

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ВАСИЛЯ СТУСА

**НІКОЛАЄВ ПАВЛО ОЛЕКСАНДРОВИЧ**

Допускається до захисту:

завідувач кафедри інформаційних  
технологій, к.т.н., доцент

\_\_\_\_\_ Т.В. Нескородева

« \_\_\_\_ » \_\_\_\_\_ 20\_\_ р.

**АВТОМАТИЗОВАНА СИСТЕМА ЗБОРУ НОВИН «НОВИННИЙ  
АГРЕГАТОР»**

Спеціальність 122 Комп'ютерні науки

**Кваліфікаційна (бакалаврська) робота**

Керівник:

Нескородева Т.В., к.т.н., доцент,  
завідувач кафедри ІТ

\_\_\_\_\_,  
(назва кафедри)

Оцінка: \_\_\_\_ / \_\_\_\_ / \_\_\_\_  
(бали за шкалою СКТС/за національною шкалою)

Голова ЕК: \_\_\_\_\_  
(підпис)

## АНОТАЦІЯ

**Ніколаєв П.О Розробка автоматизованої системи збору новин.** Спеціальність 122 «Комп'ютерні науки», спеціалізація «Інформаційні технології»

Донецький національний університет імені Василя Стуса, Вінниця, 2021.

У кваліфікаційній (бакалаврській) роботі досліджені сучасні методи подання інформації через засоби масової інформації. Розгляд аналогів, порівняльний аналіз представлених веб-джерел та систем збору новин. Пошук методів для вирішення знайдених проблем. Показана ефективність автоматизованої системи збору новин перед класичними методами подання масової інформації. Розроблений веб-додаток з ціллю донесення інформації сучасними методами.

Ключові слова: сучасні методи подання інформації, автоматизована система збору новин, веб-додаток, новинний агрегатор.

58 с., 20 рис., 3 дод., 22 джерела.

## ABSTRACT

**Nikolaev PO Development of the automated system of collecting news.** Specialty 122 "Computer Science", specialization "Information Technology". Vasyl Stus Donetsk National University, Vinnytsia, 2021.

In the qualification (bachelor's) work the modern methods of presenting information with the help of mass media are investigated. Consideration of analogues, comparative analysis of presented web sources and systematic collection of news. Search for methods to solve the problems found. The efficiency of the automated system of collecting news before the classical methods of presenting mass information is shown. Developed a web application to convey information by modern methods.

Key words: modern methods of presenting information, automated news collection system, web application, new aggregator.

## ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ .....	5
ВСТУП .....	6
1. АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ, ПОСТАНОВКА ЗАДАЧІ ТА ОГЛЯД ІСНУЮЧИХ АНАЛОГІВ .....	7
1.1 Актуальність теми.....	7
1.2. Розгляд аналогів .....	8
1.3. Постановка задачі .....	11
2. АНАЛІЗ ТА ВИБІР АКТУАЛЬНИХ ТЕХНОЛОГІЙ ДЛЯ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ .....	13
2.1. Вибір програмних засобів розробки .....	13
2.2. Поняття односторінкового додатку .....	35
3. ОПИС РОЗРОБКИ ТА ПРОЕКТУВАННЯ ПОСТАВЛЕНОЇ ЗАДАЧІ .....	34
3.1. Загальна структура веб-додатку.....	39
3.2. Розробка веб-інтерфейсу.....	40
3.3. Реалізація серверу та хостингу.....	45
3.4. Реалізація парсингу даних .....	47
3.5. Обробка даних та робота з отриманою інформацією .....	51
3.6 Взаємодія користувача та веб-додатку .....	54
ВИСНОВОК.....	55
ЕЛЕКТРОННІ РЕСУРСИ ВІДДАЛЕНОГО ДОСТУПУ .....	56



## ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

СУБД – Система управління базами даних;

БД – База даних;

ЗМІ – Засоби масової інформації;

JS – JavaScript (Мова програмування ДжаваСкрипт);

DOM – Document Object Model (Об'єкт модель документа)

HTML – HyperText Markup Language (Мова розмітки гіпертексту)

CSS – Cascading Style Sheets (Каскадні таблиці стилів)

API – Application programming interface (Програмний інтерфейс додатку)

RTA – Real Time Application (Додаток, що працює в реальному часі)

NPM – Node Packet Manager (Менеджер пакетів Node.js)

JSON – JavaScript Object Notation (Запис об'єктів JavaScript)

GIT – Розподілена система керування версіями файлів

XML – Extensible Markup Language (Розширювана мова розмітки)

XHR – XML Http Request (XML запис)

JQuery - фреймворк, розроблений на основі технологій JavaScript, який допомагає спростити написання великих фрагментів коду.

ODM – Original Design Manufacturer (Виробник оригінального дизайну)

SPA – Single Page Application (Односторінковий веб-додаток)

## ВСТУП

Новинні сайти видань, новини в соціальних мережах, блоги ЗМІ містять велику кількість інформації та постів, що не завжди зручно та цікаво читати користувачеві. У таких випадках користувачеві доводиться витратити багато часу на пошук потрібної інформації.

Сьогодні існує багато методів для отримання інформації, вони відрізняються між собою дизайном, тематикою пропонованих новин, масштабом додатку. Водночас, мало які сайти можуть надати вибір розділу по новинним статтям та надають актуальну інформацію.

Тема бакалаврської роботи – «Розробка автоматизованої системи збору новин». Дана тема актуальна з наступних причин:

- Не всі новинні сайти надають цікаву та актуальну користувачеві інформацію. При цьому, більшість людей хочуть зберегти свій час і одразу читати тільки актуальні для них новини без реклами та зайвої інформації;
- Власники сайтів не дають гарного інструменту вибору актуальної тематики новин для користувача.

Мета даної бакалаврської роботи - є створення агрегатора новин у вигляді веб-додатку, що дозволить користувачам, підключеним до мережі Інтернет, мати доступ до актуальних новин від перевірених джерел.

Перевагою даного веб-додатку перед аналогами є те, що в ньому зібрані лише актуальні розділи з новинами, відсутність реклами та зайвої інформації. Присутня можливість читати думки експертів про різні події в Україні та світі. Також, представлений розділ для перегляду новин Донецького Національного Університету ім. Василя Стуса.

# 1. АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ, ПОСТАНОВКА ЗАДАЧІ ТА ОГЛЯД ІСНУЮЧИХ АНАЛОГІВ

## 1.1. Актуальність теми

У всі часи людям було необхідно дізнаватись новини про всі події світу. В наш час це робити досить легко, варто лише перейти на популярний веб-додаток чи додаток у своєму смартфоні та переглядати новини, які підібрав алгоритм для користувачів. Але всі ці сервіси не є ідеальними та часто містять багато неактуальних новин, також, містять рекламу, яка відволікає від перегляду новин.

Також можна зазначити, що зараз доволі часто на просторах інтернету можна зустріти сервіси з неправдивими новинами, націленими на те, щоб дезінформувати велику частину населення та людей, які не поглиблюються в актуальність та правдивість новин, які вони переглядають. В свою чергу, дезінформація населення є значно більшою проблемою ніж здається на перший погляд, адже цим можуть зловживати з різними цілями неправдиві ЗМІ.

Зараз у світі відбуваються тенденції автоматизації будь-яких процесів. Перегляд та доставка новин на веб-джерела не є виключенням. Можна зазначити, що статистика говорить про те, що відсоток людей, які переглядають новини через веб-агрегатори у порівнянні з авторськими сайтами лише збільшується. Це говорить про те, що людям необхідно мати засіб для швидкого перегляду новин без пошуку відповідних джерел.

Тому розробка сервісу з лише актуальними новинами без реклами та зайвої інформації - є доволі актуальною, тому що даний сервіс дозволить людям економити свій час та дізнаватись актуальні та правдиві новини. І до того ж, розробка даного веб-додатку сприяє тенденції автоматизації всіх процесів у нашому житті.



## 1.2. Розгляд аналогів

### 1.2.1 Google News

Google Новини — це безплатний новинний агрегатор, який був створений та досі підтримується компанією Google.

Бета-версія була запущена у вересні 2002 року і офіційно випущена в січні 2006 року. Ідею розробив Крішна Бхарат.

3 червня 2009 року запрацювала українська версія сервісу. Вона стала 49-ю регіональною версією Google News. На момент утворення до сервісу було підключено понад 300 українських видавців.

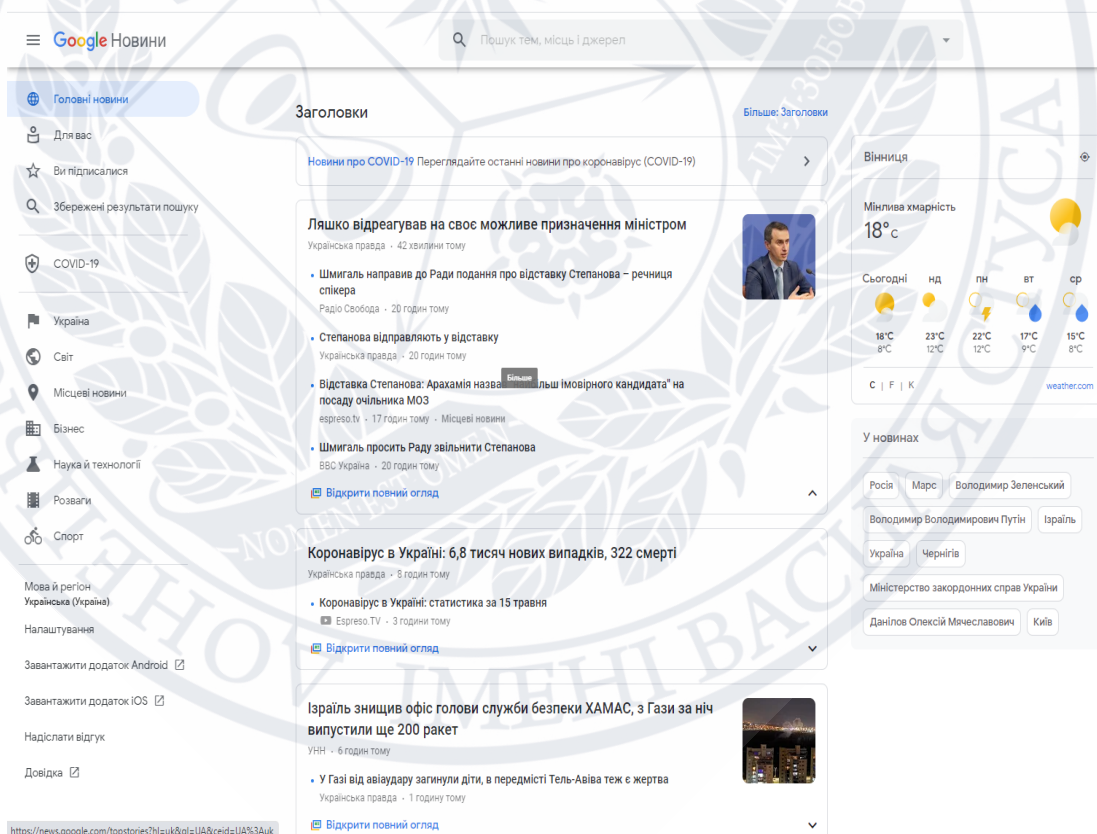


Рис. 1.1 — головна сторінка Google News.



### 1.2.2 Flipboard.com

Flipboard - це агрегатор новин та компанія з агрегування соціальних мереж, що базується в Пало-Альто, штат Каліфорнія, з офісами в Нью-Йорку, Ванкувері та Беджіні.

Його програмне забезпечення, також відоме як Flipboard, було вперше випущено в липні 2010 року. Він збирає вміст із соціальних мереж, стрічок новин, сайтів для обміну фотографіями та інших веб-сайтів, представляє його у форматі журналу та дозволяє користувачам "гортати" статті, зображення та спільні відео. Читачі також можуть зберігати історії в журналах Flipboard.

Станом на березня 2016 року компанія стверджує, що користувачами на Flipboard створено 28 мільйонів журналів. Доступ до послуги можна отримати через веб-браузер, або за допомогою програми Flipboard для Microsoft Windows та macOS, а також за допомогою мобільних додатків для iOS та Android. Клієнтське програмне забезпечення доступне безкоштовно і локалізоване 21 мовою.

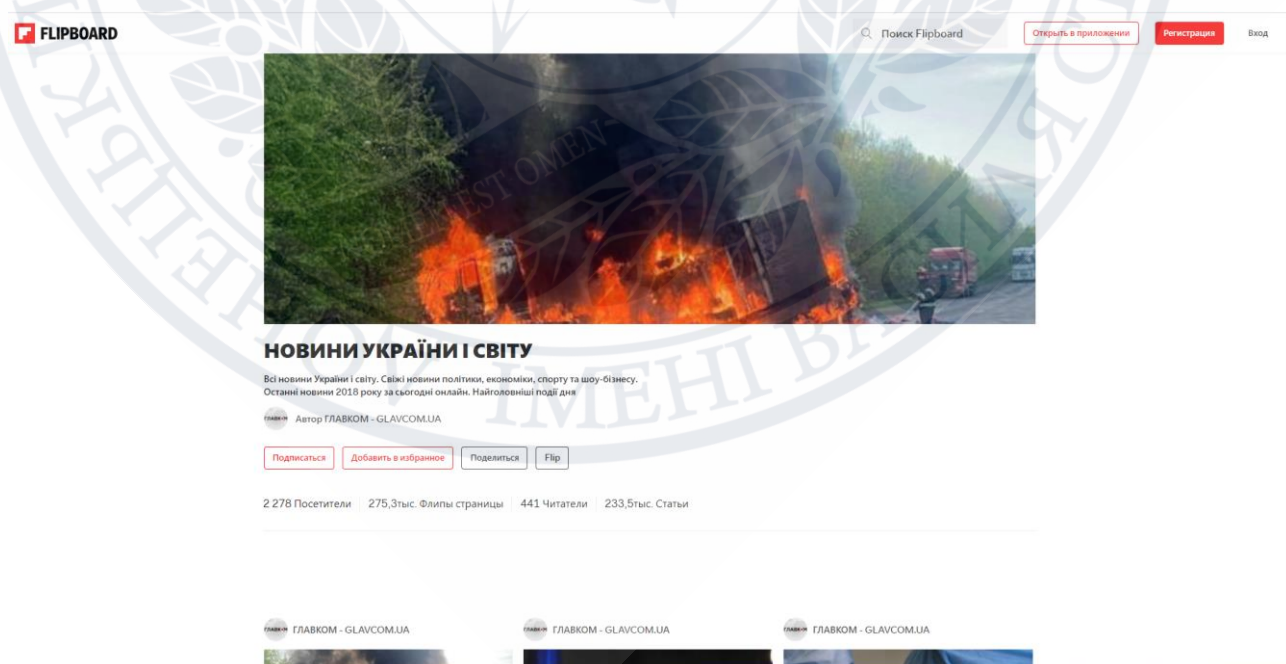


Рис. 1.2 – головна сторінка Flipboard.

### 1.2.3 Ukr.net

Ukr.net (Укр.нет) — один із найперших агрегаторів новин України. Також входить в список найбільш відвідуваних сайтів українського Інтернету. Станом на 24 травня 2020 року займає 8-ме місце за відвідуваністю.

Новини відображаються незалежно від політичної думки авторів та концепції інтернет-ресурсів, які дані новини публікують. Ресурс також надає поштову скриньку з тримовним інтерфейсом (українська, російська та англійська мови).

Поштовий сервіс було засновано 2000 року, тоді це була одна з перших безкоштовних електронних пошт в Україні.

Щодня до стрічки новин сайту автоматично додаються близько тридцяти тисяч записів від 1500 джерел.

The screenshot displays the Ukr.net homepage with a light blue and white color scheme. At the top left is the Ukr.net logo with the tagline 'це – мій інтернет!'. Below it are links for 'День Європи', 'День науки', 'Міжнародний день сім'ї', and 'Міжнародний день астрономії'. A search bar with a magnifying glass icon and the word 'Пошук' is positioned to the right of the logo. Below the search bar is a section for 'Головне' (Main) with a list of news items, including 'Израильские ВВС нанесли удар по высоте в Газе' and 'Збірна України з футболу розпочала підготовку до Євро-2020'. To the right of the main news section is a weather widget for 'Калинівка' showing a temperature of +17°C and a forecast for the next 10 days. Below the weather widget is a section for 'Великий вибір гаджетів' (Great choice of gadgets) with a link to 'Цитрус'. On the left side of the page, there is a sidebar with a login form for 'Пошта' (Mail) and a list of 'Вибране' (Selected) links to various services like Sinoptik, Rozetka, Kasta, and others. At the bottom of the page, there are sections for 'Економіка' (Economy) and 'Авто' (Cars) with links to related news and services.

Рис. 1.3 – головна сторінка Ukr.net.

### 1.3 Постановка задачі

Мета бакалаврської роботи полягає в аналізі існуючих агрегаторів новин, пошуку та вирішення їх проблем та створення веб-додатку “Автоматизована система збору новин”.

Для досягнення визначеної мети необхідно:

- проаналізувати всі сучасні агрегатори новин;
- визначити їх недоліки;
- пошук рішення актуальних проблем аналогів;
- пошук та вибір надійних та перевірених веб-джерел з новинами;
- побудувати схему веб-додатку;
- визначитись з вибором технологій для створення додатку;
- обґрунтувати вибір технологій;
- розробити повноцінний веб-додаток з базою даних та клієнт-серверною частиною.

При створенні власного веб-сайту необхідно пройти через кілька етапів - від придумування ідеї до її втілення. Щоб створити свій сайт, необхідно процес розділити на етапи:

- постановка задачі або технічне завдання розробки сайту. На даному етапі розробляється архітектура та структура майбутнього додатку.
- розробка і затвердження дизайну веб-додатку.
- інтеграція дизайну і системи управління сайтом. Даний веб-додаток за визначенням потребує деякої системи управління, оскільки планується велика кількість оновлюваного матеріалу.
- хостинг веб-додатку в Інтернет. Для додавання сайту в Інтернет необхідно декілька завдань: -
- Перше - полягає у виборі адреси сайту його імені (домен).



- Друге завдання - це вибір фізичного розміщення вашого сайту (хостинг).
- заповнення сайту матеріалами.
- підтримка сайту.





## 2. АНАЛІЗ ТА ВИБІР АКТУАЛЬНИХ ТЕХНОЛОГІЙ ДЛЯ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

### 2.1 Вибір програмних засобів розробки

В даному розділі переглянуті програмні засоби, які знадобились для розробки веб-додатку, такі як: JavaScript, React.js, Node.js, Express, Heroku, MongoDB, SemanticUI, Axios, Cheerio, Unirest, Mongoose, Redux, HTML, CSS, Git, Path та VS Code – в якості середовища для написання та тестування коду.

#### 2.1.1. JavaScript

JavaScript – це мова сценаріїв, або програмування, що дозволяє впроваджувати на веб-сторінках складні функції, кожен раз, коли веб-сторінка робить більше, ніж просто відображає статичну інформацію, яку користувач може переглянути. За допомогою JavaScript може відображатись своєчасне оновлення вмісту, інтерактивні карти, анімовані 2D, тривимірні графіки, відображення відео-плеєру тощо. Це третій шар стандартних веб-технологій, два з яких це HTML та CSS.

Як і HTML та CSS, JavaScript є головною з основних технологій веб-розробки. Майже усі сайти використовують дану мову програмування на клієнтській стороні веб-додатку для визначення поведінки сайту, часто включаючи сторонні бібліотеки. Усі найпопулярніші браузері мають спеціальний застосунок для виконання коду JavaScript на пристрої користувача.

ECMAScript – мова програмування сценаріїв, розроблена разом з Microsoft та Netscape. Дана мова є похідною від JavaScript Netscape, яка широко використовує мови сценаріїв та відповідає на веб-сторінках за те, як вони

виглядають або поведуться на стороні клієнта. Офіційний стандарт, ECMA-262, був розроблений під наглядом Європейської асоціації виробників комп'ютерів (ЕСМА). Наявність єдиного стандарту допоможе забезпечити більшу узгодженість між компаніями, які розробили даний стандарт та іншими реалізаціями веб-сценаріїв.

ECMAScript є об'єктно орієнтованим і задуманий як основна мова, до якої можуть додаватися об'єкти будь-якого конкретного домену або контексту, наприклад ідея «документа». (наприклад, об'єктна модель документу консорціуму Всесвітньої павутини).

ECMAScript разом із об'єктною моделлю документа відповідає поточним реалізаціям JavaScript та JScript. Незважаючи на те, що ECMAScript може використовуватися в основному як стандартна мова сценаріїв для Всесвітньої павутини, ECMAScript також може використовуватися для будь-яких сценаріїв.

Спочатку двигуни JavaScript використовувались лише у браузерях, але зараз вони є основними компонентами інших програмних систем, зокрема серверів та різноманітних додатків.

В першу чергу JavaScript широко використовується у фронтенд-розробці. Ця мова разом з HTML і CSS входить в базовий набір інструментів фронтендера. На JavaScript створюються додатки, які виконуються в браузері на стороні клієнта. Вони забезпечують інтерактивність сайтів. Наприклад, коли користувач заповнює форму і натискає кнопку «Підписатися», миттєва реакція на цю дію зазвичай забезпечується кодом, написаним на JavaScript.

Сфери застосування JavaScript не обмежуються браузерами і веб-додатками. За допомогою цієї мови вирішують такі завдання:

- Розробка нативних додатків. Наприклад, за допомогою фреймворка React Native створюються додатки для Android і iOS;
- Серверна розробка. Node.js застосовується для бекенд-розробки;
- Розробка десктопних додатків. JS застосовується в офісних пакетах Microsoft і OpenOffice, в додатках компанії Adobe;

- Програмування обладнання та побутової техніки, наприклад, платіжних терміналів, телевізійних приставок.

В даний момент JavaScript - єдина високорівнева динамічна мова програмування, доступна практично скрізь, в тому числі (і в першу чергу) на веб-сторінках. Це дозволяє вивчити одну мову, частково перевикористати готові напрацювання для клієнта, сервера, в робото-будуванні, для інтернету речей, і навіть для машинного навчання.

```
//
// z(i+1) = z(i)^2 + c
// terminate when |z| > 2.0
// returns 4 iteration counts
//
function mandelx4(c_re4, c_im4) {
    var z_re4 = c_re4,
        z_im4 = c_im4,
        four4 = SIMD.float32x4.splat (4.0),
        two4 = SIMD.float32x4.splat (2.0),
        count4 = SIMD.int32x4.splat (0),
        one4 = SIMD.int32x4.splat (1),
        i, z_re24, z_im24, mi4, new_re4, new_im4;

    for (i = 0; i < max_iterations; ++i) {
        z_re24 = SIMD.float32x4.mul (z_re4, z_re4);
        z_im24 = SIMD.float32x4.mul (z_im4, z_im4);

        mi4 = SIMD.float32x4.lessThanOrEqual (SIMD.float32x4.add (z_re24, z_im24), four4);
        // if all 4 values are greater than 4.0, there's no reason to continue
        if (mi4.signMask === 0x00) {
            break;
        }

        new_re4 = SIMD.float32x4.sub (z_re24, z_im24);
        new_im4 = SIMD.float32x4.mul (SIMD.float32x4.mul (two4, z_re4), z_im4);
        z_re4 = SIMD.float32x4.add (c_re4, new_re4);
        z_im4 = SIMD.float32x4.add (c_im4, new_im4);
        count4 = SIMD.int32x4.add (count4, SIMD.int32x4.and (mi4, one4));
    }
    return count4;
}
```

Рис. 2.1 – Приклад JavaScript коду

У практичній частині бакалаврської роботи мова JavaScript використовується як єдина мова програмування, так як вона розвинулась до того стану, що однією мовою програмування за допомогою різних бібліотек є змога розробляти як клієнтську, так і серверну частину веб-додатку. Також, написання серверної частини на мові програмування JavaScript, забезпечує швидкодію та можливість використання великої кількості бібліотек та фреймворків.



### 2.1.2. React.js

React (React.js або ReactJS) - інтерфейсна бібліотека JavaScript із відкритим кодом для побудови користувальницьких інтерфейсів або компонентів інтерфейсу.

React - найпопулярніша інтерфейсна бібліотека JavaScript у галузі веб-розробки. Її використовують великі, засновані компанії та нові стартапи (Netflix, Airbnb, Instagram та New York Times, щоб назвати декілька). React приносить в розробку багато переваг, що робить його кращим вибором, ніж інші фреймворки, такі як Angular.js. React зазвичай використовують через його особливості архітектури додатку – усі елементи зовнішнього виду сайту зберігаються в компонентах та окремих файлах, що спрощує розробку та масштабування веб-додатку.

Методи життєвого циклу ReactJS дозволяють додатку виконувати код на будь-яких стадіях життєвого циклу компонента, наприклад:

- `shouldComponentUpdate` – за допомогою даного методу можна запобігти перемалювання компонента за допомогою повернення `false`.
- `componentDidMount` – запускається під час першого рендеру компонента. Використовується також для отримання даних через API.
- `render` - головний метод у життєвих циклах. Кожен компонент має доступ до цього методу. Викликається лише при зміні даних якогось компоненту для перерисовки даних в інтерфейсі користувача.

Для написання коду React, використовується спеціальний тип файлів – JSX. Даний тип файлів – це деяка суміш JavaScript та HTML коду, але насправді це все JS код.



```

1 import React from 'react';
2 import 'semantic-ui-css/semantic.min.css'
3 import { Item, Label, Link } from 'semantic-ui-react';
4
5 const Post = ({ title, description, preDescription, authorName, image, url, time, readNext }) => {
6   return (
7     <Item>
8       <Item.Image src={image} />
9       <Item.Content>
10        <Item.Header as='a' href={url}>{title}</Item.Header>
11        <Item.Description>{description}<Item.Description as='a' href={url}>{readNext}</Item.Description></Item.Description>
12        <Item.Extra>
13          <Label icon='pencil alternate icon' content={`Автор: ${authorName}`} />
14          <Label icon='clock outline icon' content={`Дата старту: ${time}`} />
15        </Item.Extra>
16      </Item.Content>
17    </Item>
18  );
19 };
20
21 export default Post;

```

Рис. 2.2 – приклад JSX коду в React (особиста розробка)

#### Відомі проекти на React:

- Facebook. У ньому React використовується частково, але і в версії для ПК, і в мобільному додатку;
- Instagram. В такому популярному додатку даній бібліотеці відводиться величезна роль. Починаючи з можливості визначення геопозиції і закінчуючи точністю функціоналу пошуку - подібні речі часто роблять на React;
- Netflix. Найактивніше задіюється на платформі Gibbon. Основною функцією стає можливість налаштувати параметри для телевізорів з низькою продуктивністю. Бібліотека допомагає прискорити завантаження і підвищити продуктивність - ось де програмісти точно знали для чого потрібен React js;
- Yahoo! Mail. Завдяки Facebook, ці сервіси стали впорядкованими в плані архітектури. Знадобилося безліч оновлень, щоб привести до бажаного результату: легкої налагодженні, низького рівня входження, незалежно розгортається компонентів. Реактив підійшов їм через низку його властивостей: одностороннього потоку даних, можливості використовувати віртуальний DOM, активна спільнота;

- WhatsApp - фахівці цього сервісу вирішили використовувати React для створення користувацьких інтерфейсів;
- Dropbox. На хвилі популярності бібліотеки, її почали застосовувати і для цього сайту.

В розробці веб-додатку React.js використовується як основний фреймворк для створення користувацького інтерфейсу зручним та легким способом, який легко масштабується, а для веб-додатку на тематику новин, це є великою перевагою.

### 2.1.3. Node.js

Node.js – це середовище виконання коду на стороні серверу з відкритим кодом, побудованим на механізмі JS V8 Chrome. Він забезпечує керування подіями, асинхронний ввід-вивід та крос-платформенне середовище для виконання коду високо-масштабованого серверного додатку побудованого на базі JavaScript.

Node.js може бути використаний для побудови різних типів додатків, такий як утиліта командного рядка, веб-програма, додаток часу в режимі реального часу, REST API сервер тощо. Але в основному він використовується для побудови мережових додатків, такий як веб-сервери, схожі до PHP, ASP.NET і Java.

Node.js – фреймворк, який використовує JavaScript для створення цілого додатку на стороні серверу. Легкий каркас, що включає мінімум модулів. Інші модулі можуть бути включені відповідно до потреби програми. За замовченням присутня асинхронність, що дозволяє програмувати важкі додатки. І через це, він працює швидше, ніж інші фреймворки. Підтримується Windows, Linux та MAC.

У традиційній моделі веб-сервера кожен запит обробляється виділеним потоком із кластеру потоків. Якщо жодного потоку не спостерігається, тоді запит

чекає до наступного доступного потоку. Виділений потік завжди виконує певний запит і не повертається до кластеру потоків, поки не завершить виконання і не поверне відповідь.

Node.js обробляє запити користувачів по-різному порівняно з традиційною моделлю сервера. Даний фреймворк працює в одному процесі, а код програми працює в одному потоці і потребує менше ресурсів, ніж інші платформи. Усі запити користувачів до веб-програми оброблятимуться одним потоком, і всі роботи виводу або вводу або тривале завдання виконуються асинхронно для конкретного запиту. Отже, цей єдиний потік не повинен чекати завершення запиту і може вільно обробляти наступний запит. Коли асинхронна робота вводу-виводу завершується, вона обробляє запит далі і надсилає відповідь.

Модель процесу Node.js збільшує продуктивність та масштабованість за допомогою кількох застережень. Node.js не придатний для програми, яка виконує інтенсивні процесорні операції, такі як обробка зображень або інші важкі обчислювальні роботи, оскільки для обробки запиту потрібен час і тим самим блокує один потік.

Вибір серверного JavaScript для бекенд забезпечує проекту ряд переваг:

- зростання ефективності розробки завдяки використанню однієї мови для фронт-і бекенд і можливості перевикористання коду;
- можливість використовувати npm - найбільший пакетний менеджер;
- простіший у порівнянні з іншими стеками технологій, так як JavaScript входить в число найпопулярніших мов програмування.

Node.js добре підходить для розробки RTA - веб-додатків, що реагують на дії користувача в режимі реального часу. Наприклад, це може бути онлайн-редактор типу Google Docs, який дозволяє працювати над одним документом кільком користувачам одночасно.

Так як, Node.js має змогу обробляти велику кількість запитів одночасно і як наслідок цього серверний JavaScript часто використовують для створення SPA



- односторінкових веб-додатків, в яких рендеринг виконується на стороні клієнта. Node.js на бекенд використовують Netflix, Uber, eBay, Groupon, Yahoo та інші відомі організації та проекти.

```
57 app.use('/ukr', (req, res) => {  
58   console.log('Api')  
59   Ukraine.find()  
60     .then((data) => {  
61     // console.log('Data: ', data);  
62     console.log('Ukraine')  
63     res.json(data);  
64   })  
65   .catch((error) => {  
66     console.log('error: ', error);  
67   });  
68 });  
69  
70 app.use('/IT', (req, res) => {  
71   console.log('Api')  
72   IT.find()  
73     .then((data) => {  
74     //console.log('Data: ', data);  
75     console.log('IT')  
76     res.json(data);  
77   })  
78   .catch((error) => {  
79     console.log('error: ', error);  
80   });  
81 });
```

Рис. 3.4 – приклад Node.js коду

За замовчуванням до Node.js входить унікальний за своєю концепцією установник пакетів npm. В ньому присутні майже всі відомі бібліотеки та пакети, які можуть знадобитись розробникові для створення додатку.

На даний момент існують такі найпопулярніші фреймворки Node:

- Adonis.js;
- Express.js;
- Fastify;
- Koa.js;
- LoopBack.js;
- Meteor.js;
- Nest.js;
- Sails.js.



### 2.1.4. Стек MEAN, MERN, MEVN

Стек MERN – це скорочення, що використовується для опису певного набору технологій на основі JavaScript, який використовуються в процесі розробки веб-додатків. Він розроблений з ідеєю зробити процес розробки максимально плавним. MERN включає такі компоненти з відкритим кодом:

- MongoDB;
- ExpressJS;
- ReactJS/Redux;
- NodeJS.

Кожен з цих компонентів відіграє вирішальну роль у процесі розробки веб-додатків. Все це забезпечує наскрізну структуру для роботи розробників. Наприклад, MongoDB – це система баз даних, Node JS - це внутрішнє середовище виконання, Express JS - це внутрішній веб-фреймворк, а React - це інтерфейсний фреймворк.

Node.js в розробці додатку використовується як набір утиліт для створення повноцінної серверної частини з взаємодіями з клієнтською частиною, базою даних та будь-яких інформаційних веб-джерел, з яких витягується інформація для новинних статей.

У той час як традиційні реляційні бази даних мають типовий дизайн схеми, заснований на шпальтах і таблицях, MongoDB не вимагає схем. Дані зберігаються в гнучких документах за допомогою мови запитів на основі JSON (JavaScript Object Notation). Зміст, розмір і кількість полів в документах можуть відрізнятися від одного до іншого. Це означає, що структура даних буде змінюватися з часом.

Express є структурою веб-додатку для Node.js, іншого компонента MERN. Замість того, щоб писати повний код веб-сервера вручну на Node.js, розробники використовують Express, щоб спростити завдання написання коду сервера.

Немає необхідності повторювати один і той же код знову і знову, як це було б з HTTP-модулем Node.js. Платформа Express призначена для створення надійних веб-додатків і API.

Замість того щоб покладатися на шаблони для автоматизації створення повторюваних елементів HTML або DOM (об'єктна модель документа), React використовує повнофункціональний мову програмування (JavaScript) для створення повторюваних або умовних елементів DOM. З React один і той же код може виконуватися як на сервері, так і в браузері.

### 2.1.5. MongoDB

MongoDB - це багато-платформна БД, яка має напрям на документи з відкритим кодом, і також є різновидом бази даних NoSQL. Як база даних NoSQL, MongoDB не використовує структури таблиць на основі традиційних баз даних, задля адаптування до подібних до JSON документів, що мають динамічні схеми, які мають назву BSON. Це робить інтеграцію даних для певних класів програм простішою та швидшою. MongoDB створений для масштабованості, високої доступності та продуктивності від розгортання одного сервера до великих та складних багатосайтових інфраструктур.

Особливості MongoDB:

- Спеціальні запити - підтримує пошук за полем, пошук за регулярними виразами та запити по діапазону.
- Індексация - будь-яке поле документа BSON може бути проіндексовано.
- Реплікація - забезпечує високу доступність за допомогою наборів реплік, що складається з двох або більше копій вихідних даних.
- Агрегація - MapReduce може бути застосована, щоб дозволити пакетну обробку даних, а також виконувати операції агрегування.

- Зберігання файлів - MongoDB може використовуватися як файлова система.

MongoDB стала однією з популярних не реляційних баз даних, використовуючись як серверна база для багатьох найпопулярніших веб-сайтів, включаючи eBay, Craigslist, SourceForge та The New York Times. MongoDB доступний під загальною публічною ліцензією GNU Affero, тоді як його мовні драйвери доступні під ліцензією Apache. Також пропонуються комерційні ліцензії.

MongoDB часто вибирають, коли потрібна масштабована база даних, в даний час її використовують як сховище внутрішніх даних багато організацій, такі як IBM, Twitter, Zendesk, Forbes, Facebook, Google та інші.

Приклади, коли MongoDB підходить для проекту:

- Каталог товарів в електронній комерції;
- Блоги та системи управління контентом, особливо ті, де багато контенту, в тому числі відео та зображення;
- Зберігання даних датчиків і пристроїв;
- Робота з великими даними для машинного навчання і досліджень в різних галузях;
- Ведення даних на основі розташування, тобто геопросторових даних;
- Соціальні мережі, новинні форуми та інші схожі сценарії використання середовища для зберігання даних;
- Слабко-зв'язані дані без чіткої схеми зберігання;
- Стартапи і розгортання нових проектів, де структура даних поки невідома.

Приклади, коли MongoDB краще не використовувати:

- Транзакційні системи, додатки, що вимагають транзакцій на рівні бази даних, наприклад банківські програми;
- Проекти, де модель даних визначена заздалегідь;



- Зберігання сильно-зв'язаних даних.

В практичній частині бакалаврської роботи MongoDB використовується як СУБД для зберігання інформації з різних веб-джерел для відображення її в подальшому в клієнтському інтерфейсі для перегляду новин.

#### **2.1.6. Express.js**

Express.js - це основна бібліотека для Node.js. Його називали фактичним стандартним серверним фреймворком для Node.js.

На даний момент, бібліотека Express.js використовується практично в кожному веб-додатку, де використовується Node.js, через свою простоту та ефективність в створенні серверних додатків. Express.js використовується в таких компаніях як: Fox Sports, PayPal, Uber та IBM.

Для розробки була вибрана бібліотека Express.js для створення серверу, веб-шляхів та практично всієї серверної частини веб-додатку.

#### **2.1.7. Heroku**

Heroku – хмарна платформа, для розміщення веб-додатків на хостинг. Підтримує найпопулярніші мови програмування, такі як: JavaScript, Python, Java, Scala, PHP та Go.

Сервер Heroku Git обробляє сховища додатків, що надходять від користувачів. Варто відзначити, що сервіс є безкоштовним, що є великим плюсом для розробників-початківців, і також повністю синхронізується з найпопулярнішою системою контролю версій – GitHub. Усі сервери Heroku використовують хмарні послуги Amazon та Azure.

Heroku повністю самокерована, що дає розробникам свободу зосередитися на своєму продукті. Heroku надає послуги, інструменти для врегулювання



робочого процесу - все це призначене для підвищення продуктивності розробників.

Heroku дуже корисний для компаній, у яких проектів багато і проекти ці не банальні сайти-візитки, а щось більш складне. Зазвичай в таких компаніях є цілий відділ DevOps, які розгортають всю інфраструктуру в тому ж EC2. Найбільша перевага Heroku перед конкурентами полягає в тому, що відділ DevOps вже не потрібен. Всі розгортання проектів, додатків, підключення будь-якого сервісу (бази даних, логирование, моніторинг, розсилка пошти і т.д.) робляться в пару кліків. Виходить, що вся робота девопсів вже зроблена.

У бакалаврській роботі сервіс Heroku був вибраний через свою простоту, зручність, широкий функціонал для керування та відслідковування стану серверу. Потрібно зазначити, що безкоштовна версія сервісу має досить широкий функціонал та ні в чому не обмежує розробників.

#### **2.1.8. SemanticUI**

Semantic - це бібліотека, яка допомагає легко та швидко створювати адаптивний та зручний для користувача макет та дизайн сайту. Варто зазначити, що SemanticUI представляє для розробників лише основу макетів та дизайнів, та під кожен додаток потрібно редагувати вже готові класи та додавати власноруч зроблені.

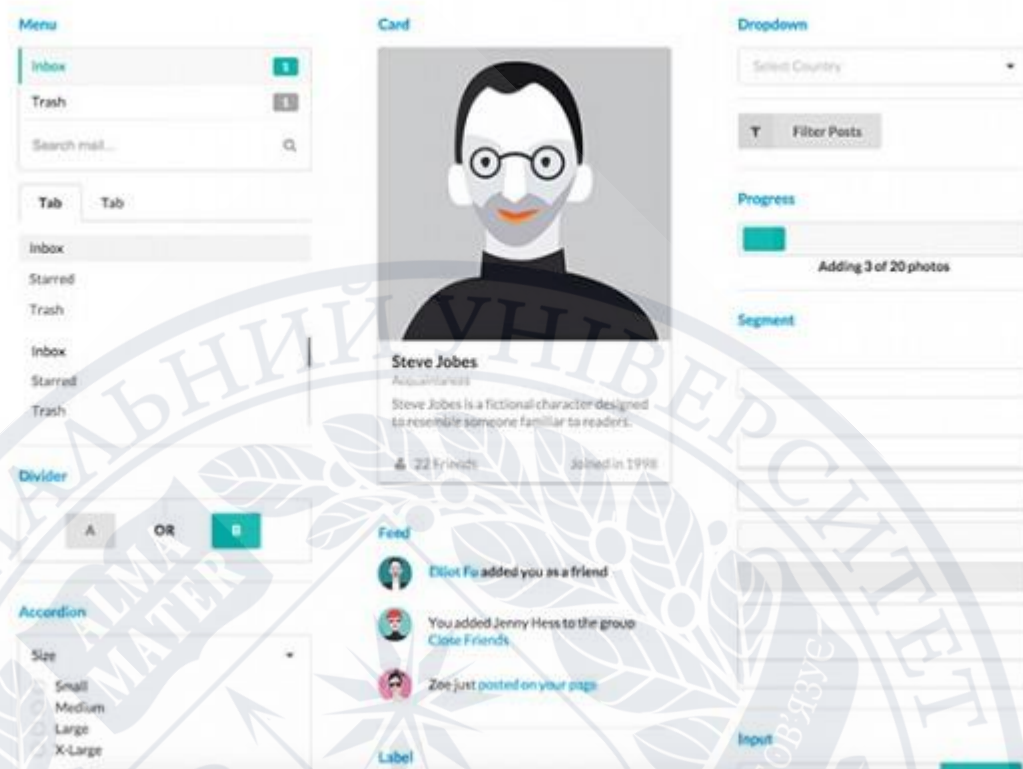


Рис. 2.3 – приклад макету та дизайну SemanticUI

Для розробки була вибрана бібліотека SemanticUI для створення макету користувацького інтерфейсу. У веб-додатку можна використовувати готові класи, змінивши їх в залежності від особливостей додатку.

### 2.1.9. Axios.js

Axios - це основна бібліотека JavaScript для створення HTTP-запитів, яка працює як на платформах будь-яких браузерів, так і на Node.js. Дана бібліотека підтримується усіма браузерами.

Axios заснований на обіцянках (Promise), і це дозволяє писати код `async/await` для дуже простого виконання запитів XHR.

Використання Axios має чимало переваг перед власним API Fetch:

- підтримує старі браузери (Fetch потребує полізаповнення);

- має метод для переривання запиту;
- має метод встановлювання часу очікування відповіді;
- підтримує хід завантаження, і це дозволяє відслідковувати статус відповіді до серверу;
- виконує автоматичне перетворення даних JSON;
- працює в Node.js.

Axios може бути встановлений за допомогою менеджера пакетів NPM:

- `npm install axios.`



```
axios({
  url: 'https://dog.ceo/api/breeds/list/all',
  method: 'get',
  data: {
    foo: 'bar'
  }
})
```

Рис. 2.2 – приклад використання Axios

Axios має такі основні методи роботи:

- `axios.delete();`
- `axios.put();`
- `axios.patch();`
- `axios.options();`
- `axios.get();`
- `axios.post();`

У розробці була вибрана бібліотека Axios в якості утиліти для створення API веб-додатку.



### 2.1.10. Cheerio.js

Cheerio - це швидка, гнучка та заощадлива до ресурсів реалізація ядра jQuery, розробленого спеціально для серверної частини веб-додатку. Це бібліотека з відкритим кодом, яка допомагає витягувати відповідні дані із HTML документу.

Cheerio має дуже змістовну документацію та приклади використання конкретних методів. У ньому також є методи модифікації HTML, тому за допомогою даної бібліотеки можна легко додавати або редагувати будь-який елемент в HTML документі.

У веб-додатку Cheerio використовується в якості бібліотеки, за допомогою якої витягуються певні дані з різних веб-джерел для подальшого їх зберігання у базі даних MongoDB.

```

const cheerio = require("cheerio");

const scrapSteam = async () => {
  const steamUrl =
    "https://store.steampowered.com/search/?filter=weeklongdeals";

  const html = await fethHtml(steamUrl);

  const selector = cheerio.load(html);

  // Here we are telling cheerio that the "<a>" collection
  //is inside a div with id 'search_resultsRows' and
  //this div is inside other with id 'search_result_container'.
  //So,'searchResults' is an array of cheerio objects with "<a>" elements
  const searchResults = selector("body")
    .find("#search_result_container > #search_resultsRows > a");

  // Don't worry about this for now
  const deals = results.map((idx, el) => {
    const elementSelector = selector(el);
    return extractDeal(elementSelector)
  })
  .get();

  return deals;
};

```

Рис. 2.3 – приклад використання Cheerio

### 2.1.11. **Mongoose**

Mongoose - це шар об'єктного моделювання документів (ODM), який знаходиться поверх API Node.js MongoDB. Якщо порівнювати з класичними SQL базами, то Mongoose схожий на ORM (об'єктно-реляційне зіставлення).

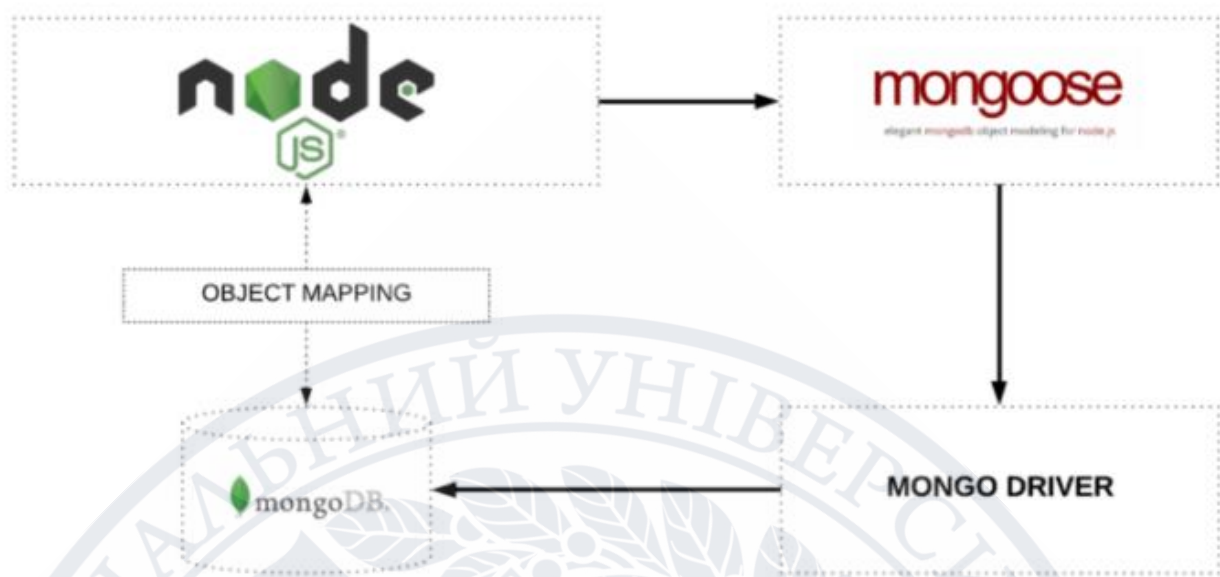


Рис. 2.4 – схема взаємодії Node, MongoDB та mongoose

У додатку дана бібліотека використовується для зв'язку між сервером Node віддаленою базою даних MongoDB. У сучасному веб-додатку, де використовується БД MongoDB, неможливо уявити без бібліотеки mongoose, адже це практично єдиний спосіб для легкого та безпечного керування базою даних.

### 2.1.12. Redux

Redux - це набір утиліт для JavaScript для можливості керування станом веб-додатку. Часто працює у зв'язці з такими фреймворками як React або Angular, для побудови користувацьких інтерфейсів.

Redux - це мала бібліотека з простою реалізацією API призначеним для керуваннями станами програми. Дана бібліотека працює на зменшення і функціонального програмування.

Бібліотека Redux - це спосіб управління станом додатку. Вона заснована на кількох концепціях, з допомогою яких, можна легко вирішувати будь-які



проблеми зі станом. Рекомендується використовувати Redux в середніх і великих додатках. Зміни в стані додатку можливі тільки при відправці на сервер action (дії).

Дія (action) - це об'єкт в додатку, який детально описує зміну в стані додатку.

Сховище (store) - це об'єкт, який:

- тримає в собі стан додатку;
- відображає стан через влаштований метод `getState()`;
- може постійно оновлюватися через `dispatch()`;
- дозволяє бути іншим об'єктам в атрибуті слухача зміни стану через спеціальний метод об'єкту `subscribe()`.

Потік даних в Redux завжди однонаправлений. Передача активностей з потоками даних відбувається через виклик методу `dispatch()` в сховище. Саме сховище передає дії і генерує наступний стан, а потім оновлює стан і повідомляє про це всіх слухачів.

В розробці дана бібліотека використовується через необхідність постійно отримувати інформацію про стан додатку в користувача, наприклад: який розділ вибрав користувач для перегляду новин і потім дану інформацію читає сервер та передає потрібну інформацію з бази даних.

### **2.1.13. GitHub**

GitHub - це платформа, яка користується великою популярністю у спільноти розробників. Це сайт із хостингом із відкритим кодом, який допомагає розробникам зберігати свій код та керувати ним. Дана платформа пропонує розподілений контроль версій через систему Git, а також управління вихідним кодом.

Класи систем, що допомагають розробникам ПО слідкувати та редагувати зміни комп'ютерних програм, веб-сайтів, додатків та документів чи інших даних, відомий як Git.

Програмісти постійно покращують і редагують код додатків після його виходу. З кожним оновленням ПО вони розробляють нову версію першої офіційно випущеної версії програмного забезпечення.

Завдання систем контролю версій - це підтримка цих оновлень, зберігання змін в будь-якому сховищі, або місці, де зберігається весь код для проекту. Даний процес полегшує роботу розробникам над спільними працею над новими версіями. Розробники можуть завантажувати версії зі центрального сховища, редагувати їх та повторно вивантажувати розроблену ними нову версію. Для слідкування над змінами, що зберігаються у центральному хранилищі, можуть отримати доступ інші програмісти коли це буде необхідно.

Алгоритм роботи з GitHub:

1. Встановити git і створити обліковий запис GitHub;
2. Створити локальне сховище git;
3. Додати новий файл до репозиторію;
4. Додати файл до проміжного середовища;
5. Створити запит на зміну коду;
6. Створити нову гілку з кодом;
7. Створити нове сховище на GitHub;
8. Просунути гілку до GitHub;
9. Створити запит на витяг коду;
10. Отримайте зміни на GitHub назад до свого комп'ютера.

При розробці GitHub використовується для зручного зберігання коду, для перегляду будь-якої інформації про зміну коду і, насамперед, для взаємодії хостингу Heroku з лістингом додатку через GitHub.

## 2.1.14. JSON

JSON (Javascript Object Notation) - це формат обміну даними, який використовується для представлення простих структур даних та об'єктів у кодї веб-браузера. JSON також іноді використовують в осередках програмування на робочому столі та на стороні сервера. Спочатку JSON ґрунтувався на мові програмування Javascript і був включений як мова сценаріїв сторінок для веб-браузера Netscape Navigator. JSON – це універсальні структури даних. Фактично всі сучасні мови програмування підтримують. Має сенс, що формат даних, який є взаємозамінним з мовами програмування, також має базуватися на цих структурах.

JSON можна використовувати в Javascript в Інтернеті як заміна формату XML для будови даних. Як і XML, JSON не залежить від мови і може поєднуватися з C ++, Java, Python, Lisp та багатьма іншими мовами. Проте, на відміну від XML, JSON - це лише спосіб представити структури даних, на відміну від повної мови розмітки. Документи JSON відносно легкі і швидко виконуються на веб-сервері.

JSON складається з пар "ім'я: об'єкт" та розділових знаків у вигляді дужок, дужок, крапок з комою та двокрапки. Кожен об'єкт визначається за допомогою оператора типу "text:" або "image:", а потім групується зі значенням для цього оператора. Проста структура та відсутність математичних позначень або алгоритмів JSON легко зрозуміти та швидко освоїти навіть користувачам з обмеженим офіційним досвідом програмування, що спонукало прийняти формат як швидкий доступний спосіб створення інтерактивних сторінок.

В додатку даний формат файлу використовується для зберігання новин у базі даних, та подальшої передачі їх на веб-сайт для перегляду користувачам.



### 2.1.15. Path

Модуль path надає утиліти для роботи з шляхами до файлів і тек. До нього можна отримати доступ таким чином:

- `const path = require('path')`

Дана утиліта використовується в розробці веб-додатку для взаємодії отриманих даних з веб-джерел з новинами для перетворення тексту в JSON файли та подальшу відправку їх у базу даних.

### 2.1.16. Visual Studio Code

Visual Studio Code - це безплатна програма для редагування коду, який був розроблений компанією Microsoft для усіх популярних операційних систем. Особливостями можна відмітити те, що даний редактор коду включає функцію налагодження, висвітлення синтаксису, розумне авто-заповнення програмного коду, частини коду, ре факторинг коду та влаштований за замовченням Git. Розробники мають змогу міняти тему, гарячі клавіші, параметри та встановлювати розширення, що надають додатковий функціонал.

Як аналог системи додатків даний редактор коду дозволяє розробникам писати код в одному або кількох каталогів, які згодом можливо зберегти у вибраних теках для повторного написання коду. Дана функція дозволяє розробляти додатки на різних мовах програмування. Visual Studio Code має можливість підтримки для усіх мов програмування.

Керування джерелом коду - це вбудована функція Visual Studio Code. Даний редактор коду має спеціальну вкладку всередині рядка меню, де можна одержати доступ до параметрів контролю версій та дивитись на зміни, які були

внесені до проекту. Щоб застосувати дану функцію, необхідно зв'язати Visual Studio Code з будь-якою підтримуваною системою контролю версій (Git, Subversion, Perforce тощо). Дана функція надає можливість створювати простори, робити усі запити та команди до даних сховищ безпосередньо з програми.

В практичній частині бакалаврської роботи Visual Studio Code використовується як додаток для написання та рефакторингу коду, оскільки він має безліч функцій, необхідності використовувати інші додатки для написання та редагування коду - немає.

## **2.2 Поняття односторінкового додатку**

Термін "односторінковий додаток" або SPA застосовується для характеристики певного типу програмного забезпечення, яке розроблено та побудовано для ефективної роботи в Інтернеті.

Як і до інших веб-сайтів, до SPA-центрів можна отримати доступ через веб-браузер, але вони мають можливість надавати більш динамічну взаємодію з користувачем, подібно до того, що можна очікувати від власного мобільного додатку чи настільного додатка.

SPA відрізняються від традиційних веб-сайтів тим, що вони сильно зменшують кількість оновлень різних сторінок, необхідних для забезпечення бажаної взаємодії користувачів з веб-додатком. Це досягається через використання AJAX - інструмент, який дозволяє сайту обмінюватися інформацією із серверними серверами та завантажувати дані в додаток, не виконуючи повного оновлення сторінки.

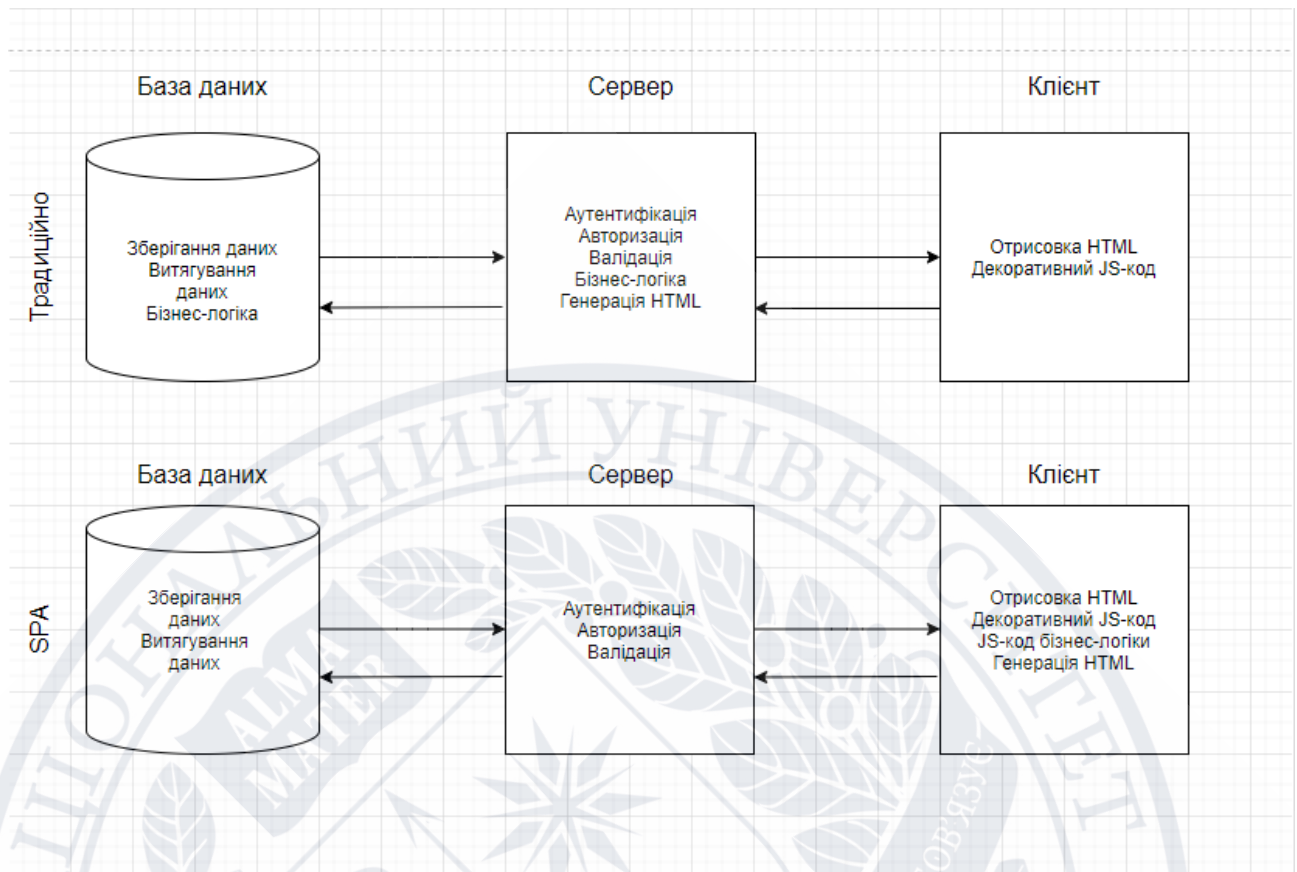


Рис. 2.4 Схеми порівняння роботи односторінкового додатку та традиційного підходу до створення веб-сайтів

Більш безпосередньо, програма на одній сторінці передає більшу частину роботи з обробкою на сторону клієнта, що дозволяє користувачеві завантажувати одну сторінку HTML і динамічно оновлювати вміст на цій сторінці, не оновлюючи його, оскільки вони взаємодіють із додатком.

SPA - це чудовий варіант для забезпечення чудового користувацького досвіду в браузері, оскільки вони можуть забезпечити інтуїтивно зрозумілі користувацькі веб-інтерфейси без перезавантаження сторінок з незначним або відсутнім часом очікування.

Багатосторінкові програми - це більш «традиційний» спосіб забезпечення динамічного користувацького досвіду всередині веб-браузера.

Однією з найважливіших переваг SPA є їх здатність завантажувати всі дані за один запит. Ця структура гарантує, що користувачеві не потрібно робити кілька запитів на сервер кожного разу, коли він хоче бачити нову



інформацію, оскільки весь вміст програми надходить до клієнта за один запит. Оскільки весь вміст, шаблон та дані попередньо завантажені, SPA може забезпечити швидшу навігацію сторінками, перегляд та продуктивність та забезпечити кращий досвід для користувачів.

Процеси розробки, розгортання та налагодження також полегшують SPA для розробників. Ці програми не часто вимагають кодування на стороні сервера і можуть бути розгорнуті на сервері, подібно до типової веб-сторінки. Розробники також можуть налагоджувати односторінкові програми за допомогою веб-браузера Google Chrome, який надає користувачам можливість контролювати мережеві операції, досліджувати окремі елементи сторінки та переглядати дані, підключені до кожного елемента.

Незважаючи на те, що SPA-послуги швидкі та прості в експлуатації, ця швидкість коштує платно, але велика кількість вмісту в одній сторінці програми може сповільнити завантаження на початковий запит сервера. Якщо завантаження сторінки триває занадто довго, клієнт може втратити перегляди та конверсії. Крім того, оскільки SPA-служби сильно залежать від JavaScript, щоб увімкнути свої функції, користувачі, які вимкнули JavaScript у своїх браузерах, можуть не мати доступу до усіх функцій та функцій програми.

Іншим значним мінусом є те, що оптимізація SPA для пошукових систем може бути проблематичною. Використання AJAX (асинхронний JavaScript та XML) для завантаження вмісту без виклику серверу для оновлення сторінки ускладнює оптимізацію вмісту для сканерів пошукової системи, що може призвести до зниження рейтингу пошукової системи та зниження видимості. Але дане питання постійно вивчається та переглядається спільнотою розробників.

Соціальні мережі та веб-сайти, які відображають стрічки новин, часто використовують односторінкові програми для впорядкування взаємодії з клієнтами. Однак стрічки новин - це не єдині приклади SPA. Ось короткий список SPA:

- Facebook;
- LinkedIn;
- Twitter;
- Gmail;
- Netflix.

Переваги SPA перед традиційними веб-додатками:

- SPA-послуги швидкі, оскільки більшість ресурсів (наприклад: HTML, CSS, скрипти) завантажуються лише один раз протягом усього терміну служби програми;
- Розробка продукту проста і впорядкована. Вам не потрібно писати код, щоб відображати сторінки на сервісі. Насправді, як правило, можна розпочати роботу, не використовуючи жодного сервера;
- Ви можете налагоджувати односторінкові програми за допомогою Chrome, який дозволяє стежити за мережевими операціями, перевіряти елементи сторінки та перевіряти дані, пов'язані з окремими елементами;
- SPA відправляють один запит сервера, зберігають дані локально та зберігають можливість використовувати ці дані знову і знову без іншого запиту, навіть коли користувач перебуває в автономному режимі.

### 3. ОПИС РОЗРОБКИ ТА ПРОЕКТУВАННЯ ВЕБ-ДОДАТКУ

#### 3.1 Загальна структура веб-додатку

Загальна структура веб-додатку виглядає наступним чином (рис.3.1):

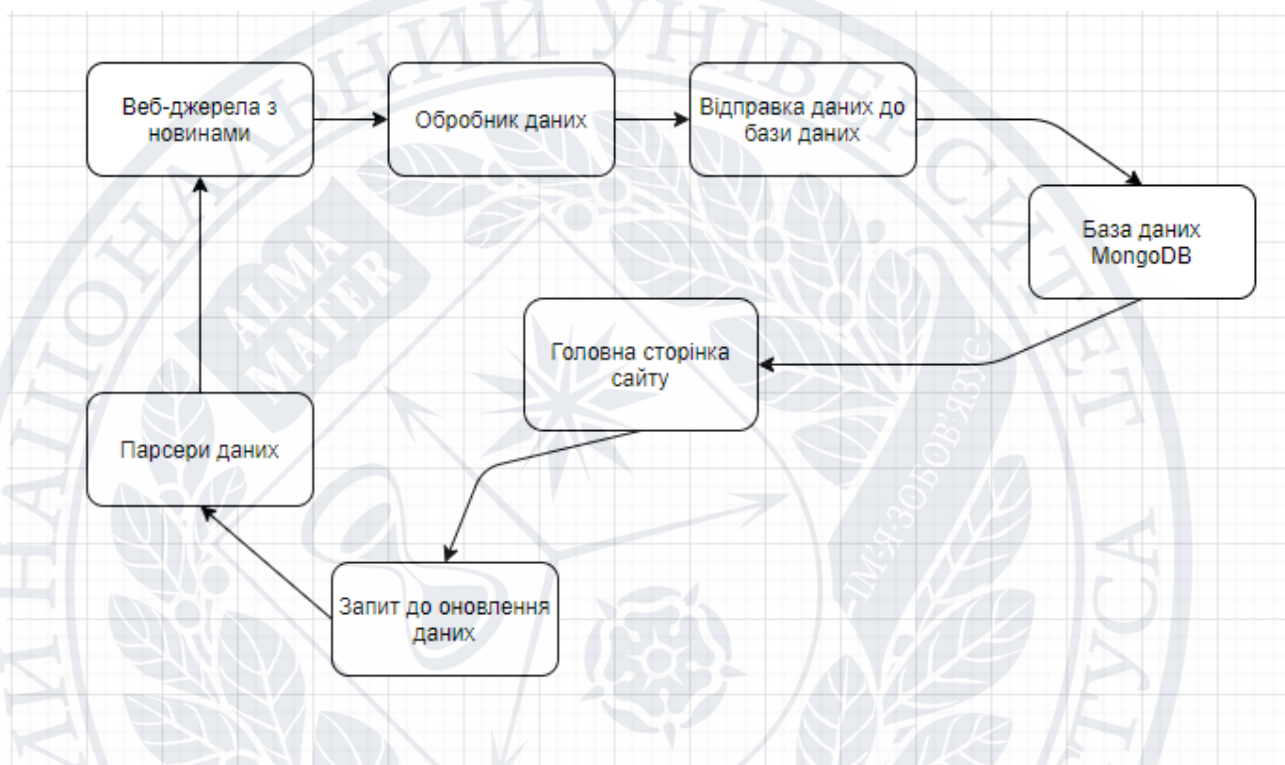


Рис. 3.1 – Структура веб-додатку

На головній сторінці знаходиться «шапка» сайту, поле з назвою вибраного розділу, зручна таблиця з новинами та впливаюче меню, за допомогою якого користувач може вибрати один з шести таких розділів:

1. Новини України;
2. IT і наука;
3. Новини світу;
4. Коронавірус;
5. Експертна думка;
6. Донецький Національний Університет ім. Василя Стуса;
7. Війна.



### 3.2 Розробка веб-інтерфейсу

Відповідно до загальної структури, представленої в розділі 3.1, був спроектований сайт. Головна сторінка сайту (Рисунок 3.2) містить всі елементи, зазначені в структурі, всі переходи здійснюються за допомогою стану (state) веб-додатку.

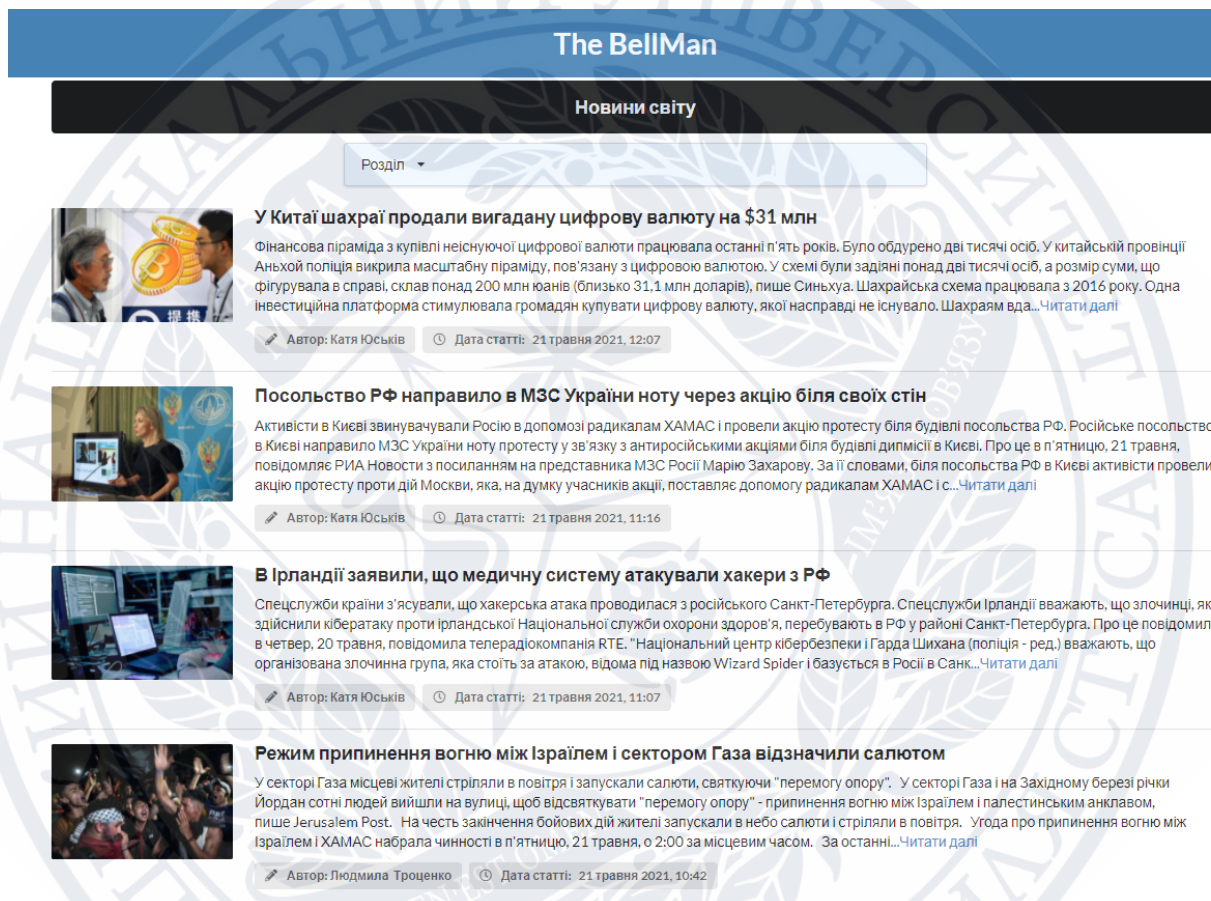


Рис. 3.2 – Головна сторінка веб-додатку

Весь макет та дизайн створений за допомогою React.js, Redux та бібліотеки SemanticUI. Використовуючи фреймворк React, можна розділити будь-яку частину користувацького інтерфейсу на компоненти. Це дозволяє легко ними керувати, редагувати та масштабувати в цілому весь додаток.

```

1 import React from 'react';
2 import 'semantic-ui-css/semantic.min.css'
3 import { Item, Label, Link } from 'semantic-ui-react';
4
5 const Post = ({ title, description, preDescription, authorName, image, url, time, readNext }) => {
6   return (
7     <Item>
8       <Item.Image src={image} />
9       <Item.Content>
10        <Item.Header as='a' href={url}><title></Item.Header>
11        <Item.Description><description><Item.Description as='a' href={url}><readNext></Item.Description></Item.Description>
12        <Item.Extra>
13          <Label icon='pencil alternate icon' content={`Автор: ${authorName}`} />
14          <Label icon='clock outline icon' content={`Дата статті: ${time}`} />
15        </Item.Extra>
16      </Item.Content>
17    </Item>
18  );
19 };
20
21 export default Post;

```

Рис. 3.3 – код шаблону новинного посту створеного за допомогою SemanticUI

З фреймворком React, веб-додаток стає легко-масштабованим, що є великою перевагою перед іншими рішеннями для створення користувацького інтерфейсу, так як тематика веб-додатку - це перегляд новин, а заздалегідь не можна знати скільки новинних постів буде представлено в інтерфейсі після кожного оновлення даних.

```

148   return (
149     <Container>
150       <div class="main"></div>
151       <div class="footer">
152         <h1 class="h1">The BellMan</h1>
153       </div>
154       <div class="ui inverted segment"><h3 className="secondHeader">{ this.genreText(this.props.genre.genre) }</h3> </div>
155     </div>
156     <div class="ui compact menu">
157       <div class="ui simple dropdown item">
158         Розділ
159         <i class="dropdown icon"></i>
160       <div class="menu">
161         <div class="item" onClick={() => this.props.changeGenre('UKR')} && this.fetchPostsHromadske() && $(''.menu').hide())>Україна</div>
162         <div class="item" onClick={() => this.props.changeGenre('IT')} && this.fetchPostsUnian())>IT</div>
163         <div class="item" onClick={() => this.props.changeGenre('WORLD')} && this.fetchPostsKoresp())>Cair</div>
164         <div class="item" onClick={() => this.props.changeGenre('COR')} && this.fetchPostsKorona())>Коронавірус</div>
165         <div class="item" onClick={() => this.props.changeGenre('EXP')} && this.fetchPostsExpert())>Експертна думка</div>
166         <div class="item" onClick={() => this.props.changeGenre('DONNU')} && this.fetchPostsDonnu())>Донецький Національний Університет ім. Василя Стуса</div>
167       </div>
168     </div>
169   </div>
170 </div>
171 <Item.Group divided>
172   {(items.map(({ url, title, description, preDescription, authorName, image, time, readNext }, key) => [
173     <Post
174       key={key}
175       readNext={readNext}
176       time={time}
177       url={url}
178       title={title}
179       preDescription={` ${preDescription.substring(0, 540)} `}
180       description={` ${description.substring(0, 540)} `}
181       authorName={authorName}
182       image={image}
183     </>
184   ])}
185 </Item.Group>

```

Рис. 3.4 – Приклад коду на React

Використана, для розробки веб-інтерфейсу, бібліотека Redux дозволяє нам оновлювати дані через стан додатку. Це є перевагою, так як ми можемо оновлювати дані, не оновлюючи саму сторінку сайту, що збільшує швидкодію веб-додатку та дозволяє економити трафік користувачам.

Дизайн адаптивний для будь-яких пристроїв, таких як:

- Персональний комп'ютер;
- Планшет;
- Мобільний телефон;
- Телевізор зі смарт ТВ;
- Будь-які пристрої які мають змогу виходити в браузер.



# The BellMan

Україна

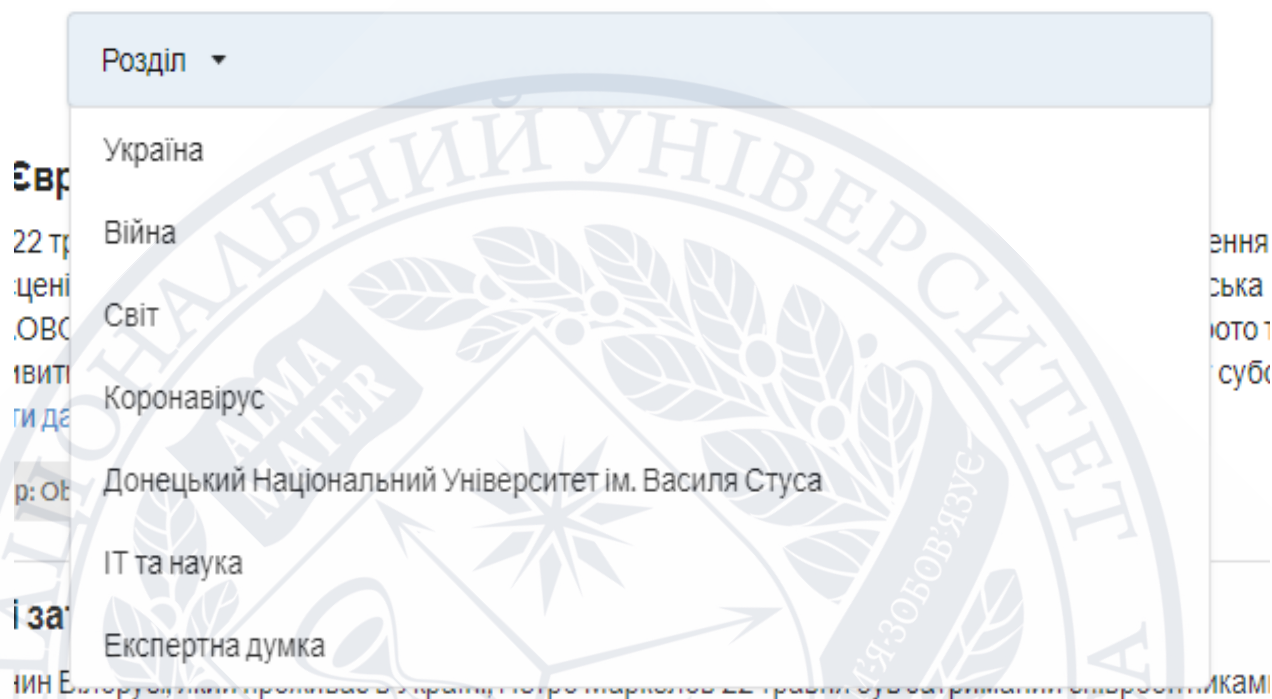


Рис. 3.5 – вибір розділу новин



Рис. 3.6 – мобільний вигляд веб-додатку

### 3.3 Реалізація серверу та хостингу

При перегляді веб-сторінок у браузері користувач відправляє запит на інший комп'ютер в Інтернеті, який надсилає у відповідь на веб-сторінку. Комп'ютер, з яким взаємодія відбувається через інтернет, називається веб-сервером. Веб-сервер отримує запрошення HTTP від клієнтів, у тому числі від вашого браузера, і відправляє відповіді HTTP, наприклад сторінки HTML або код JSON з API.

Для виводу веб-сторінок сервер використовує різноманітне програмне забезпечення. Програмне забезпечення може бути клієнтським або серверним. Клієнтське програмне забезпечення відповідає за виведення контенту, наприклад для кольорових панелей навігації та стилю тексту. Серверне програмне забезпечення відповідає за обмін, обробку та зберігання даних. Для обробки мережових запитів браузера та взаємодії з базами даних в основному відповідає серверний код.

Node.js дозволяє розробникам використовувати JavaScript для створення серверного коду, хоча традиційно дана мова програмування використовувалась у браузері для створення клієнтського коду. Об'єднання клієнтського та серверного коду в одному середовищі розробки спрощує створення веб-серверів, а саме тому Node.js став популярним інструментом для написання серверного коду.

Для створення серверної частини веб-додатку використовується такий стек технологій:

- Node.js;
- MongoDB;
- Express.js.

За допомогою Express.js створюються шляхи за якими користувач може переходити та отримувати дані для перегляду на своєму браузері, потім



обробник запитів, також Express.js, читає запит та відправляє необхідну інформацію в користувацький інтерфейс.

Фреймворк React дозволяє створювати додатки без серверної частини додатку, але такий метод не дозволяє використовувати в процесі розробки користування базою даних, що є необхідністю для поставленої мети. Тому для розробки веб-додатку використовується Node.js та інші утиліти для правильної роботи серверу.

Node.js в свою чергу дозволяє налаштовувати взаємодію між бібліотекою для творення користувацького інтерфейсу React та утилітою для створення серверу Express.js, що дозволяє легко писати код в одному середовищі.

Важливою частиною всієї системи є сховище, яке в даному випадку є REST-сервіс.

REST - це набір архітектурних принципів і стиль проектування додатків, орієнтований на створення мережевих систем, в основі яких лежать механізми для опису і звернення до ресурсів. прикладом такої системи може служити World Wide Web.

У REST визначається суворий поділ відповідальності між компонентами клієнт-серверної системи, що полегшує реалізацію необхідних акторів (actors). Іншою метою REST є спрощення семантики взаємодії компонентів мережевих систем, що дозволяє поліпшити масштабованість і підвищити продуктивність. В основу REST закладений принцип автономності запитів, що означає, що запити, оброблювані клієнтом або сервером, повинні включати всю контекстну інформацію, необхідну для їх розуміння. При роботі REST-систем для обміну даними стандартних медіа-типів використовується мінімальна кількість запитів.

REST-системи використовують URI (універсальні ідентифікатори ресурсів) для пошуку та отримання доступу до уявлених необхідних ресурсів.

Протягом останніх декількох років розробники створювали REST сервіси для своїх веб-додатків, використовуючи найрізноманітніші технології.

Архітектура REST відрізняється своєю простотою, вимагаючи від додатків забезпечити тільки можливість прийому повідомлень з HTTP заголовками. Ця функція легко реалізується простими контролерами в ASP.NET MVC.

У веб-додатку взаємодія між всіма компонентами відбувається наступним чином: React код перетворюється в один файл `index.js` (build), та відправляється на сервер за запитом, який читає `Express.js`. В свою чергу створена структура додатку відправляється на сервіс контролю версій (GitHub), і звідти все читає сервіс для хостингу Heroku.

Heroku дозволяє виставляти на хостинг не сильнонавантажені веб-додатки без будь-яких обмежень. Даний хостинг-сервіс дає багато утиліт для керування веб-додатком, перегляду усіх логів, і в разі збою будь-якого компоненту, відправляє електронний лист сповіщаючи що в додатку відбувся збій.

### 3.4 Реалізація парсингу даних

Так як серверна частина додатку написана на мові JavaScript, яку усі знають через підтримку менеджера пакетів `npm`, який в свою чергу містить безліч компонентів та бібліотек, в тому числі для вирішення задачі парсингу інтернет-сторінок, розумно використати якусь з них для реалізації власного парсинг-модуля. В результаті пошуку та вибору потрібної бібліотеки, було вирішено використовувати дві бібліотеки: `Unirest` – для створення запитів до веб-джерел та `Cheerio` – безпосередньо для парсингу отриманих даних.

Під час парсингу веб-джерел, сканер шукає відповідні HTML-елементи, та вносить їх до бази даних. Сканер «втягує» наступні дані з статті з новиною:

- Заголовок;
- Текст статті;
- Автор статті;

- Головне зображення. Якщо відсутнє – автоматично вноситься зображення по замовченню відповідно до теми новини;
- Час статті.

Для зручності текст статті скорочено до 300 символів. Якщо користувача зацікавила стаття, він може натиснути на заголовок або словосполучення «Читати далі» розташоване в кінці скороченого тексту статті, що виділене кольором для зручності та розуміння, і перейти на оригінал статті та прочитати її повністю.

Модуль парсера повинен працювати зі всіма типами кодування, таким чином це забезпечує від «витягування» незрозумілих символів. Якщо будь-який з елементів необхідний для формування посту новини у веб-додатку виявляється пустим або недоступним, ітерація в циклі парсера пропускається, і паралельно запускається друга спроба для даного посилання з новою. Далі представлений алгоритм роботи парсер-модуля (рис. 3.4):



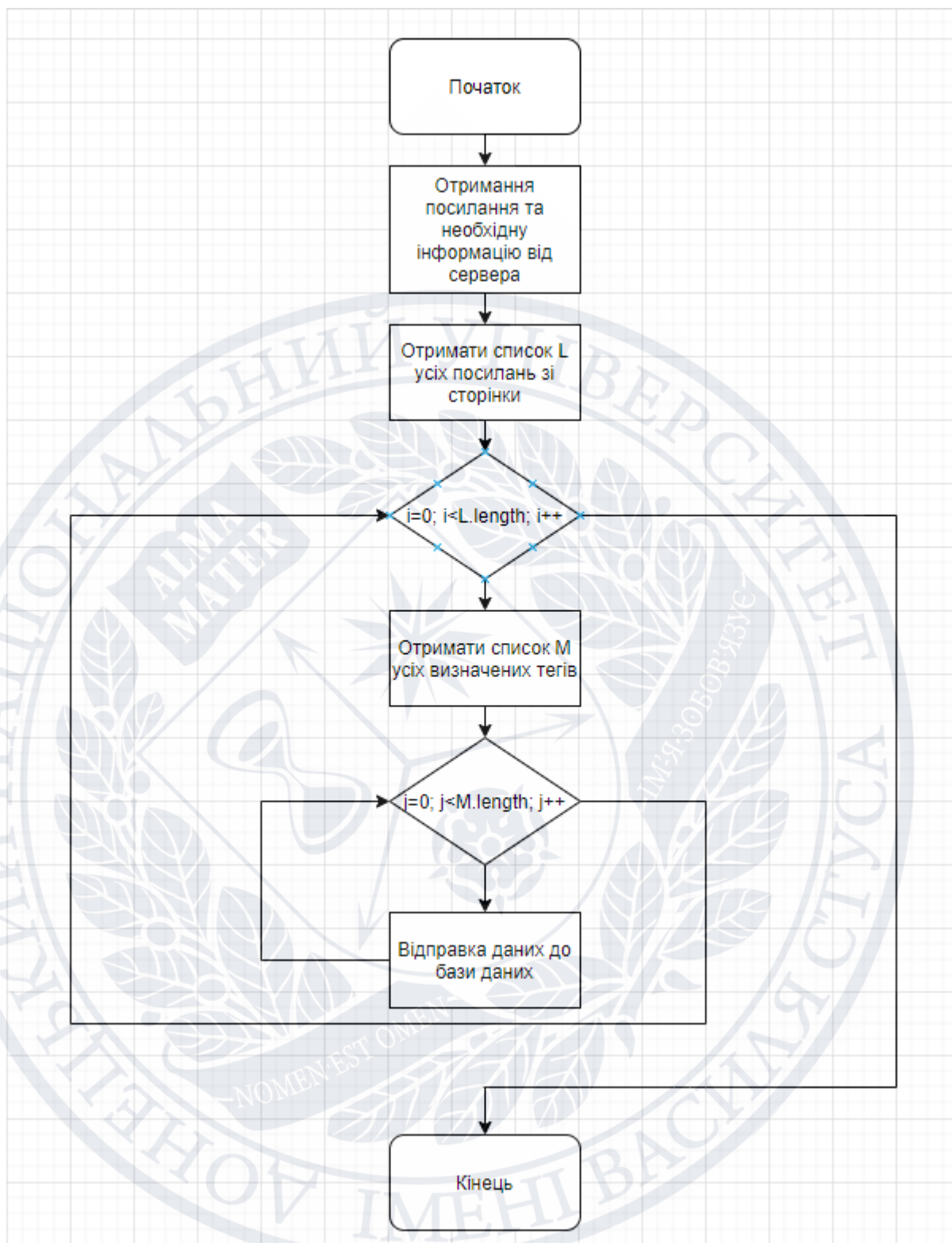


Рис. 3.7 – алгоритм роботи модуля-парсера

В даній роботі бібліотеки Cheerio та Unirest були налаштовані таким чином, що у разі виникнення будь-якої помилки, сканер не перестає працювати, а продовжує збирати дані з інших джерел, в результаті був отриманий безперебійний, універсальний модуль парсингу даних, який підходить для

будь-якого веб-джерела, і написання коду для нового сайту з новинами займе не більше години роботи, що дозволяє масштабувати веб-додаток у будь-які розміри.

```

19 function parseNewUnian(url, elems) {
20   return new Promise((resolve, reject) => {
21     unirest.get(url).end(({ body, error }) => {
22       const $ = cheerio.load(body);
23       let id = 0
24
25       const title = $(elems.title).text().trim()
26       const preDescription = $(elems.preDescription).text().trim()
27       const description = $(elems.description).text().trim()
28       const authorName = $(elems.authorName).text().trim()
29       const image = $(elems.image).attr('src')
30       const time = $(elems.time).text().trim()
31       const readNext = '...Читати далі'
32
33       const post = {
34         id: id,
35         readNext: readNext,
36         time: time,
37         url: url,
38         title: title,
39         preDescription: preDescription,
40         description: description,
41         authorName: authorName,
42         image: image
43       }
44       if (error) {
45         return reject(error);
46       }
47       resolve(post);
48     });
49   });
50 }

```

Рис. 3.8 – приклад коду з Cheerio та Unirest

Взаємодія Cheerio та Unirest дає широкий вибір утиліт та функцій для забезпечення проміжного рівня для обробки даних:

- Cookie та підтримка оброблення сеансів;
- HTTP-методи, такі як кешування, аутентифікація та стиснення;
- Підміна User-Agent параметру, що забезпечує високий рівень безперебійності сканеру;
- Обмеження глибини сканування.

Запуск сканерів даних по веб-джерелам відбувається повністю автоматично (можливий запуск вручну локально або через консоль веб-хостингу Heroku з правами адміністратора) в таких випадках:

- Щогодини;
- Користувач зайшов у веб-додаток та в цей час сканер не запущений;
- Якщо сканер отримав будь-яку помилку, відбувається паралельний перезапуск сканерів даних.

Cheerio являє собою аналог jQuery для роботи з Node.js, для вибору тегів HTML-документа використовуються селектори. Синтаксис селектора був запозичений з jQuery. Шукаємо селектор заголовків новин і посилання на нього за допомогою Chrome DevTools.

В першу чергу потрібно довантажити потрібний HTML. Цей крок в jQuery є прихованим, оскільки jQuery працює на одній вбудованій DOM. Тепер в Cheerio потрібно передати HTML-документ. Після завантаження HTML, потрібно повторити всі рядки таблиці `<tr>`, щоб витягти кожну новину на сторінці.

Багато веб-сайтів використовують архітектуру односторінкових додатків (SPA) для динамічного створення контенту на своїх веб-сайтах за допомогою JavaScript. Ми можемо отримати відповідь від початкового HTTP-запиту і не можемо запустити javascript для візуалізації динамічного вмісту за допомогою axios і інших подібних пакетів npm, таких як request. Отже, витягувати дані можна тільки з статичних веб-сайтів.

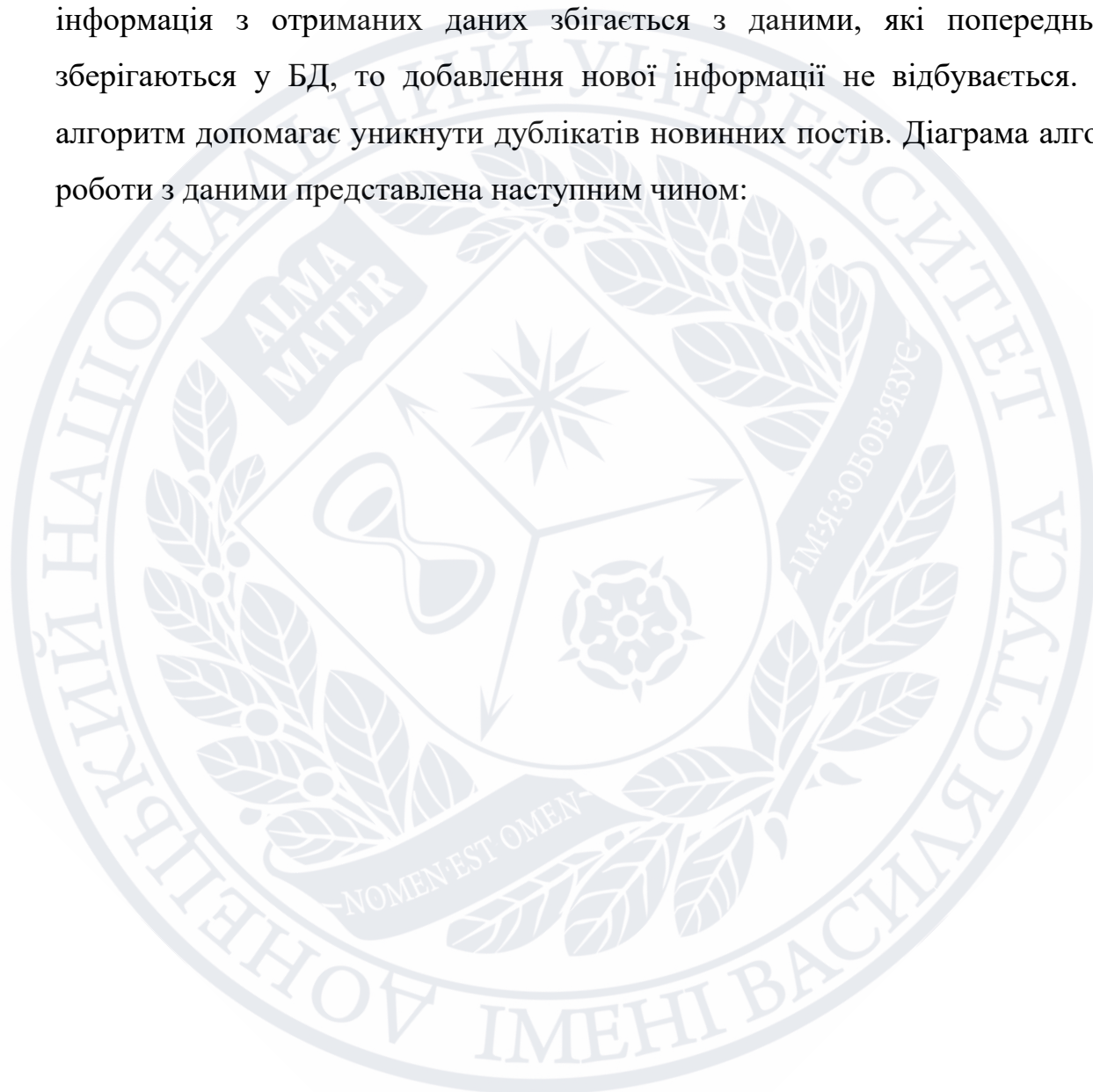
### **3.5 Обробка даних та робота з отриманою інформацією**

У випадку, коли сканер запускається, він збирає усі дані та тимчасово зберігає всі дані у локальному сховищі на сервері. Після успішного завершення парсера даних усі дані відправляються у віддалену базу даних MongoDB за завчасно створеною схемою вигляду новинного посту, та видаляє всю



інформацію з локального сховища. У свою чергу усі дані в базі даних відправляються до користувацького інтерфейсу веб-додатку, для користувацького перегляду новин.

Під час відправлення даних до БД, усі попередні дані, які вже зберігаються у базі даних MongoDB скануються на унікальність. Якщо будь-яка інформація з отриманих даних збігається з даними, які попередньо вже зберігаються у БД, то додавання нової інформації не відбувається. Даний алгоритм допомагає уникнути дублікатів новинних постів. Діаграма алгоритму роботи з даними представлена наступним чином:



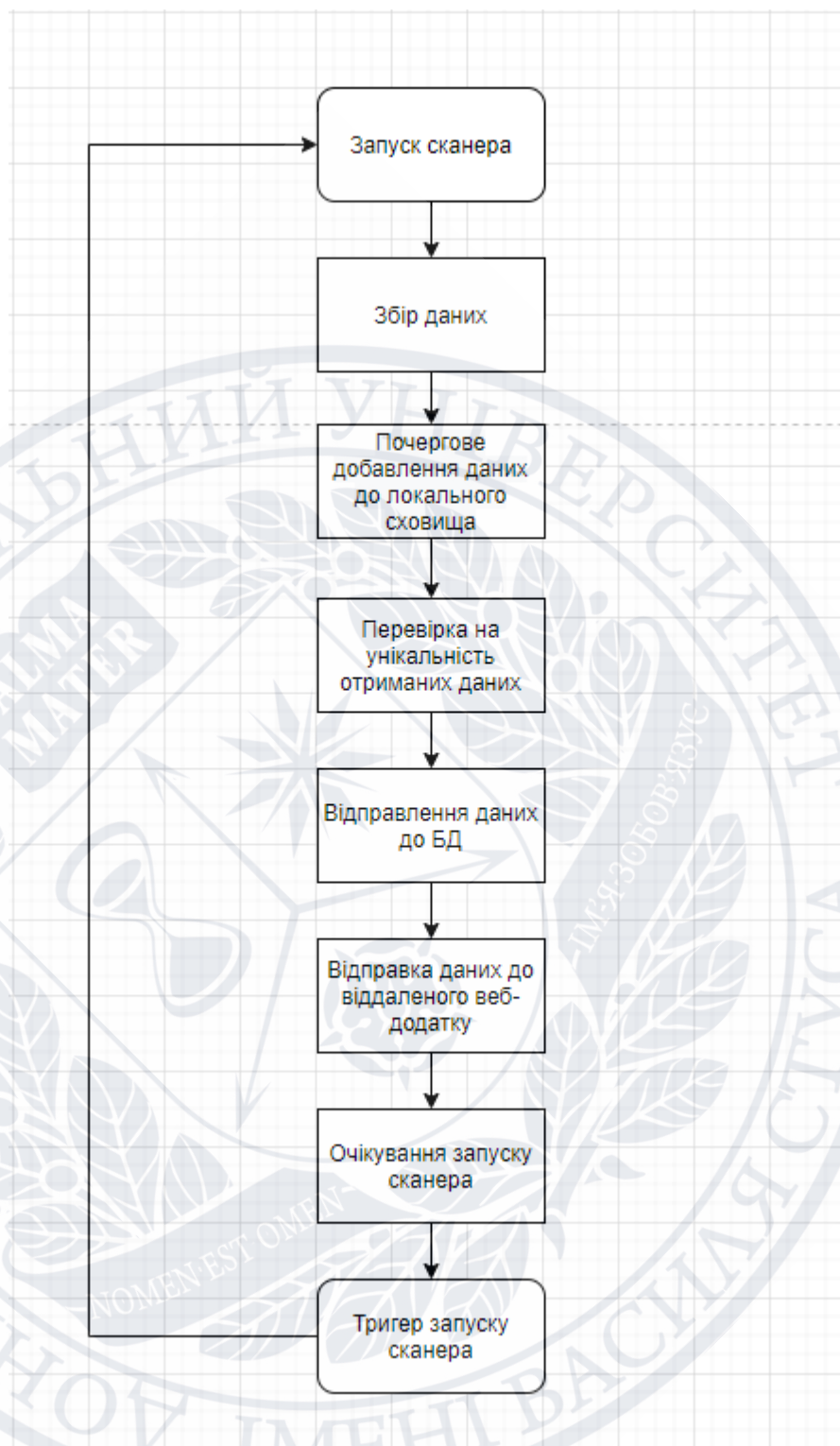


Рис. 3.9 – Алгоритм роботи з даними

Підбиваючи підсумки алгоритму роботи з сканерами та отриманими даними, можна сказати що функціонал працює надійно та безперебійно. У випадку якщо веб-джерело, до якого відбувається звернення сканера не відповідає, ітерація пропускається та збою в системі не відбувається.

Після будь-якого збою в роботі парсерів, запускається інший сканер, який повторить спробу отримати дані з веб-джерела, яке було недоступне під час роботи попереднього сканера.

### **3.6 Взаємодія користувача та веб-додатку**

Після переходу користувача до серверу веб-додатку, відбувається перевірка чи запущений на даний момент сканер даних, якщо сканер не запущений, він запускається. Тривалість сканеру приблизно 1-2 хвилини. Під час очікування користувач може переглянути новини за попередню годину (в кожного новинного поста присутня дата написання статті в оригінальному веб-джерелі) в будь-якому з семи розділів.

Під час зберігання нових даних до бази даних, у активного користувача відбувається оновлення даних до актуальних. Можливі випадки, коли на якомусь з веб-джерел нових статей виставлено не було, тому оновлення даних по даному веб-джерелу не відбувається.

Можна підсумувати, що користування даним веб-додатком є дуже простим, адже необхідно лише перейти за посиланням і відразу з'являються новини з розділу, який вибраний за замовченням. Також можна вибрати у списку розділ, який зацікавив користувача і новини будуть готові до перегляду. Така простота є перевагою, адже люди з будь-яким досвідом з роботою ПК або іншого пристрою зможуть легко розібратись в користуванні даним веб-додатком.



## ВИСНОВОК

В атестаційній роботі представлені результати, котрі відповідають поставленій меті і є вирішенням задачі проектування автоматизованої системи збору новин. Для вирішення задачі було проаналізовано аналоги, пошук достовірних веб-джерел та вирішення недоліків представлених джерел. Оскільки формою кінцевої реалізації вибраний веб-додаток, то тому було розглянуто приклади веб-додатків на тему «новинний агрегатор» з виділенням головних властивостей кожного з переглянутих додатків.

Далі було представлено інструментарій для розробки з обґрунтованим вибором технологій. Для серверної частини додатку був обраний JavaScript фреймворк Node.js, бібліотеки Unirest і Cheerio були вибрані як основні технології для розробки модуля-парсера, користувацький веб-інтерфейс був спроектований в мінімалістичному стилі з використанням фреймворку React.js та шаблонів SemanticUI.

Основні функції агрегатора новин, такі як можливість обрати розділ новинної стрічки, актуальність новин та масштабованість інтерфейсу були реалізовані. Надалі доцільно буде розширити можливості додатку таких як, врахування вподобань користувача при використанні системи. Наприклад покращення фільтрації, додавання варіантів вибору відображення оновлень, додаванню соціальних можливостей тощо.

## ЕЛЕКТРОННІ РЕСУРСИ ВІДДАЛЕНОГО ДОСТУПУ

1. React.js [Електронний ресурс] – Режим доступу до ресурсу:

<https://reactjs.org/tutorial/tutorial.html>

2. Creating of web server in Node.js [Електронний ресурс] – Режим доступу до ресурсу:

<https://www.digitalocean.com/community/tutorials/how-to-create-a-web-server-in-node-js-with-the-http-module-ru>

3. Visual Studio Code [Електронний ресурс] – Режим доступу до ресурсу:

[https://en.wikipedia.org/wiki/Visual\\_Studio\\_Code](https://en.wikipedia.org/wiki/Visual_Studio_Code)

4. JavaScript Object Notation [Електронний ресурс] – Режим доступу до ресурсу:

<https://www.json.org/json-en.html>

5. Path.js [Електронний ресурс] – Режим доступу до ресурсу:

<https://js-node.ru/site/article?id=30>

6. GitHub [Електронний ресурс] – Режим доступу до ресурсу:

<https://www.freecodecamp.org/news/the-beginners-guide-to-git-github/>

7. Tutorial for GitHub [Електронний ресурс] – Режим доступу до ресурсу:

<https://product.hubspot.com/blog/git-and-github-tutorial-for-beginners>

8. Redux [Електронний ресурс] – Режим доступу до ресурсу:

<https://www.valentinog.com/blog/redux/>

9. Redux for beginners [Електронний ресурс] – Режим доступу до ресурсу:

<https://tproger.ru/translations/redux-for-beginners/>

10. Web scrapping with Node.js and Cheerio [Електронний ресурс] – Режим доступу до ресурсу:

[https://dev.to/diass\\_le/tutorial-web-scraping-with-nodejs-and-cheerio-2jbh](https://dev.to/diass_le/tutorial-web-scraping-with-nodejs-and-cheerio-2jbh)

11. Cheerio tutorial [Електронний ресурс] – Режим доступу до ресурсу:

<https://zetcode.com/javascript/cheerio/>

12. Axios Node.js [Електронний ресурс] – Режим доступу до ресурсу:

<https://flaviocopes.com/node-axios/>

13. Google News [Електронний ресурс] – Режим доступу до ресурсу:

[https://en.wikipedia.org/wiki/Google\\_News](https://en.wikipedia.org/wiki/Google_News)

14. Flipboard [Електронний ресурс] – Режим доступу до ресурсу:

<https://en.wikipedia.org/wiki/Flipboard>

15. Ukr.net [Електронний ресурс] – Режим доступу до ресурсу:

<https://uk.wikipedia.org/wiki/Ukr.net>

16. JavaScript [Електронний ресурс] – Режим доступу до ресурсу:

<https://developer.mozilla.org/en-US/docs/Web/JavaScript>

17. MERN Stack [Електронний ресурс] – Режим доступу до ресурсу:

<https://www.mongodb.com/mern-stack>

18. MongoDB [Електронний ресурс] – Режим доступу до ресурсу:

<https://docs.mongodb.com/manual/tutorial/getting-started/>

19. Express.js [Електронний ресурс] – Режим доступу до ресурсу:

<https://expressjs.com/en/guide/routing.html>

20. Heroku [Електронний ресурс] – Режим доступу до ресурсу:

<https://devcenter.heroku.com/categories/reference>



21. SemanticUI [Електронний ресурс] – Режим доступу до ресурсу:

<https://semantic-ui.com/introduction/getting-started.html>

22. Mongoose.js [Електронний ресурс] – Режим доступу до ресурсу:

<https://mongoosejs.com/docs/>



---

Прізвище, ім'я, по-батькові

---

Факультет

---

Шифр і назва спеціальності

---

Освітня програма

## ДЕКЛАРАЦІЯ

Усвідомлюючи свою відповідальність за надання неправдивої інформації, стверджую, що подана кваліфікаційна (бакалаврська) робота на тему:

« \_\_\_\_\_ »  
\_\_\_\_\_»

є написаною мною особисто.

Одночасно заявляю, що ця робота:

- не передавалась іншим особам і подається до захисту вперше;
- не порушує авторських та суміжних прав, закріплених статтями 21-25 Закону України «Про авторське право та суміжні права»;
- не отримувались іншими особами, а також дані та інформація не отримувались у недозволений спосіб.

Я усвідомлюю, що у разі порушення цього порядку моя кваліфікаційна (бакалаврська) робота буде відхилена без права її захисту, або під час захисту за неї буде поставлена оцінка «незадовільно».

---

дата

---

підпис здобувача