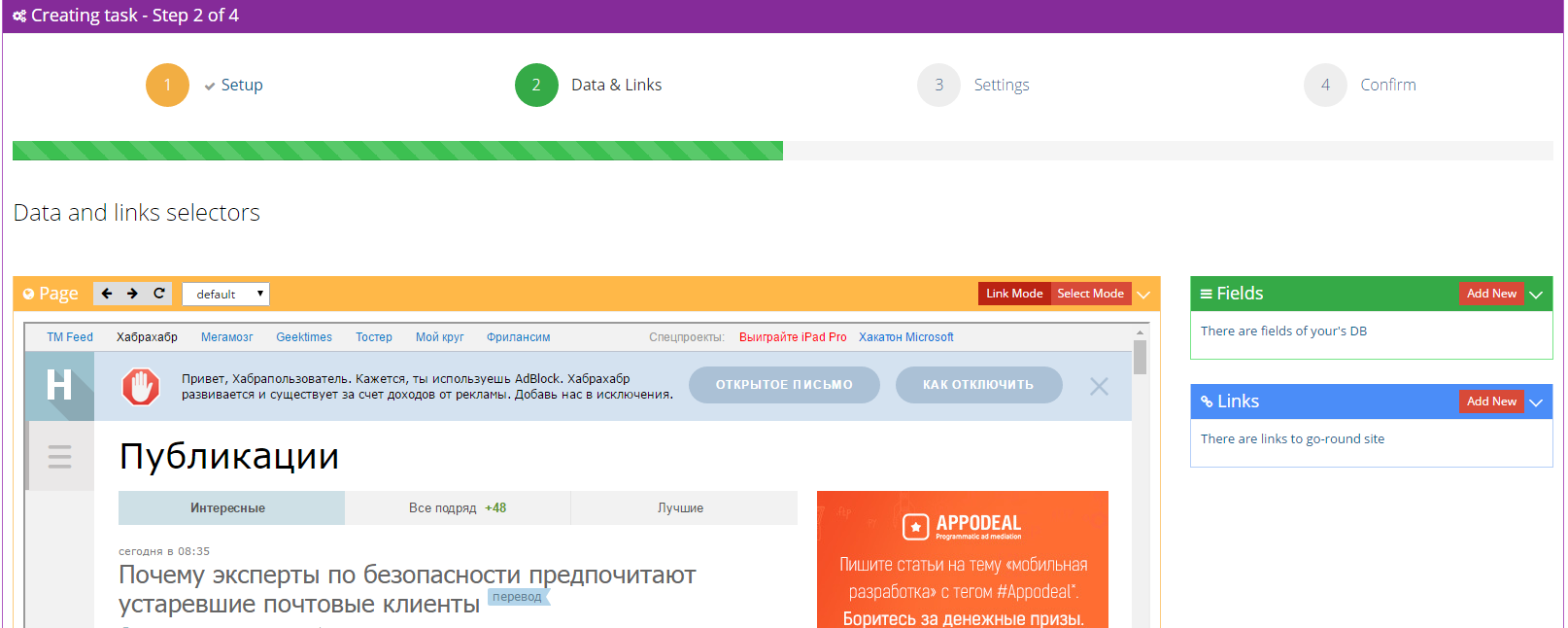


Рисунок 5.7 – Сторінка налаштування конфігурації – другий етап

1. Задання посилання на стартову Web-сторінку для початку обходу (рисунок 5.8).

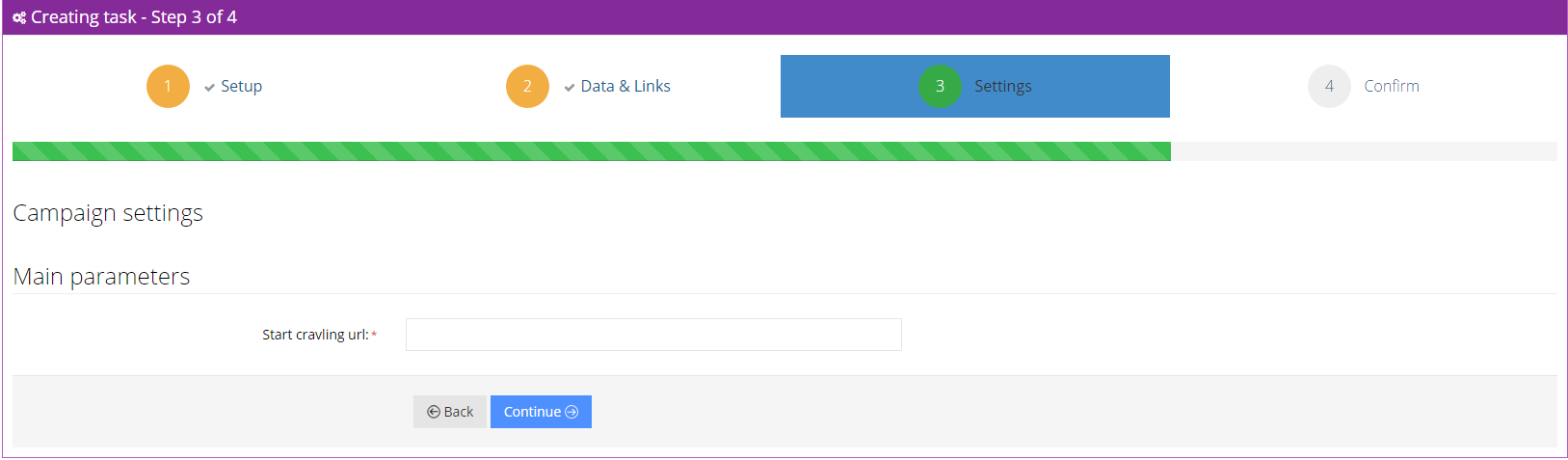


Рисунок 5.8 – Сторінка налаштування конфігурації – третій етап

Після створення завдання користувач перенаправляється на сторінку зі списком всіх його активних та закінчених завдань (рис. 5.9).

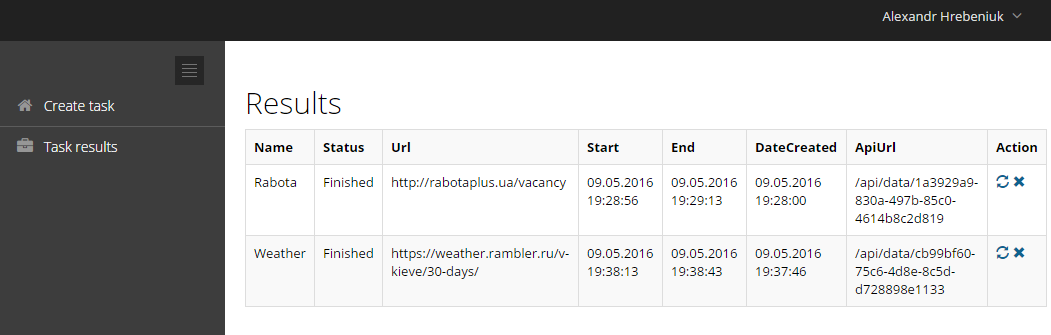


Рисунок 5.9 – Сторінка зі списком усіх завдань користувача

Також користувач має можливість перезапускати та видаляти завдання на сторінці «Список завдань» (рисунок 5.11).

Для перегляду структурованих даних через API, необхідно скопіювати «API Url» на сторінці «Список завдань». Через API користувач може отримати дані як в форматі json, так і в форматі xml в залежності від типу даних, які було потрібно під час запиту. Завдяки такому підходу можливо створення різних застосунків для роботи з даними. Вибір технології залежить від простоти реалізації API. На рисунку 5.10 представлено перегляд даних через API.

Також користувач має можливість аутентифікації, щоб його отримані дані булі доступні тільки йому. Перший екран, який він бачить у застосунку це логін (рисунок 5.12).

Якщо у користувача немає аккаунту то він має можливість зареєструватись, натиснувши кнопку «Sign up!» (рисунок 5.13).

Результати роботи зберігаються в базу даних MongoDb. Вони представлені на рисунку 5.14. Саме ці дані буде повертати API.

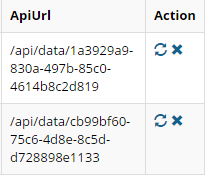


Рисунок 5.11 – Перезапуск та видалення завдань

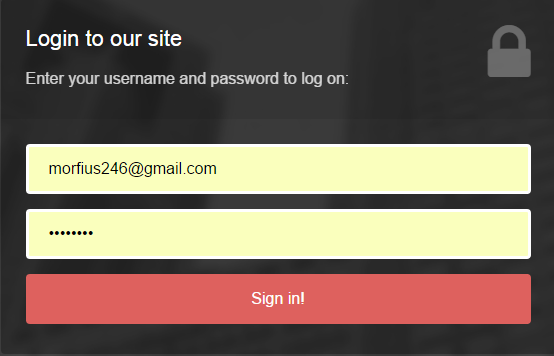


Рисунок 5.12 – Логін

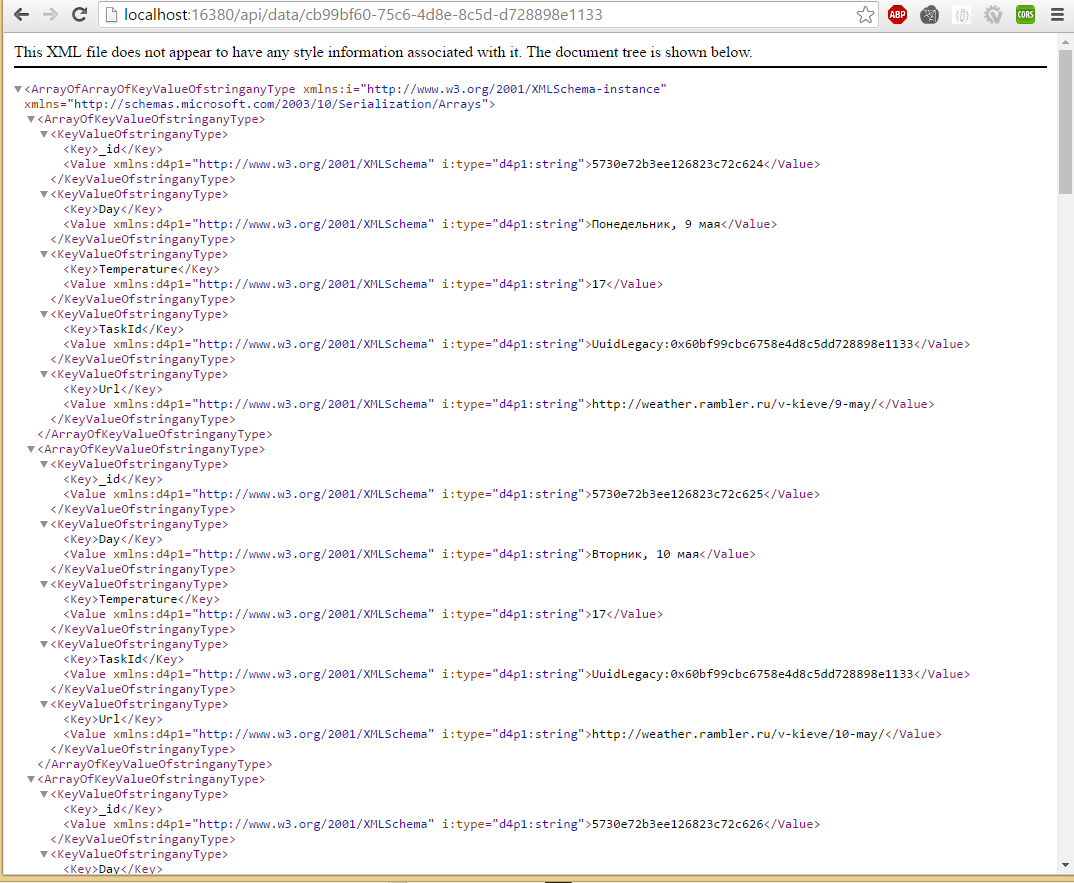


Рисунок 5.10 – Перегляд здобутих даних через API

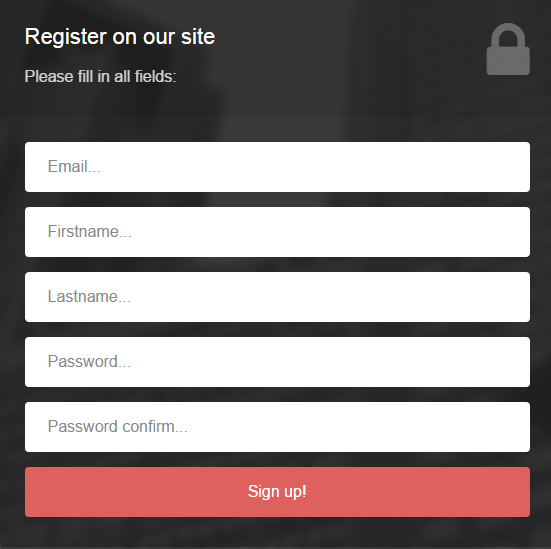


Рисунок 5.13 – Сторінка реєстрації

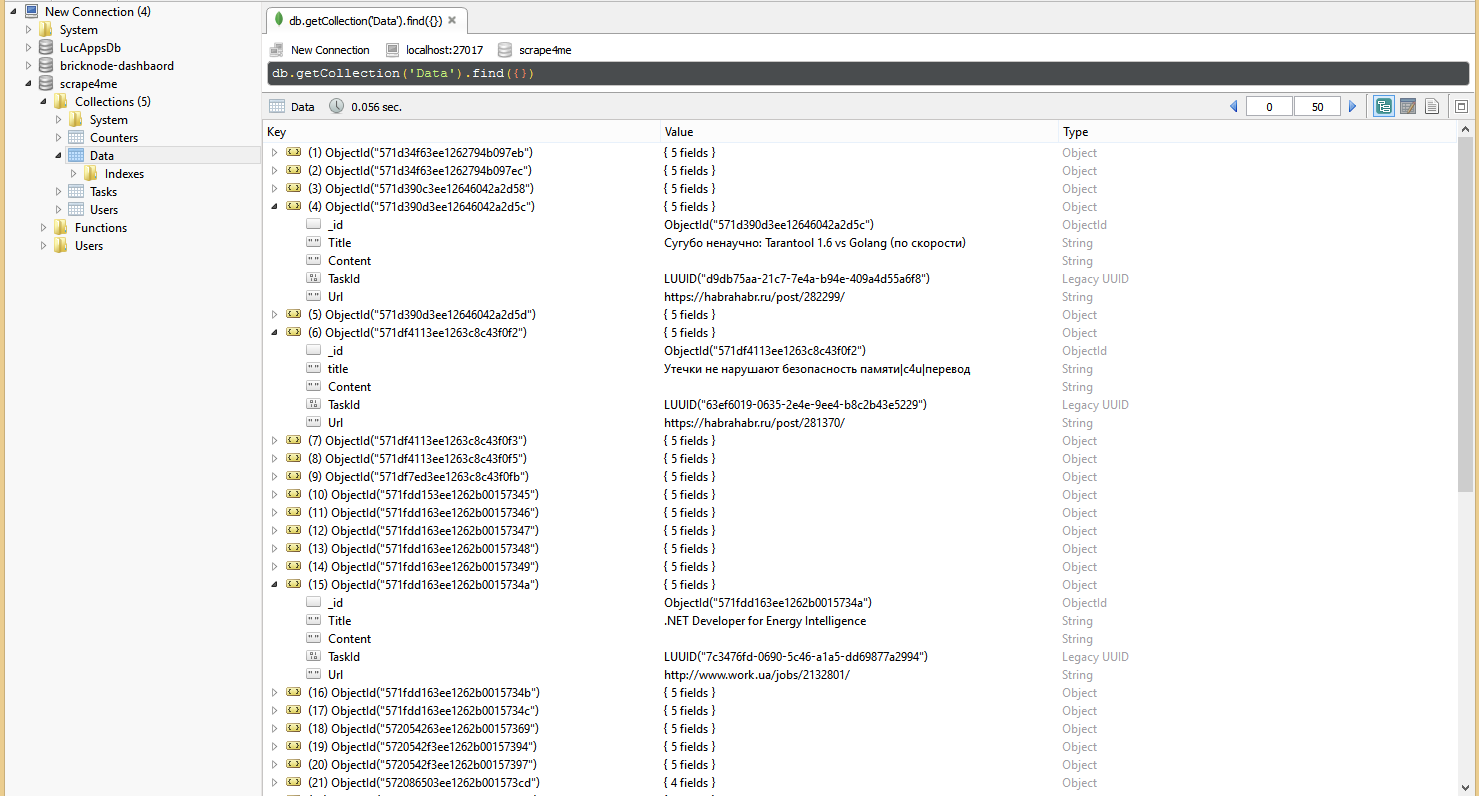


Рисунок 5.14 – Дані в базі даних MongoDb

ВИСНОВКИ

У ході виконання даної роботи було успішно розроблено систему A/B тестування, що дозволяє створювати та налаштовувати задачі A/B тестування, відслідковувати дії відвідувачів сайту користувача та зберігати статистичні дані у вигляді результатів A/B тестування. Було застосовано новітні та високопродуктивні інструменти розробки, які значно прискорюють роботу користувацького інтерфейсу, а також час відклику від API [12]. Система має простий користувацький інтерфейс, який дозволяє користувачам без додаткових знань користуватися системою.

Розроблено систему аналізу сторінки користувача, та можливість зручного вибору HTML елементу без додаткових технічних знань.

Розроблено механізм відображення A та B елементів на сторінці користувача та механізм збору статистичних даних.

Розглянуто переваги використання даної системи у сучасних бізнес процесах, наведено приклади існуючих систем, надано опис методів маркетингового дослідження, зокрема методу A/B тестування.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Troelsen A. Pro C# 5.0 and the .NET 4.5 Framework / A. Troelsen. – NY.: Apress, 2012. – 1463p.
2. Лабберс П. Pro HTML5 Programming: Powerful APIs for Richer Internet Application Development / П. Лабберс, О. Брайан, С. Френк., 2011. – 272 с.
3. Про авторське право і суміжні права: Закон України від 23.12.93 р. № N 3793-XII. – 1993. – 23 грудня. – С. 10.
4. Sharp J. Microsoft Visual C# 2012 Step by Step (Step By Step (Microsoft)) / J. Sharp. – C.: O’Reilly, 2013. – 775p.
5. Бэнкер К. MongoDB в действии. / К. Бэнкер. − М.: ДМК Пресс, 2012. − 394с.
6. Freeman A. Pro ASP.NET MVC 4 / A. Freeman. – NY.: Apress, 2012. – 717p.
7. Webharvy [Електронний ресурс] – Режим доступу до ресурсу: https://www.webharvy.com/.
8. Import.io [Електронний ресурс] – Режим доступу до ресурсу: https://www.import.io/.
9. Google Web Scraper [Електронний ресурс] – Режим доступу до ресурсу: https://chrome.google.com/webstore/detail/web-scraper/jnhgnonknehpejjnehehllkliplmbmhn.
10. Skonnard A. Essential XML Quick Reference / Aaron Skonnard., 2002. – 430 с. – (Addison-Wesley).
11. CSV [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/CSV.
12. Sai Srinivas S. JavaScript and JSON Essentials / Sriparasa Sai Srinivas., 2013. – 452 с. – (Amazon).
13. TSV [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Tab-separated\_values.
14. API [Електронний ресурс] – Режим доступу до ресурсу: - http://www.asp.net/web-api.
15. Munro J. Knockout.js Building Dynamic Client-Side Web Applications / Jamie Munro., 2014. – 102 с. – (O'Reilly Media).
16. Шевчук Ф. jQuery / Фнтон Шевчук. – Харків, 2012. – 122 с. – (Nix solutions).
17. Рихтер Д. CLR via С#. Программирование на платформе Microsot .NET Framework 4.0 на языке С#. 3-е изд / Д. Рихтер. — СПб.: Питер, 2012. — 928с.
18. Chodorow K.. Scaling MongoDB / K. Chodorow. – CA.: Course Technology, 2011. – 898 p.
19. Miner D. MapReduce Design Patterns / D. Miner, A. Shook., 2012. – 250 с. – (O'Reilly Media).
20. Professional Microsoft IIS 8 / K.Schaefer, J. Cochran, S. Forsyth, D. Glendenning., 2013. – 947 с. – (John Wiley & Sons).