

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіoeлектроніки,

Факультет комп'ютерних наук і технологій

(повне найменування інституту, назва факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему РОЗРОБКА ВЕБЗАСТОСУНКУ ДЛЯ ПЕРЕГЛЯДУ ТА ВЗАЄМОДІЇ
З ІНФОРМАЦІЙНИМ ПОРТАЛОМ
WEB APPLICATION DEVELOPMENT FOR VIEWING AND
INTERACTING WITH THE INFORMATION PORTAL

Виконав: студент 4 курсу, групи КНТ-137
Спеціальності 121 Інженерія програмного
забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

Боберський А.Я.

(прізвище та ініціали)

Керівник Гладкова О.М.

(прізвище та ініціали)

Рецензент Морщавка С.В.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
 (повне найменування закладу вищої освіти)

Інститут, факультет ІРЕ, ФКНТ
 Кафедра програмних засобів
 Ступінь вищої освіти бакалавр
 Спеціальність 121 Інженерія програмного забезпечення
 (код і найменування)
 Освітня програма (спеціалізація) Інженерія програмного забезпечення
 (назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
 “ ” 20 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Боберського Андрія Ярославовича

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Розробка вебзастосунку для перегляду та взаємодії з інформаційним порталом. Web Application Development for Viewing and Interacting with the Information Portal.

керівник проєкту (роботи) Гладкова Ольга Миколаївна, к.т.н.,
 (прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “30” березня 2021 року № 103

2. Строк подання студентом проєкту (роботи) 5 травня 2021 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Розробка архітектури програми. 3. Основні рішення щодо реалізації компонентів системи. 4. Інструкції по використанню вебзастосунку.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	Гладкова О.М., доцент		
Нормоконтроль	Дейнега Л.Ю., ст. викладач		

7. Дата видачі завдання “28” лютого 2021 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області	2-3 тижні	Розділ 1
3	Розробка архітектури програми	4 тиждень	Розділ 2
4	Розробка програми	5-6 тижні	Розділ 3
5	Тестування та експериментальне дослідження програми	7 тиждень	Розділ 4
6	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	8 тиждень	Додатки
7	Захист роботи	9 тиждень	

Студент(ка)

(підпис) Боберський А.Я.
(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис) Гладкова О.М.
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 89 с., 2 табл., 28 рис., 2 дод., 26 джерел.

ПОШУК НОВИН, ВЕБЗАСТОСУНОК, ВАРІАТИВНІ ДЖЕРЕЛА НОВИН, JAVASCRIPT, REACT.JS, VISUAL STUDIO CODE.

Об'єкт дослідження – програмні засоби для пошуку та перегляду новин.

Предмет дослідження – вебзастосунки та веб-сайти для пошуку та перегляду новин.

Мета роботи – створення програмного застосунку для пошуку та перегляду новин з можливістю широкого налаштування фільтрів та збереження у локальне сховище.

Матеріали, методи та технічні засоби: функціональне програмування, мови програмування JavaScript, мова розмітки гіпертексту HTML, мова стилю сторінок CSS, бібліотеки React.js, Redux, Axios, ноутбук з процесором Intel Core i3 під управлінням операційної системи Ubuntu.

Результати. Створено максимально оптимізований програмний вебзастосунок на основі бібліотеки React.js та серверного REST API, за допомогою якого користувач може шукати цікаві йому новини та зберігати їх для перегляду в майбутньому.

Висновки. Розроблено вебзастосунок для пошуку та перегляду останніх новин з можливістю збереження у локальне сховище браузера та перегляд на оригінальному сайті, збереження налаштувань, перегляд погоди за геолокацією та курсів валют.

Галузь використання – повсякденне життя людини віком від 14 років.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 89 pages, 28 figures, 2 tables, 2 appendixes, 26 sources.

NEWS SEARCH, WEB APPLICATION, VARIABLE NEWS SOURCES, JAVASCRIPT, REACT.JS, VISUAL STUDIO CODE.

Object of research is a software for searching and viewing news.

Subject of the study are web applications and websites for searching and viewing news.

The purpose of the work is to create a software application for searching and viewing news with the ability to widely customize filters and save to local storage.

Materials, methods, and tools: functional programming, JavaScript programming languages, HTML hypertext markup language, CSS page style language, React.js, Redux, Axios libraries, laptop with Intel Core i3 processor running Ubuntu operating system.

Results. An optimized software application based on the React.js library and the REST server API has been created, with the help of which the user can search for news of interest to him and save it for future viewing.

Conclusions. Developed a web application for finding and viewing the latest news with the ability to save to the local browser storage and view on the original site, save settings, view weather for geolocation and exchange rates.

The field of use is the daily life of a person over 14 years of age.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК.....	8
ВСТУП.....	9
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ І ОБҐРУНТУВАННЯ СТРУКТУРИ ПРОЕКТОВАНОЇ СИСТЕМИ.....	10
1.1 Аналіз предметної області.....	10
1.2 Огляд існуючих методів вирішення завдання.....	11
1.2.1 Вебсайт “Українська правда”	11
1.2.2 Вебсайт “Gazeta.ua”	12
1.2.3 Вебсайт “Страна.ua”	13
1.2.4 Вебсайт “24 канал”	14
1.3 Порівняння функціоналу існуючих рішень	15
1.4 Технічне завдання.....	15
1.5 Висновок до розділу 1.....	16
2 РОЗРОБКА АРХІТЕКТУРИ ПРОГРАМИ	17
2.1 Вибір інструментарію розробки програмного забезпечення.....	17
2.1.1 Вибір мови програмування.....	17
2.1.2 Вибір фреймворку	19
2.1.3 Вибір середовища розробки	26
2.2 Висновки до розділу 2.....	27
3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ	28
3.1 Основні розроблені алгоритми	28
3.1.1 Секція останніх новин.....	28
3.1.2 Секція додаткової інформації.....	29
3.1.3 Модуль пошуку новин за категорією, країною, мовою та ключовою строкою.....	30
3.1.4 Сторінка збережених новин	31
3.1.5 Сторінка нотаток	32
3.1.6 Сторінка курсів валют.....	32

4 ІНСТРУКЦІЇ ПО ВИКОРИСТАННЮ ВЕБЗАСТОСУНКУ	34
4.1 Визначення вимог до технічних засобів	34
4.2 Інструкція для програміста	34
4.3 Структура проекту	35
ВИСНОВКИ	37
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ	38
ДОДАТОК А 40Текст програми	40
ДОДАТОК Б Слайди презентації.....	83

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

API	– Application Programming Interface;
CMS	– Content management system;
REST	– Representational State Transfer;
SPA	– Single Page Application
IDE	– Integrated Development Environment,
JS	– JavaScript
ЗМІ	– Засоби масової інформації

ВСТУП

Масмедіа є одним з найважливіших інститутів громадянського суспільства, що реалізує інформаційні потреби суспільства та забезпечує політичну комунікацію. За статистичними даними, пересічна людина проводить 7 з половиною годин щодня (437 хв.), споживаючи масмедіа. Зазвичай, саме ЗМІ створюють враження та формують думку людини про ту чи іншу подію, діяча чи орган влади [1].

На жаль, не завжди інформація, надана з джерела ЗМІ, є достовірною та правильно сформульованою. Іноді це трапляється через помилки авторів, не достатку зібраної інформації чи поспішний темп праці, проте невірна інформація може стати результатом спроби маніпулювання масами та нав'язування чужих ідей. Сьогодні джерел ЗМІ велика кількість і кожне формулює інформацію зі своєї точки зору, тому іноді дуже важко відрізнити правду від помилки чи пропаганди [4].

З появою Інтернету інформація почала широко розповсюджуватись через вебсайти, що дозволяло людям мати доступ до архіву новин та аналізувати різні джерела. Проте через надмірну кількість інформації стало важко з'ясувати де істина, а де брехня, тому інколи такий прогрес в цифровому світі приводить до інформаційної війни [3]. Саму тому дуже важливо вміти аналізувати і перевіряти те, що пишуть ЗМІ і ніколи не користуватись одним джерелом інформації.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ І ОБҐРУНТУВАННЯ СТРУКТУРИ ПРОЕКТОВАНОЇ СИСТЕМИ

1.1 Аналіз предметної області

В час розвитку цифрових технологій список видів масмедіа тільки поширюється, однак переважна кількість джерел припадає на телебачення та Інтернет:

- масова преса (газета, журнали, книги, бюлетені, листівки);
- масові статичні зображення (рекламні щити й листівки);
- рухомі зображення (телебачення, кінематограф, вебсайти, комп'ютерні програми з візуальним інтерфейсом);
- аудіозаписи чи трансляції (радіо, подкасти);
- тиражовані знімні носії інформації (касет, компакт-диски) [1].

Інтернет-ЗМІ (інтернет-видання, інтернет-газета) — інформаційний сайт, який регулярно оновлюється і виконує функцію засобу масової інформації (ЗМІ), користується певною популярністю і авторитетом (має свою постійну аудиторію) [2].

Зазвичай такі інформаційні сайти розроблені на готових чи власних CMS, що дозволяє повністю керувати контентом вебсайту [5].

Головний недолік Інтернет-ЗМІ полягає в неможливості зручно перевірити достовірність наданої інфомарції та отримати посилання на аналогічні статті в інших інформаційних джерелах, адже така функція може призвести до виходу користувача з сайту.

Існують різні програмні продукти для масового забезпечення інформацією суспільства, однак не кожен користувач бажає бачити те, що йому нав'язує застосунок та, зазвичай, спосіб фільтрації новин не є зручним, а то й зовсім відсутній. Деякі рішення націлені виключно на пропаганду та дезінформацію, інші не повністю вирішують проблему користувача.

1.2 Огляд існуючих методів вирішення завдання

1.2.1 Вебсайт “Українська правда”

Українська правда (рис 1.1) — українське суспільно-політичне Інтернет-ЗМІ, засноване у квітні 2000 року [6].

Переваги програмного застосунку полягають в важливих речах, а саме:

- зручний та сучасний дизайн;
- акцентована увага на заголовки статей;
- власна система керування контентом;
- легкочитаємий текст.

Але цей вебсайт має ряд своїх недоліків:

- відсутній функціонал пошуку на головній сторінці;
- велика кількість зайвої реклами;
- неможливість фільтрації контенту;
- один основний напрямок подачі новин.



Рисунок 1.1 –Вебсайт “Українська правда”

1.2.2 Вебсайт “Gazeta.ua”

Gazeta.ua (рис 1.2) — вебсайт для перегляду новин, блогів різних стилів та опитування користувачів щодо різних тем [7].

Переваги вебзастосунку полягають в:

- легкий та зрозумілий дизайн;
- велика кількість фото та відео матеріалів;
- зафіксований функціонал для пошуку новини;
- додатковий функціонал погоди та гороскопу;
- опитування користувачів.

Але програмний продукт також має свої недоліки:

- статті з “кусаючими” заголовками, що створені задля утримання уваги користувача;
- нераціонально використане місце в заголовку сайта;
- немає підтримки браузера internet explorer;
- переважний один тип новин;
- не адаптивний до всіх типів екранів;
- занадто багато інформаційних блоків в тісних блоках.

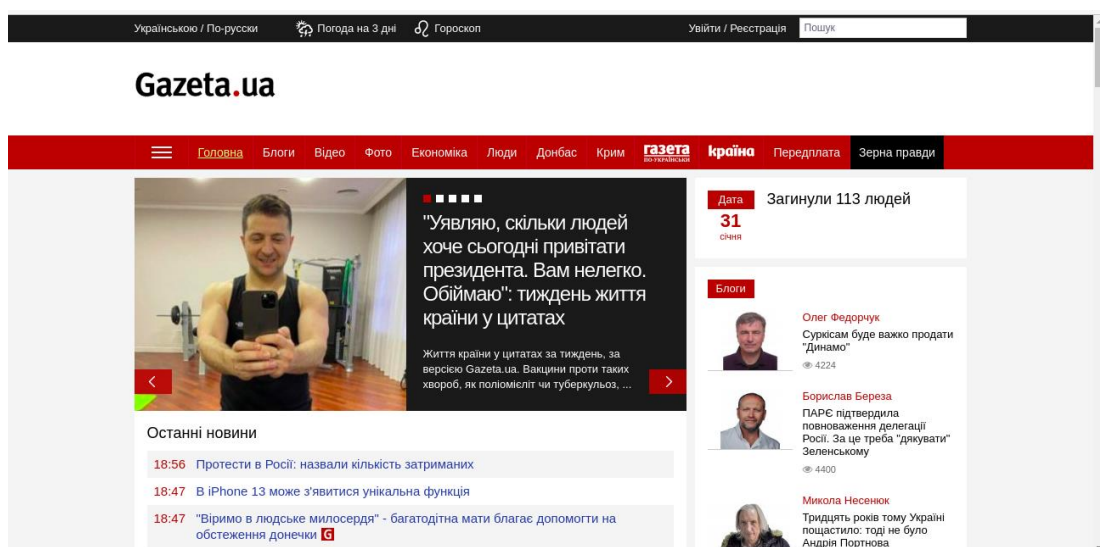


Рисунок 1.2 –Вебсайт “Gazeta.ua”

1.2.3 Вебсайт “Страна.ua”

Страна.ua (рис 1.3) — проросійське інтернет-видання в Україні, відкрите 16 лютого 2016 року Ігорем Гужвою [8].

Недоліків цього сайту більше ніж переваг:

- незручний дизайн;
- нераціонально розташовані блоки;
- нечитаємий текст;
- занадто мало зображень у відношенні с текстом;
- незручний пошук;
- відсутність фільтрації;
- відсутність посилань на джерело інформації.

Переваг застосунку я не знайшов, адже він переповнений контентом, який неможливо відфільтрувати, а пошук виконаний стандартним маленьким блоком, який ледь помітний.

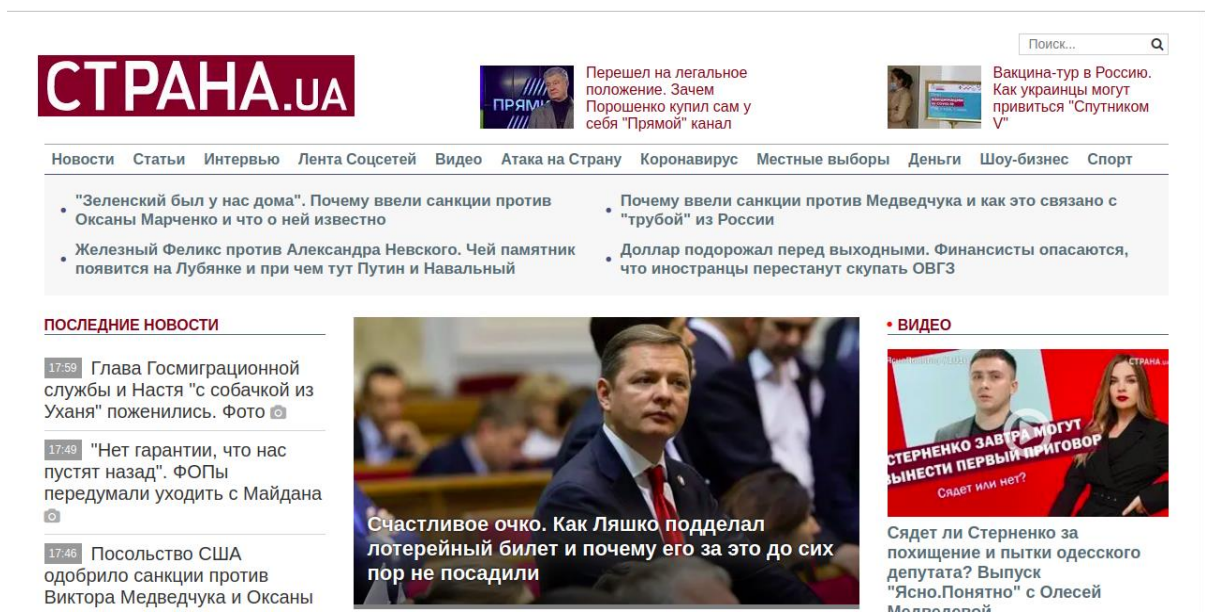


Рисунок 1.3 –Вебсайт “Gazeta.ua”

1.2.4 Вебсайт “24 канал”

24tv.ua (рис 1.4) — сайт новинного 24 каналу, його було створено в жовтні 2008-го року. Сайт регулярно входить до топ-5 найпопулярніший інформаційних порталів України, а також до топ-20 найчастіше відвідуваних в Україні [9].

Вебсайт має безліч переваг:

- сучасний та комфортний дизайн;
- зручна фільтрація новин;
- посилання на джерело майже в кожній статті;
- витримане відношення зображень до тексту;
- добре читаємий текст, збережений контраст;
- реалізоване меню з вибором категорії статей.

Проте можна знайти і недоліки:

- реклама азартних ігор;
- немає ніякої взаємодії користувача зі статтями.

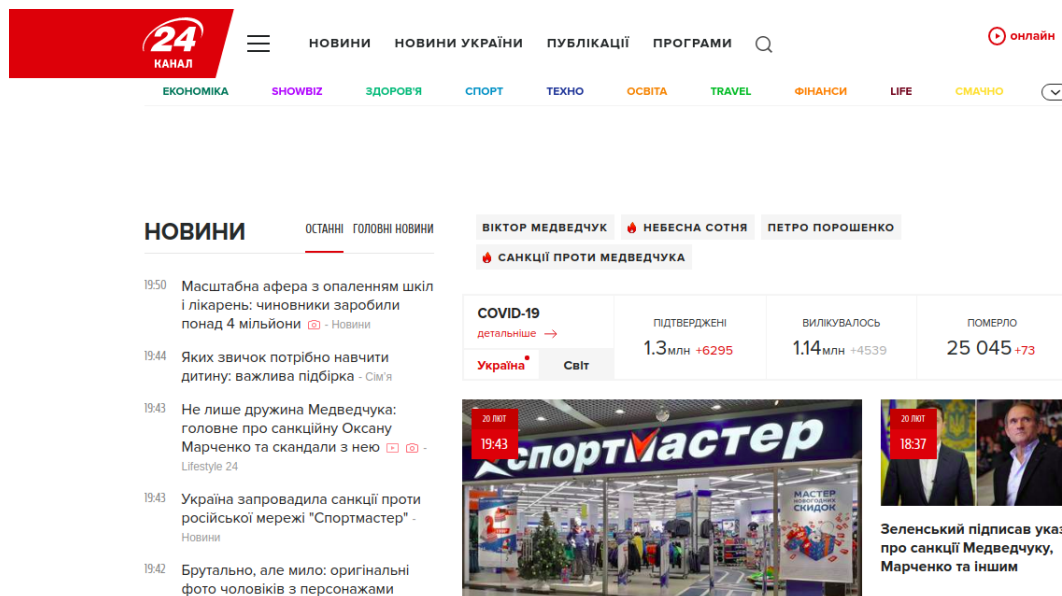


Рисунок 1.4 –Вебсайт “24 Канал ”

1.3 Порівняння функціоналу існуючих рішень

Проаналізувавши існуючі рішення реалізації застосунків та вебсайтів для пошуку новин, було виділено основні переваги та недоліки усіх функціональних особливостей існуючого програмного забезпечення (табл. 1.1).

Таблиця 1.1 – Порівняльна характеристика найпоширеніших програмних продуктів

Функціональні можливості	Назва програмних продуктів				
	Українська Правда	Gazeta.ua	Страна.ua	24 канал	Новий продукт
Зручний дизайн	+	+	-	+	+
Легкочитаємий текст	+	+	-	+	+
Пошук за ключовою фразою	+	+	+	+	+
Відсутність реклами	-	-	-	-	+
Зручна фільтрація статей	+-	-	-	+	+
Порівняння статей	-	-	-	-	+
Адаптивність до різних екранів пристроїв	+	-	+	+	+
Додаткові дані (погода, курси валют)	-	+	-	-	+
Посилання на джерело	+	-	-	+	+
Додатковий функціонал	-	-	-	+	+

1.4 Технічне завдання

Вебзастосунок буде реалізований як SPA і матиме свою архітектуру, усі дані будуть запитуватись за готовим REST API.

Програмний продукт призначений для перегляду та пошуку новин за певними критеріями, а саме:

- категорія статті;
- ключове слово або фраза;
- мова.

Також в програмі буде реалізовано інші дані та функції:

- перегляд курсів валют на сьогодні та за останні 7 днів у вигляді графіку;
- перегляд погоди на сьогодні по геолокації користувача;
- отримання фото дня;
- перегляд останніх найновіших новин;
- налаштування геолокації, збереження вибраних категорій та джерел змі;
- можливість додати новину до обраних;
- можливість зберегти і переглянути новину пізніше;
- можливість фіксувати та перетягувати блоки зі статтями;
- можливість писати та зберігати нотатки;
- можливість додати статтю до порівняння з іншими.

1.5 Висновок до розділу 1

Проаналізувавши методи та програмні засоби реалізації вебзастосунків Інтернет-ЗМІ можна сказати, що наразі не існує гнучкого та зручного програмного засобу для пошуку та перегляду новин з можливістю аналізу та пошуку в іншому джерелі. Саме цьому новий програмний продукт має поєднати в собі функціонал, який є у існуючих альтернативах та стати повноцінним інструментом для отримання та аналізу інформації з різних джерел ЗМІ, усунувши ті недоліки, які є в існуючих рішеннях основної задачі.

2 РОЗРОБКА АРХІТЕКТУРИ ПРОГРАМИ

2.1 Вибір інструментарію розробки програмного забезпечення

2.1.1 Вибір мови програмування

Для розробки вебзастосунку для пошуку та перегляду новин обрано стандартизовані мови HTML5[10] та CSS3[11] - для створення каскаду та придання стилістики вебсторінці та JavaScript для керування даними на сторінці. Вирішено розробляти програмний продукт як SPA [12].

HTML – це мова тегів, засобами якої здійснюється розмічання вебсторінок для мережі Інтернет. Елементи HTML є будівельними блоками сторінок HTML. За допомогою конструкцій HTML, зображення та інші об'єкти, такі як інтерактивні форми, можуть бути вбудовані у візуалізовану сторінку. HTML надає засоби для створення структурованих документів, позначаючи структурну семантику тексту, наприклад заголовки, абзаци, списки, посилання, цитати та інші елементи.

CSS - це спеціальна мова стилю сторінок, що використовується для опису їхнього зовнішнього вигляду. CSS використовується авторами та відвідувачами веб-сторінок, щоб визначити кольори, шрифти, верстку та інші аспекти вигляду сторінки. Одна з головних переваг — можливість розділити зміст сторінки (або контент, наповнення, зазвичай HTML, XML або подібна мова розмітки) від вигляду документу (що описується в CSS).

Для створення вебзастосунку використовується мова програмування JavaScript, оскільки це єдина мова, яку підтримують браузер.

Single Page Application (SPA чи односторінковий застосунок) реалізує зручні для користувача сервіси, що наповнені інтерактивом. Найпростішим прикладом є Gmail. SPA — веб-застосунок, розташований на одній фізичній HTML сторінці. Така сторінка одноразово завантажує усі необхідні ресурси (JavaScript, CSS, images тощо.) і більше не перезавантажується. Переходи за посиланнями не призводять до реального перезавантаження сторінки, а її вміст змінюється «на льоту», тобто динамічно. За необхідністю виконується

запит на сервер для отримання даних, і після їх отримання формується контент «нової» сторінки. Кожна окрема віртуальна сторінка прописується у маршрутизаторі (router)[20].

JavaScript (JS) — динамічна, об'єктно-орієнтована прототипна мова програмування. Реалізація стандарту ECMAScript [14]. Найчастіше використовується для створення сценаріїв вебсторінок, що надає можливість на боці клієнта (пристрої кінцевого користувача) взаємодіяти з користувачем, керувати браузером, асинхронно обмінюватися даними з сервером, змінювати структуру та зовнішній вигляд вебсторінки [13].

JS класифікують як прототипну (підмножина об'єктно-орієнтованої), скриптову мову програмування з динамічною типізацією. Окрім прототипної, JS також частково підтримує інші парадигми програмування (імперативну та частково функціональну) і деякі відповідні архітектурні властивості, зокрема: динамічна та слабка типізація, автоматичне керування пам'яттю, прототипне наслідування, функції як об'єкти першого класу [13].

Мова JS використовується для:

- написання сценаріїв вебсторінок для надання їм інтерактивності створення односторінкових та прогресивних вебзастосунків (React, AngularJS, Vue.js , [16,17,18]);
- програмування на боці сервера (Node.js(Express.js));
- стаціонарних застосунків (Electron, NW.js);
- мобільних застосунків (React Native, Cordova);
- сценаріїв в прикладних програмах (наприклад, в програмах зі складу Adobe Creative Suite чи Apache JMeter);
- всередині PDF-документів.

JS дуже гнучка та потужна мова, проте для розробки застосунку з динамічно змінними блоками HTML та впровадження бізнес логіки і локального збереження даних потрібно скористатись JS-фреймворками та бібліотеками для State Management [15, 16].

2.1.2 Вибір фреймворку

Для створення SPA виділяють три основні фреймворки та бібліотеки:

- React;
- Vue;
- Angular.

Чим фреймворк відрізняється від бібліотеки? Бібліотека зазвичай фокусується на вузькій області. Вона надає набір допоміжних функцій, методів які ви можете викликати в своєму проєкті для досягнення певної функціональності. В основному це колекція визначень класів, написана в для сприяння повторному використанню коду. Немає необхідності щоразу починати з нуля, ви можете легко використовувати написане іншими, щоб заощадити час і енергію.

Оскільки бібліотека має вузьку сферу застосування, API-інтерфейси також менше, і, в кінцевому рахунку, проєкту потрібно менше залежностей для досягнення мети.

Фреймворки - це фундамент, на якому розробники створюють програми для конкретних платформ. Вони призначені для зменшення числа проблем, з якими програміст стикається під час розробки.

Фреймворки можуть мати певні або невизначені функції і об'єкти, які програміст може використовувати або перезаписати для створення програми. Основне їхнє завдання - надання стандартизованого коду, який може бути застосований до різних проєктів програм.

І бібліотека, і фреймворк грають життєво важливу роль в розробці програмного забезпечення. Бібліотека виконує певну або чітко визначену операцію, в той час як фреймворк надає каркас, де програмісти визначають зміст додатки в операції. Але коли справа доходить до ключової різниці - інверсія контролю. Коли ми викликаємо функцію або метод з бібліотеки, ми контролюємо ситуацію. Але в іншому випадку фреймворк викликає наш код і, таким чином, управління інвертується. У більшості випадків структура

тільки забезпечує концепцію. Робота програми полягає в подальшому визначенні функціональності для кінцевих користувачів.

Тисячі бібліотек доступні практично для кожної мови і допомагають програмістам повторно використовувати код, написаний іншими. З іншого боку, фреймворки допомагають їм зосередитися на розвитку, а не на проблемах.

Перш за все, варто зазначити, що Angular застосунки пишуться на TypeScript, а не на чистому JavaScript. Версія синтаксису для JavaScript не отримала широкого розповсюдження, тому на даний момент у документації усі синтаксичні конструкції описані з використанням синтаксису TypeScript.

Архітектура Angular складається з:

- Module;
- Component;
- Template;
- Service;
- Router;
- Pipe;
- Directives.

Модулі (Module) — структурні одиниці застосунку, які інкапсують певну логіку. В Angular це структури, які зберігають певні компоненти, директиви та сервіси, об'єднані певною логікою. Прикладом може слугувати профіль користувача, модуль для написання листа, огляд списку листів тощо.

Компоненти (Component) — typescript клас, який зберігає дані та логіку відображення цих даних у шаблоні (представленні). Шаблон тісно пов'язаний з компонентом. Дані з компонента можна з легкістю відобразити у шаблоні, використовуючи спеціальний синтаксис. Компонент також може «знімати» дані з шаблону та отримувати їх безпосередньо у скрипті. Шаблон (Template) — фрагмент html-коду з додаванням спеціального синтаксису. Він дозволяє впроваджувати в шаблон дані з компонента без використання

innerHTML та подібних методів. Шаблон прописується у компоненті та є частиною його конфігурації.

Сервіс (Service) в Angular являє собою typescript класи, які виконують задачі, пов'язані з отриманням, зберіганням та обробкою даних. Наприклад, логування, перетворення даних для подальшої передачі у компонент, звернення до backend та ін. На відміну від компонентів та директив сервіси не працюють з представленнями (шаблонами) напряму. Задачі сервісів: Надання даних застосунку. Сервіс сам може зберігати дані у пам'яті або, з метою отримання даних, звертатися до якогось джерела даних, наприклад, до сервера; Сервіс може організувати канал взаємодії між окремими компонентами застосунку; Сервіс може інкапсулювати бізнес-логіку, різноманітні обчислювальні задачі, задачі з логування, які краще виносити поза компоненти. Таким чином, код компонентів буде зосереджений, безпосередньо, на роботі з представленням. До того ж, можемо розв'язати проблему повторення коду, якщо нам знадобиться виконати одну й ту саму задачу у різних компонентах і класах.

Роутер (Router)— маршрутизатор, який призначений для переходу між екранами з метою відображення різного контенту. Іншими словами, коли в адресному рядку браузера у вас змінюється фрагмент URL, маршрутизатор відстежує ці зміни та завантажує ту або іншу частину застосунку. Завершимо теоретичну частину та перейдемо до написання коду.

Директиви та Пайпи — більш специфічні конструкції, які простіше продемонструвати у коді, ніж описати словами [20].

Angular потужний та не зовсім гнучкий фреймворк, він підходить для написання більш важких і потенційно ресурсозатратних програм, проте затримає розробку на початкових етапах і в майбутньому. Для розробки вебзастосунку для перегляду новин цей фреймворк не підходить.

Vue.js — JavaScript-фреймворк що використовує шаблон MVVM для створення інтерфейсів користувача на основі моделей даних, через реактивне зв'язування даних [18].

Головні особливості фреймворку Vue:

Шаблони. Vue використовує синтаксис шаблонів на основі HTML, що дозволяє декларативно зв'язувати рендеринг DOM [22] з основними екземплярами даних в Vue. Всі Vue шаблони валідні HTML, і можуть бути розпарсені браузером та HTML парсерами. Всередині Vue компілює шаблони в рендерингові функції віртуального DOM. В поєднанні з реактивною системою, Vue здатний розумно обчислити кількість компонентів для ре-рендингу та застосувати мінімальну кількість маніпуляцій з DOM, коли стан застосунку зміниться. В Vue ви можете використовувати синтаксис шаблонів або напрямку писати рендерингові функції використовуючи JSX [21]. Для того, щоб це зробити просто замініть шаблон на рендерингову функцію. Рендерингова функція відкриває можливості для потужних патернів базованих на компонентах - для прикладу, нова транзитна система тепер повністю базована на компонентах, що використовує рендерингові функції всередині.

Реактивність Одна із найвиразніших особливостей Vue — це ненав'язлива реактивна система. Моделі це просто плоскі JavaScript об'єкти. Це робить керування станами дуже простим та інтуїтивним. Vue надає оптимізований ре-рендеринг з коробки без потреби робити що-небудь додатково. Кожен компонент слідує за своїми реактивними залежностями під час рендерингу, тому система знає точно коли має відбуватись ре-рендеринг і які компоненти потрібно ре-рендерити.

Переходи Vue надає різноманітні шляхи для застосування ефектів переходу, коли елемент додають, оновлюють або видаляють з DOM. Наприклад: автоматичне застосування класів для CSS переходів та анімацій інтегрування сторонніх бібліотек для CSS анімацій, таких як Animate.css використовувати JavaScript для прямих маніпуляцій з DOM під час переходів

інтегрування сторонніх JavaScript бібліотек анімацій, таких як Velocity.js. Коли елемент, який загорнутий перехідний компонент видаляють чи вставляють, ось що відбувається: Vue автоматично перевірить чи має застосовують до цього елементу CSS анімації та переходи. Якщо так, CSS класи для переходів будуть додані/видалені у відповідний час. Якщо перехідний компонент має JavaScript зачіпки, ці зачіпки будуть викликані у відповідний час. Якщо ніяких CSS переходів/анімацій не було знайдено та ніяких JavaScript не було надано, DOM операції для додавання і/та видалення відбудеться одразу ж в наступному фреймі.

Роутинг Vue сам по собі не включає роутингу, та є vue-router пакет, який вирішує це питання. Він підтримує зв'язування вкладених шляхів з вкладеними компонентами і пропонує деталізований контроль над переходами. Vue дозволяє створення додатків за допомогою компонентів. Якщо додати vue-router до цього, все що потрібно зробити це зв'язати ваші компоненти з роутами і дозволять vue-router вирішувати де їх рендерити [18].

Vue відносно новий фреймворк, він досить легкий в освоєнні та на ньому можна швидко розробити проект. Проте проблеми починаються коли прийдеться підтримувати програмний продукт. Через те, що Vue досить молодий фреймворк, в нього мале ком'юніті і якщо б розробник наткнувся на проблему в Angular, то з вірогідністю в 95% цю проблему вже вирішили на різних форумах, а у випадку з Vue проблема може бути настільки складною, що призупинить надовго проект. Крім того, Vue диктує правила розробки і вганяє в рамки, з одного боку це дозволяє уникнути помилок в коді, а з іншого це обмежує розробника рішеннями без альтернатив. Саме тому фреймворк Vue не підходить для розробки нашого продукту.

React — відкрита JavaScript бібліотека для створення інтерфейсів користувача, яка покликана вирішувати проблеми часткового оновлення вмісту веб-сторінки, з якими стикаються в розробці односторінкових застосунків. Розробляється Facebook, Instagram і спільнотою індивідуальних розробників.

Головні особливості React:

Одностороння передача даних. Властивості передаються в рендерер компоненту, як властивості html тегу. Компонент не може напряму змінювати властивості, що йому передані, але може їх змінювати через callback функції. Такий механізм називають «властивості донизу, події нагору».

Віртуальний DOM [22]. React підтримує віртуальний DOM, а не покладається виключно на DOM браузера. Це дозволяє бібліотеці визначити, які частини DOM змінилися, порівняно (diff) зі збереженою версією віртуального DOM, і таким чином визначити, як найефективніше оновити DOM браузера. Таким чином програміст працює зі сторінкою, вважаючи що вона оновлюється вся, але бібліотека самостійно вирішує які компоненти сторінки треба оновити.

JSX[21]. Компоненти React зазвичай написані на JSX. Код написаний на JSX компілюється у виклики методів бібліотеки React. Розробники можуть так само писати на чистому JavaScript. JSX нагадує іншу мову, яку створили у компанії Фейсбук для розширення PHP, XHP.

Не лише рендеринг HTML в браузері. React використовують не лише для рендерингу HTML в браузері. Наприклад, Facebook має динамічні графіки які рендеряться в теги <canvas>, Netflix та PayPal використовують ізоморфне завантаження для рендерингу ідентичного HTML на сервері та клієнті.

Методи життєвого циклу. Методи життєвого циклу – це різні методи, які вбудовуються за допомогою ReactJS. Вони дозволяють розробнику обробляти дані в різних точках життєвого циклу програми React. Наприклад:

- `shouldComponentUpdate` – це метод життєвого циклу, який каже Javascript оновити компонент, використовуючи логічні змінні;
- `componentWillMount` – це метод життєвого циклу, який каже Javascript налаштувати певні дані перед монтуванням компонентів (вставлення у віртуальний DOM);

– `componentDidMount` – це метод життєвого циклу, подібний до компонента `WillMount`, за винятком того, що він працює після методу `render`, і може використовуватися для додавання JSON-даних, а також для визначення властивостей та станів;

– `render` є найважливішим методом життєвого циклу, необхідним у будь-якому компоненті, метод `render` – це те, що з'єднується з JSX і відображати власний JSX.

Вкладені елементи. Кілька елементів на одному рівні повинні бути загорнутими в один елемент контейнера, наприклад елемент `<div>`, або повернутий як масив.

Атрибути. JSX надає ряд атрибутів елементів, призначених для відображення тих, що надаються у форматі HTML. Користувацькі атрибути також можуть бути передані компоненту. Всі атрибути будуть отримані компонентом як реквізит [17].

Проаналізувавши основні переваги та недоліки цих фреймворків (табл 2.1) вирішено, що React є максимально гнучкою і швидкою завдяки `VirtualDom`, вона не диктує правила і дозволяє використовувати інші бібліотеки разом з нею. Вона має велике ком'юніті та підтримується командою Facebook. Ця бібліотека максимально підходить для вирішення задачі.

Таблиця 2.1 – Порівняльна характеристика найпоширеніших фреймворків та бібліотек

Фреймворк / бібліотека	Переваги	Недоліки
1	2	3
Angular	Велика спільнота Декларативний стиль коду Наявність готових рішень Компонентний підхід Повноцінний фреймворк	Великий обсяг результуючого коду. Стисла, незручна документація. Великий поріг входу.

Продовження таблиці 2.1

1	2	3
React	Віртуальний DOM Велика спільнота швидкість рендерингу Велика кількість допоміжних інструментів та готових рішень	Є бібліотекою для рендеру. Розробники, іноді, мають занадто великий вибір, не має жорстко заданих правил
Vue	Зрозуміла документація Крихітний розмір. Vue.js може важити близько 20 КБ. Чудова інтеграція. Vue.js можна використовувати як для створення односторінкових додатків, так і для складніших веб-інтерфейсів додатків.	Ризик надмірної гнучкості. Іноді у Vue.js можуть виникати проблеми при інтеграції в величезні проекти, а досвіду про можливі рішення до сих пір немає. Не хватка ресурсів

2.1.3 Вибір середовища розробки

В якості середовища розробки було вибрано Microsoft Visual Studio Code.

Visual Studio Code — засіб для створення, редагування та зневадження сучасних веб-застосунків і програм для хмарних систем. Visual Studio Code розповсюджується безкоштовно і доступний у версіях для платформ Windows, Linux і OS X. Компанія Microsoft представила Visual Studio Code у квітні 2015 на конференції Build 2015. Це середовище розробки стало першим крос-платформовим продуктом у лінійці Visual Studio. За основу для Visual Studio Code використовуються напрацювання вільного проекту Atom, що розвивається компанією GitHub. Зокрема, Visual Studio Code є надбудовою над Atom Shell, що використовують браузерний рушій Chromium і Node.js. Примітно, що про використання напрацювань вільного проекту Atom на сайті Visual Studio Code і в прес-релізі і в офіційному блозі не

згадується[9]. Редактор містить вбудований зневаджувач, інструменти для роботи з Git і засоби рефакторингу, навігації по коду, автодоповнення типових конструкцій і контекстної підказки. Продукт підтримує розробку для платформ ASP.NET і Node.js, і позиціонується як легковагове рішення, що дозволяє обійтися без повного інтегрованого середовища розробки. Серед підтримуваних мов і технологій: JavaScript, C++, C#, TypeScript, jade, PHP, Python, XML, Batch, F#, DockerFile, Coffee Script, Java, HandleBars, R, Objective-C, PowerShell, Luna, Visual Basic, Markdown, JSON, HTML, CSS, LESS і SASS, Haxe [23].

2.2 Висновки до розділу 2

Для створення вебзастосунку було обрано найефективніший та найсучасніший набір інструментів. Мовою програмування було обрано JavaScript , бібліотека React, середовищем розробки Microsoft Visual Studio Code.

3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

3.1 Основні розроблені алгоритми

Основні алгоритми розробленої системи:

- виведення списку останніх новин на поточну дату;
- курси основних валют на поточну дату відносно гривні;
- історія курсів валют у вигляді графіку відносно гривні;
- погода за геолокацією користувача;
- відображення фото дня;
- пошук новин за категорією, країною, мовою та ключовою строкою;
- додавання або видалення новин зі збережених новин;
- додавання або видалення нотаток.

3.1.1 Секція останніх новин

Виведення списку останніх новин реалізовано за допомогою слайдера Swiper.js [23], данні запитуються через API – NewsAPI [24]. News API - це простий API для пошуку та отримання активних статей з усього Інтернету. Список новин запитується при ініціалізації застосунку та може бути оновлений користувачем шляхом натискання кнопки оновлення. Запит налаштований на головні новини світу українською мовою, адже основна цільова аудиторія української національності. (рис 3.1).



Рисунок 3.1 – Секція головних новин

3.1.2 Секція додаткової інформації

Секцію додаткової інформації (рис 3.2) складають блоки:

- курси валют;
- погода за геолокацією;
- фото дня.



Рисунок 3.2 – Секція додаткової інформації

Курси валют реалізовано компонентом з використанням запиту до API ПриватБанку. API повертає дані основних актуальних валют:

- долар;
- євро;
- російський рубль;
- біткоїн.

Блок погоди розроблено за допомогою OpenWeatherMap API [25].

Джаваскрипт дозволяє звертатись до поточної геолокації користувача та отримати координати широти та довготи. За цими даними програма робить запит до API та відображує отриману інформацію.

Блок фото дня було реалізовано також звертанням до спеціального API – Unsplash. Це сервіс для отримання фото за різними параметрами, але я адаптував його до за стосунку як фото дня. Фото запитується при ініціалізації застосунку, в локальному сховищі браузера записується дата запиту, наступне фото буде запитане якщо с цієї дати пройде 24 години.

3.1.3 Модуль пошуку новин за категорією, країною, мовою та ключовою строкою

Головний модуль застосунку розроблений за допомогою NewsAPI. При введенні строки або зміні параметру виконується запит до API та програма відображує отримані результати. (рис 3.3).

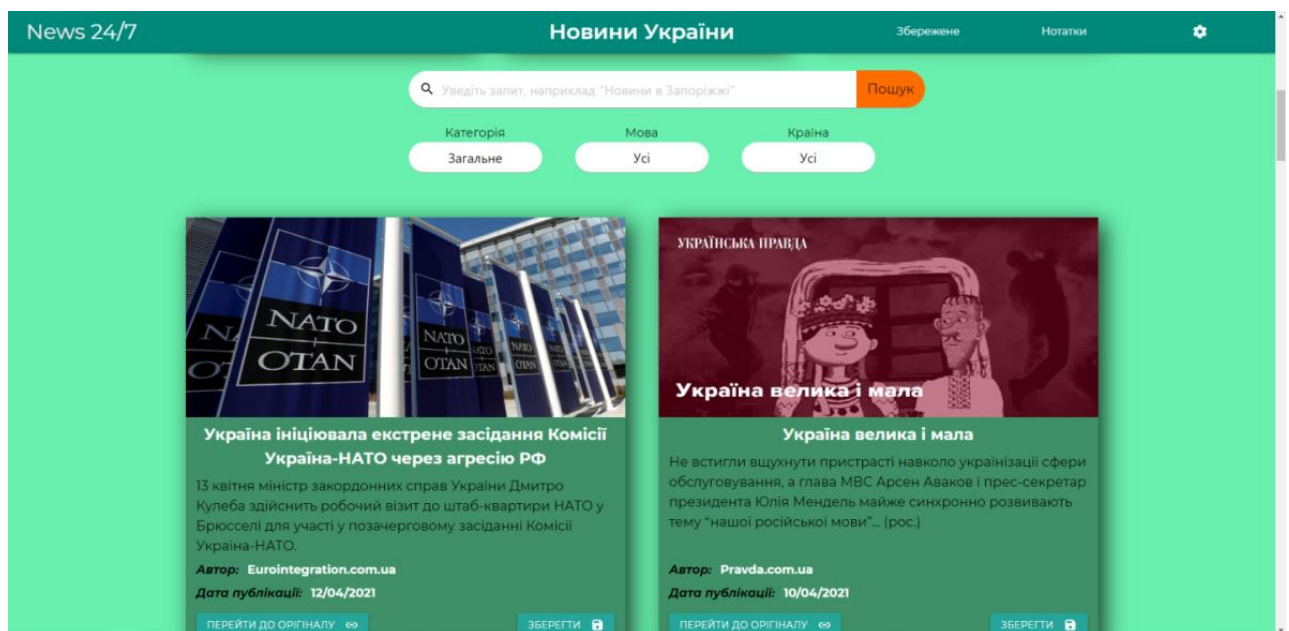


Рисунок 3.3 – Модуль пошуку новин

3.1.4 Сторінка збережених новин

Збережені новини відображаються з даних з локального сховища. Коли користувач бачить в пошуку потрібну йому новину и натискає кнопку «Зберегти» на карточці новини (рис 3.4) дані про новину зберігаються в масиві в локальному сховищі браузерa. Користувач також може видалити новину зі збережених. (рис 3.5). Усі операції з новинами супроводжуються відповідними сповіщеннями. (рис 3.6).

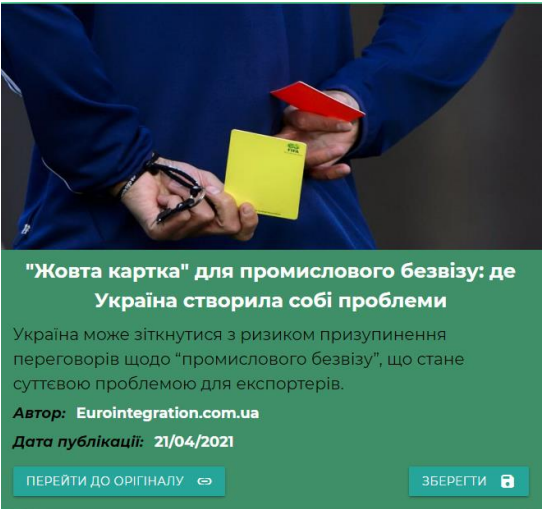


Рисунок 3.4 – Карточка новини

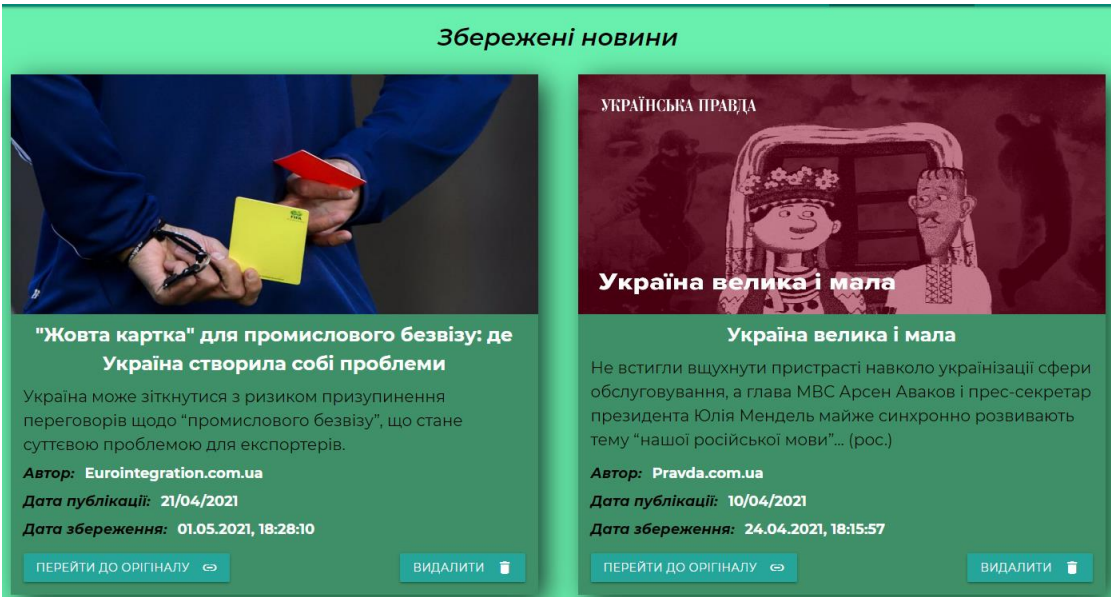


Рисунок 3.5 – Карточка новини в збережених

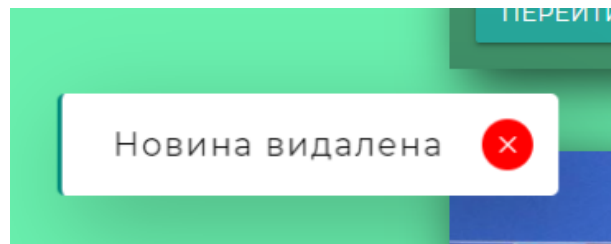


Рисунок 3.6 – Сповіщення про успішно виконану операцію

3.1.5 Сторінка нотаток

Сторінка нотаток (рис 3.7) реалізована за допомогою локального сховища браузера. Користувач має змогу записати необхідну інформацію і звернутись до неї в будь-який час. Також є можливість видалити збережені нотатки.

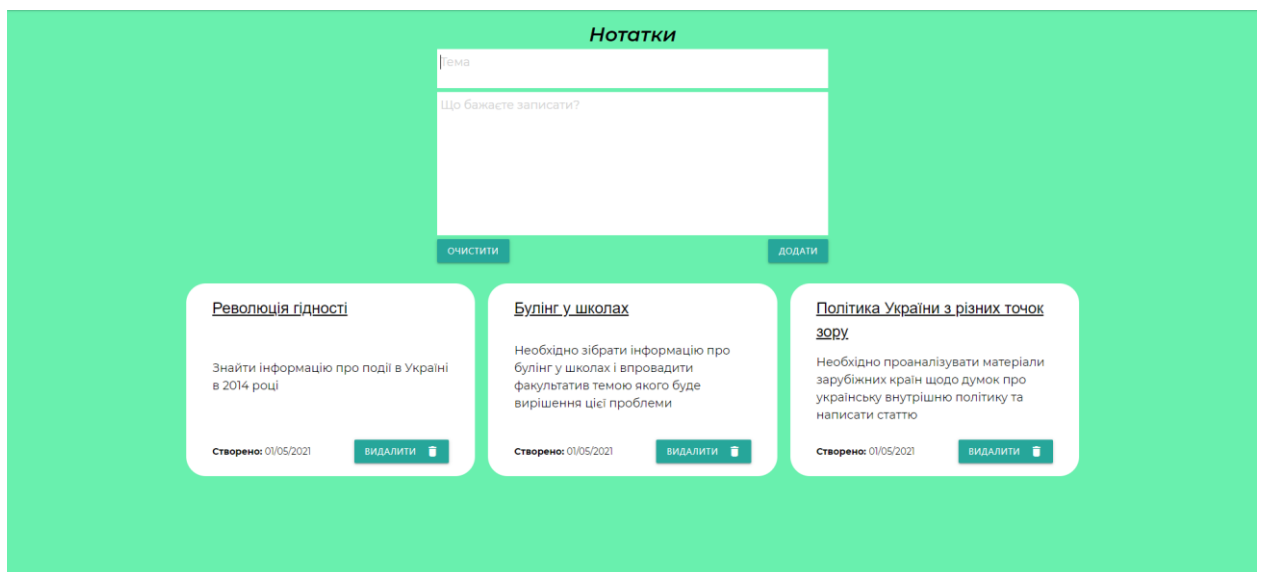


Рисунок 3.7 – Сторінка нотаток

3.1.6 Сторінка курсів валют

Сторінка курсів валют виконана на основі JS бібліотеки Chart.js [26]. Бібліотека реалізує поле канвас з графіком за заданими даними. Самі дані приходять з API Національного Банку України. Відображаються курси у

вигляді графіку та у двох варіантах: основні валюти (рис 3.8) та усі валюти (рис 3.9). Додано можливість розгортати графік для зручного перегляду.

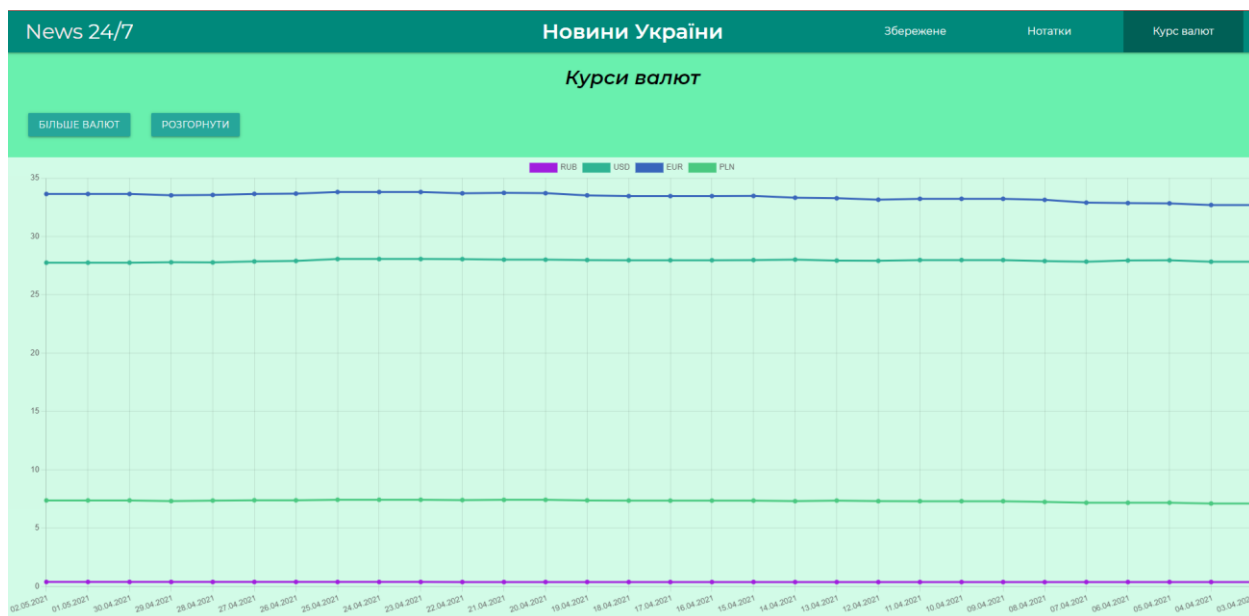


Рисунок 3.8 – Графік основних валют

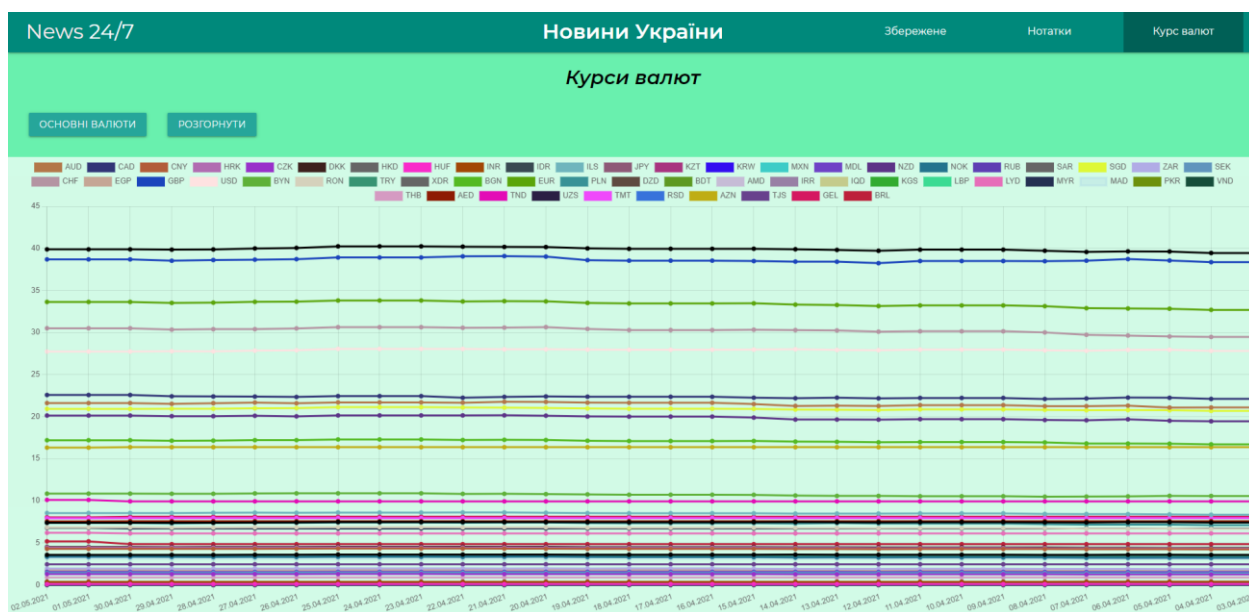


Рисунок 3.9 – Графік усіх валют

4 ІНСТРУКЦІЇ ПО ВИКОРИСТАННЮ ВЕБЗАСТОСУНКУ

4.1 Визначення вимог до технічних засобів

Технічні характеристики комп'ютера для створення та збірки програми:

- AMD Athlon 200GE 3.2GHz 4MB;
- RAM 4 ГБ;
- жорсткий диск 120 ГБ;
- nVidia GeForce GT 630 2 Gb;
- DDR3.

Екран:

- 23.8 (1920x1080) FHD.

Програмне забезпечення:

- Windows 10, 64-х розрядна;
- Node;
- VSCode.

Мова програмування:

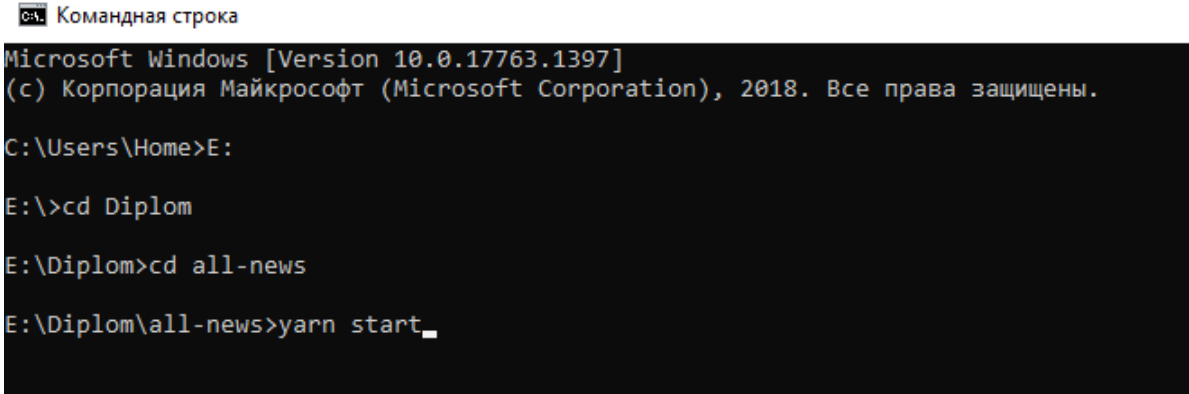
- React;
- Node.

Мінімальні вимоги до технічних засобів – це будь-який комп'ютер з можливістю виходу до мережі Інтернет через браузер.

4.2 Інструкція для програміста

Для встановлення програмного продукту та його запуску на локальному сервері необхідно мати встановлений Node.js та Node Package Manager. Також за бажанням можна встановити yarn (`npm install -g yarn`) для прискорення процесу установки.

Спочатку потрібно провести команду `npm install` або `yarn`, яка ініціалізує проєкт, виконати команду `yarn start` або `npm start` для запуску проєкту на локальному сервері (рис 4.1)



```
Командная строка
Microsoft Windows [Version 10.0.17763.1397]
(c) Корпорация Майкрософт (Microsoft Corporation), 2018. Все права защищены.

C:\Users\Home>E:

E:\>cd Diplom

E:\Diplom>cd all-news

E:\Diplom\all-news>yarn start_
```

Рисунок 4.1 – Команда для запуску програми

Після запуску команди в браузері буде відкритий `http://localhost:3000/` де буде розташований проєкт. Перехід між сторінками здійснюється за допомогою `React Router Dom` по структурі (рис 4.2), що дозволяє без перезавантаження міняти контент у застосунку.

Роутинг можна змінити у файлі `App.js`, додати чи видалити сторінку чи відображує мий компонент.

Редагування модулів та зміну структури можна реалізувати в папці з усіма головними модулями в папці “`all-news/src/modules`”

Редагування стилів компонентів реалізується в самій папці компонента в файлі с постфіксом “`module.scss`”.

4.3 Структура проєкту

Структура проєкту (рис 4.3) розроблена за програмними паттернами `SOLID` [27].

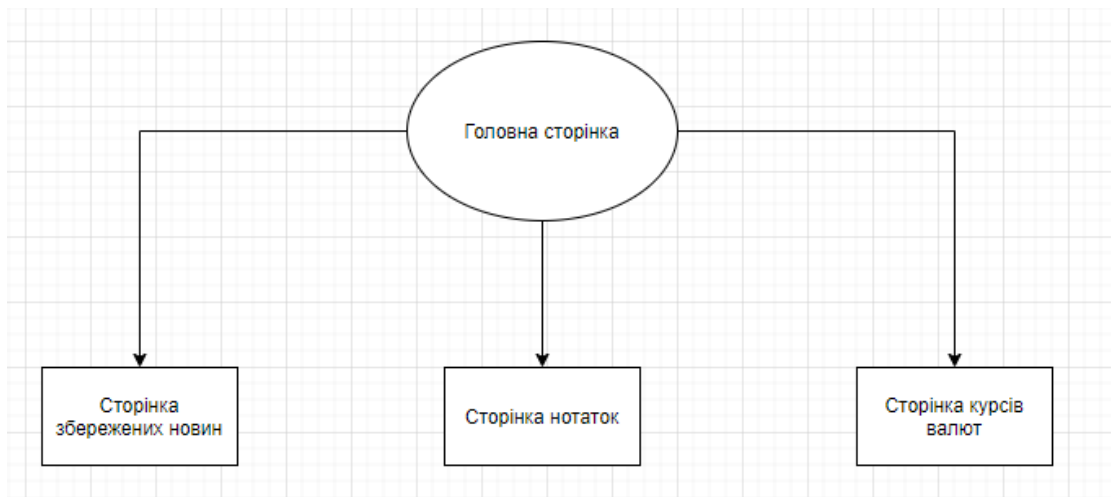


Рисунок 4.2 – Блок-схема структури роутингу вебзастосунку

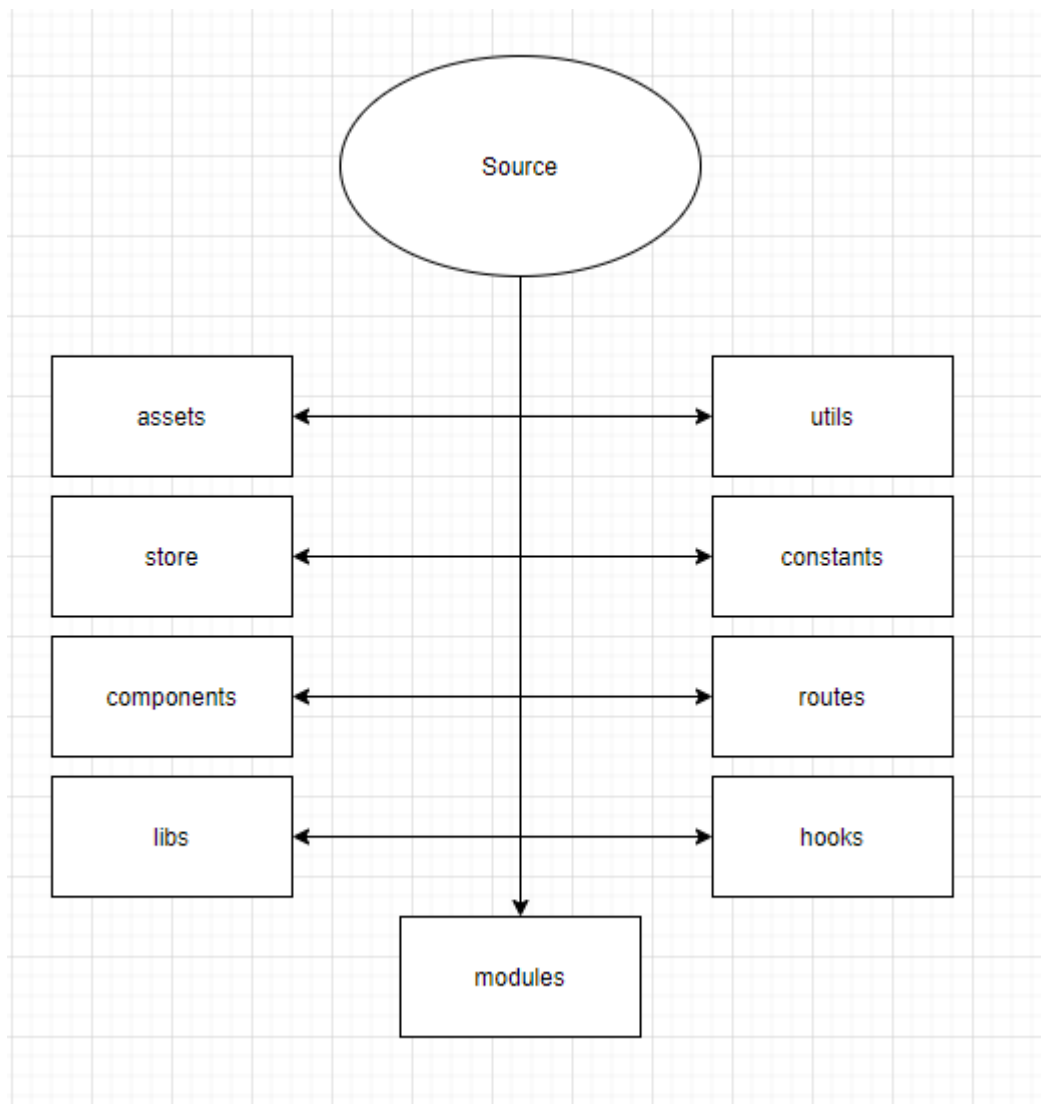


Рисунок 4.3 – Блок-схема структури папок проекту

ВИСНОВКИ

В ході експлуатації критичних помилок не було виявлено. Застосунок дозволяє швидко и зручно шукати необхідну інформацію з різних масмедіа.

Основним недоліком програми є локальне збереження даних, проте реалізація бек-енду цього проекту зовсім інша задача, яка є спонукальним моментом для розширення та покращення продукту. ПП повністю виконує поставлені користувачем задачі та може бути використаний в корисних цілях.

Перевагою можна назвати те, що на застосунок не ведеться ніякої навантаження, адже реалізації серверу та бази даних не виконано, а отже усі важкі процеси з мережею Інтернет виконуються на стороні серверних API, які підключено у програмі.

Крім основних функцій, було реалізовано інші інформативні блоки, що дозволяє в стислому чи розширеному варіанті отримати максимально корисну інформацію.

Продукт можна назвати максимально оптимізованим та «легким».

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Засоби масової інформації [Електронний ресурс]. – Режим доступу: uk.wikipedia.org/wiki/Засоби_масової_інформації.
2. Інтернет-ЗМІ [Електронний ресурс]. – Режим доступу: uk.wikipedia.org/wiki/Інтернет-ЗМІ.
3. Інформаційна війна [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Інформаційна_війна.
4. Методи впливу пропаганди та протидія їм [Електронний ресурс]. – Режим доступу: <https://imi.org.ua/monitorings/metody-vplyvu-propagandy-ta-protydiya-nyu-i30638>.
5. Система керування змістом [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Система_керування_вмістом.
6. Інтернет-ЗМІ “Українська правда” [Електронний ресурс]. – Режим доступу: <https://www.pravda.com.ua/>.
7. Інтернет-ЗМІ “Gazeta.ua” [Електронний ресурс]. – Режим доступу: <https://gazeta.ua/>.
8. Інтернет-ЗМІ “Страна.ua” [Електронний ресурс]. – Режим доступу: <https://uk.wikipedia.org/wiki/Страна.ua>.
9. Інтернет-ЗМІ “24 Канал” [Електронний ресурс]. – Режим доступу: [https://uk.wikipedia.org/wiki/24_Канал_\(сайт\)](https://uk.wikipedia.org/wiki/24_Канал_(сайт)).
10. HTML5 [Електронний ресурс]. – Режим доступу: <https://uk.wikipedia.org/wiki/HTML>.
11. CSS3 [Електронний ресурс]. – Режим доступу: <https://uk.wikipedia.org/wiki/CSS>.
12. Single Page Application [Електронний ресурс]. – Режим доступу: https://en.wikipedia.org/wiki/Single-page_application.
13. JavaScript [Електронний ресурс]. – Режим доступу: <https://uk.wikipedia.org/wiki/JavaScript>.

14. ECMAScript [Електронний ресурс]. – Режим доступу: <https://uk.wikipedia.org/wiki/ECMAScript>.
15. JS Фреймворки [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Порівняння_фреймворків_JavaScript.
16. State Management [Електронний ресурс]. – Режим доступу: https://en.wikipedia.org/wiki/State_management.
17. React.js [Електронний ресурс]. – Режим доступу: <https://uk.wikipedia.org/wiki/React>.
18. Vue.js [Електронний ресурс]. – Режим доступу: <https://uk.wikipedia.org/wiki/Vue.js>.
19. Angular [Електронний ресурс]. – Режим доступу: <https://uk.wikipedia.org/wiki/AngularJS>.
20. SPA, Angular застосунок своїми руками за 2 години [Електронний ресурс]. – Режим доступу: <https://codeguida.com/post/1509>.
21. JSX [Електронний ресурс]. – Режим доступу: <https://reactjs.org/docs/jsx-in-depth.html>.
22. DOM [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Об'єктна_модель_документа.
23. Microsoft Visual Studio Code [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Visual_Studio_Code.
24. NewsAPI [Електронний ресурс]. – Режим доступу: <https://newsapi.org/>.
25. OpenWeatherAPI [Електронний ресурс]. – Режим доступу: <https://openweathermap.org/>.
26. Chart.js [Електронний ресурс]. – Режим доступу: <https://www.chartjs.org/docs/latest/developers/api.html>.

ДОДАТОК А
Текст програми

A.1 Package.json – конфігурація проекту

```
{  
  "name": "all-news",  
  "version": "0.1.0",  
  "private": true,  
  "dependencies": {  
    "@testing-library/jest-dom": "^5.11.4",  
    "@testing-library/react": "^11.1.0",  
    "@testing-library/user-event": "^12.1.10",  
    "axios": "^0.21.1",  
    "bootstrap": "^4.6.0",  
    "chart.js": "^3.2.1",  
    "clsx": "^1.1.1",  
    "connected-react-router": "^6.8.0",  
    "formik": "^2.2.6",  
    "lodash": "^4.17.21",  
    "materialize-css": "^1.0.0-rc.2",  
    "node-sass": "4.14.1",  
    "prettier": "^2.2.1",  
    "react": "^17.0.1",  
    "react-bootstrap": "^1.4.3",  
    "react-dom": "^17.0.1",  
    "react-infinite-scroller": "^1.2.4",  
    "react-redux": "^7.2.2",  
    "react-redux-toastr": "^7.6.5",  
    "react-router-dom": "^5.2.0",  
    "react-scripts": "4.0.1",  
    "react-scrollbars-custom": "^4.0.25",  
    "redux": "^4.0.5",  
    "redux-logger": "^3.0.6",
```

```

    "redux-thunk": "^2.3.0",
    "svg-sprite": "^1.5.0",
    "swiper": "^6.4.8",
    "web-vitals": "^0.2.4"
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject",
    "sprite-gen": "node scripts/generateSvgSprite.js"
  },
  "eslintConfig": {
    "extends": [
      "react-app",
      "react-app/jest"
    ]
  },
  "browserslist": {
    "production": [
      ">0.2%",
      "not dead",
      "not op_mini all"
    ],
    "development": [
      "last 1 chrome version",
      "last 1 firefox version",
      "last 1 safari version"
    ]
  }
}

```

A.2 Компонент App.js

```

import BreakingNews from "modules/BreakingNews/BreakingNews";
import Header from "modules/Header/Header";
import Notes from "modules/Notes/Notes";
import React, { useEffect } from "react";
import MainContainer from "modules/MainContainer/MainContainer";
import ReduxToastr from "react-redux-toastr";
import { Route, Switch } from "react-router-dom";
import routesByName from "routes/routesByName";
import Page404 from "modules/Page404";
import "materialize-css";
import { connect, Provider } from "react-redux";
import { initAppAction } from "modules/app/store/actions";
import Preloader from "components/Preloader/Preloader";
import Saved from "./modules/Saved/Saved";
import ExchangePage from "modules/ExchangePage/ExchangePage";
const App = ({ initApp, appInitied }) => {
  useEffect(() => {
    initApp();
  }, []);

  return (
    <>
      <Preloader />
      {appInitied ? (
        <div>
          <Header />
          <Switch>
            <Route exact path="/">

```

```

        <BreakingNews />
        <MainContainer />
    </Route>
    <Route exact path={routesByName.exchange}>
        <ExchangePage />
    </Route>
    <Route exact path={routesByName.saved}>
        <Saved />
    </Route>
    <Route exact path={routesByName.notes}>
        <Notes />
    </Route>
    <Route exact path="*">
        <Page404 />
    </Route>
</Switch>
<ReduxToastr          position="bottom-left"          newestOnTop
closeOnToastrClick />
    </div>
    ): null}
</>
);
};
const mapStateToProps = ({ app: { inited } }) => ({
    appInited: inited,
});

const mapDispatchToProps = {
    initApp: initAppAction,
};

```



```

        activeClassName={classes.activeLink}
      >
        Збережене
      </NavLink>
    </li>
    <li className="tac col s4">
      <NavLink
        to={routesByName.notes}
        className="fz-2"
        activeClassName={classes.activeLink}
      >
        Нотатки
      </NavLink>
    </li>
    <li className="tac col s4">
      <NavLink
        to={routesByName.exchange}
        className="fz-2"
        activeClassName={classes.activeLink}
      >
        Курс валют
      </NavLink>
    </li>
  </ul>
</div>
</nav>
</div>
<div className={classes.fakePadding} />
</>
);

```

```
};
```

```
export default Header;
```

A.4 Компонент головних новин BreakingNews.js

```
import BreakingNewsItem from
"modules/BreakingNews/BreakingNewsItem/BreakingNewsItem";
import React from "react";
import { Swiper, SwiperSlide } from "swiper/react";
import clsx from "clsx";
import "swiper/swiper.scss";
import classes from "./BreakingNews.module.scss";
import { connect } from "react-redux";

const BreakingNews = ({ breakingNewsList }) => {
  return (
    <div className="teal darken-2">
      <div className="fz-3 ml-4 pt-3 pb-1 text-teal-light align-center d-flex">
        <b className="text-color-main">Головне </b>
        <span className="ml-1 mr-1"> на </span>
        {new Date().toLocaleDateString("pt-PT")}
        <i className="material-icons ml-3 cup">refresh</i>
      </div>
      <div className="pl-4 pb-3">
        <Swiper
          spaceBetween={20}
          slidesPerView="auto"
          loop
          loopedSlides={3}
```

```

    noSwiping
    noSwipingClass="no-swipe"
  >
    {breakingNewsList.map(
      ({ publishedAt, title, description, urlToImage, url }) => {
        return urlToImage ? (
          <SwiperSlide key={publishedAt} className={clsx(classes.slide)}>
            <BreakingNewsItem
              title={title}
              desc={description}
              imgSrc={urlToImage}
              url={url}
            />
          </SwiperSlide>
        ) : null;
      }
    )}
  </Swiper>
</div>
</div>
);
};

const mapStateToProps = ({ breakingNews: { library } }) => ({
  breakingNewsList: library,
});

export default connect(mapStateToProps, {})(BreakingNews);

```

A.5 Компонент поискового бара SearchBar.js

```

import useCallbackRef from "hooks/useRefCallback";
import {
  updateMetaAction,
  updateSearchParamsAction,
} from "modules/MainContainer/NewsContainer/store/actions";
import React, { useCallback, useEffect, useRef, useState } from "react";
import { connect } from "react-redux";
import M from "materialize-css";
import classes from "./SearchBar.module.scss";
import { debounce } from "lodash";

const SearchBar = ({ updateSearchParams, updateMeta, searchParams }) => {
  const [searchQuery, setSearchQuery] = useState("");
  const searchInputRef = useRef();

  const [languageNode, languageRef] = useCallbackRef();
  const [countryNode, countryRef] = useCallbackRef();
  const [categoryNode, categoryRef] = useCallbackRef();

  useEffect(() => {
    M.FormSelect.init(languageNode);
    M.FormSelect.init(countryNode);
    M.FormSelect.init(categoryNode);
  }, [languageNode, countryNode, categoryNode]);
  console.dir(categoryNode);
  const clearAndFocusSearchInput = () => {
    setSearchQuery("");
    searchInputRef?.current?.focus();
  }

```

```

[languageNode, countryNode, categoryNode].forEach((option) => {
  option[0].selected = true;
});
};

useEffect(() => {
  document.addEventListener("CLEAR_INPUT", clearAndFocusSearchInput);

  return () => {
    document.removeEventListener("CLEAR_INPUT",
clearAndFocusSearchInput);
  };
}, []);

const triggerOnChange = useCallback(
  // eslint-disable-next-line
  debounce((value) => {
    updateSearchParams({ query: value });
    updateMeta({ offset: 0, hasMore: true });
  }, 2000),
  [updateSearchParams, updateMeta]
);

const handleChangeSearchInput = useCallback(
  ({ currentTarget }) => {
    setSearchQuery(currentTarget.value);

    triggerOnChange(currentTarget.value);
  },

```

```

    [triggerOnChange]
  );

const handleSubmit = useCallback(() => {
  updateSearchParams({ query: searchQuery });
  updateMeta({ offset: 0, hasMore: true });
}, [updateMeta, updateSearchParams]);

const handleChangeSelect = useCallback(
  (e) => {
    const type = e.target.dataset.cat;
    const value = e.target.value;

    const newParam = { [type]: value };
    updateSearchParams(newParam);

    if (searchParams.query.length !== 0) {
      updateMeta({ offset: 0, hasMore: true });
    }
  },
  [searchParams, updateSearchParams]
);

const handleKey = useCallback(
  async (e) => {
    if (e.keyCode === 13) {
      await handleSubmit();
    }
  },
  [handleSubmit]
);

```

```
);
```

```
return (
  <div className={classes.root}>
    <input
      type="text"
      value={searchQuery}
      ref={searchInputRef}
      onChange={handleChangeSearchInput}
      style={{ padding: "16px 24px 16px 50px", height: "auto" }}
      placeholder='Уведіть запит, наприклад "Новини в Запоріжжі" '
      onKeyDown={handleKey}
    />
    <div className={classes.icon}>
      <i className="material-icons">search</i>
    </div>
    <div className={classes.submitBtn} onClick={handleSubmit}>
      Пошук
    </div>

    <div className="mt-2 jcsb">
      <div className={` ${classes.filter} input-field`} >
        <span>Категорія</span>

        <select
          ref={categoryRef}
          onChange={handleChangeSelect}
          data-cat="category"
        >
          <option value="general">Загальне</option>
```

```

    <option value="business">Бізнес</option>
    <option value="entertainment">Розваги</option>
    <option value="health">Здоров'я</option>
    <option value="science">Наука</option>
    <option value="sports">Спорт</option>
    <option value="technology">Технології</option>
  </select>
</div>

<div className={` ${classes.filter} input-field`} >
  <span>Мова</span>

  <select
    ref={languageRef}
    onChange={handleChangeSelect}
    data-cat="language"
  >
    <option value="">Усі</option>
    <option value="ua">Україна</option>
    <option value="de">Німецька</option>
    <option value="en">Англійська</option>
    <option value="ru">Російська</option>
    <option value="it">Італійська</option>
  </select>
</div>

<div className={` ${classes.filter} input-field`} >
  <span>Країна</span>

  <select
    ref={countryRef}
    onChange={handleChangeSelect}

```

```

    data-cat="country"
  >
    <option value="">Усі</option>
    <option value="ua">Україна</option>
    <option value="ru">Росія</option>
    <option value="us">США</option>
    <option value="de">Німеччина</option>
    <option value="pl">Польща</option>
    <option value="cz">Чехія</option>
    <option value="be">Білорусь</option>
  </select>
</div>
</div>
</div>
);
};

const mapStateToProps = ({ mainNews: { searchParams } }) => ({
  searchParams,
});

export default connect(mapStateToProps, {
  updateSearchParams: updateSearchParamsAction,
  updateMeta: updateMetaAction,
})(SearchBar);

```

A.6 Компонент пошуку новин NewsContainer.js

```

import                                NewsItem                                from
"modules/MainContainer/NewsContainer/NewsItem/NewsItem";

```

```

import { saveNewsAction } from "modules/Saved/store/actions";
import React, {
  useCallback,
  useEffect,
  useMemo,
  useRef,
  useState,
} from "react";
import { connect } from "react-redux";
import { _v } from "utils/short";
import classes from "./NewsContainer.module.scss";
import {
  setMainNewsAction,
  loadFirstNewsAction,
  updateSearchParamsAction,
} from "modules/MainContainer/NewsContainer/store/actions";
import breakingNewsServices from
"modules/BreakingNews/breakingNewsServices";
import errorToastr from "libs/toastr/errorToastr";

const defaultQuery = "Коронавирус";
const NewsContainer = ({
  newsList,
  searchParams,
  meta,
  saveNews,
  setMainNews,
  loadFirstNews,
  updateSearchParams,
}) => {

```

```

const [secondaryNews, setSecondaryNews] = useState([]);
const [needSecondary, setNeedSecondary] = useState(false);

const handleDispatchClearInput = useCallback(() => {
  const event = new Event("CLEAR_INPUT", { bubbles: true });
  document.dispatchEvent(event);
}, []);

const handleSetNewsByDefaultQuery = useCallback(async () => {
  try {
    handleDispatchClearInput();
    const {
      data: { articles },
    } = await breakingNewsServices.getAllNewsByQuery({
      query: defaultQuery,
    });
    updateSearchParams({
      query: defaultQuery,
      category: "",
      language: "",
      country: "",
    });
    setMainNews(articles);
  } catch (e) {
    errorToastr("Ошибка", e.message);
  }
}, [setMainNews, handleDispatchClearInput]);

useEffect(() => {
  if (needSecondary) {

```

```

(async () => {
  try {
    const {
      data: { articles },
    } = await breakingNewsServices.getAllNewsByQuery({
      query: searchParams.query,
    });
    setSecondaryNews(articles);
    setNeedSecondary(false);
  } catch (e) {
    errorToastr("Ошибка", e.message);
  }
})();
}, [needSecondary, searchParams.query]);

```

```

useEffect(() => {
  if (newsList.length === 0) {
    setNeedSecondary(true);
  } else {
    setNeedSecondary(false);
    setSecondaryNews([]);
  }
}, [newsList.length]);

```

```

const shouldLoad = useMemo(
  () =>
    meta.offset === 0 &&
    meta.hasMore &&
    !meta.loading &&

```

```

    searchParams.query.length !== 0,
    [meta, searchParams.query]
  );

useEffect(() => {
  if (!shouldLoad) {
    return;
  }

  (async () => {
    await loadFirstNews();
  })();
}, [shouldLoad, loadFirstNews]);

return (
  <div className={classes.root}>
    <div className={classes.title}>
      {searchParams.query ? (
        <div>
          Новини за запитом: <span>{searchParams.query}</span>
        </div>
      ) : null}
    </div>
    <div className="container">
      {newsList.length > 0 ? (
        <div className={classes.newsItems}>
          {_v(newsList).map((news) => {
            return news.urlToImage ? (
              <div key={news.publishedAt || news.title}>
                <NewsItem {...news} onSave={saveNews} />

```

```

        </div>
      ) : null;
    }}
  </div>
): (
  <div
    className="tac font-weight-bold m0-auto"
    style={{ fontSize: 30 }}
  >
    <div> Нічого не знайдено за заданими параметрами</div>
    <div
      className="text-color-main cup"
      onClick={handleSetNewsByDefaultQuery}
    >
      Спробуйте ще раз
    </div>
    {searchParams.query.length && secondaryNews.length > 0 ? (
      <>
        <div>
          Усі новини за запитом: <span>{searchParams.query}</span>
        </div>
        <div className={classes.newsItems}>
          {secondaryNews.map((news) => {
            return news.urlToImage ? (
              <div key={news.publishedAt || news.title}>
                <NewsItem {...news} onSave={saveNews} />
              </div>
            ) : null;
          })}
        </div>
      ) : null;
    )}
  </div>

```

```

        </>
      ) : null}
    </div>
  )}
</div>
</div>
);
};

```

```

const mapStateToProps = ({ mainNews: { list, meta, searchParams } }) => ({
  newsList: list,
  meta,
  searchParams,
});
export default connect(mapStateToProps, {
  saveNews: saveNewsAction,
  setMainNews: setMainNewsAction,
  updateSearchParams: updateSearchParamsAction,
  loadFirstNews: loadFirstNewsAction,
})(NewsContainer);

```

A.7 Компонент карточки новини NewsItem.js

```

import successToastr from "libs/toastr/successToastr";
import React, { useCallback } from "react";
import classes from "./NewsItem.module.scss";
const NewsItem = ({
  source: { name: author },
  urlToImage,
  title,

```

```
publishedAt,  
url,  
description,  
onSave,  
onDelete,  
saveDate,  
}) => {  
  const handleSave = useCallback(  
    (e) => {  
      e.preventDefault();  
      onSave({  
        source: { name: author },  
        urlToImage,  
        title,  
        publishedAt,  
        url,  
        description,  
      });  
      successToastr("Новина збережена");  
    },  
    [onSave]  
  );  
  const handleDelete = useCallback(  
    (e) => {  
      e.preventDefault();  
      onDelete(publishedAt);  
      successToastr("Новина видалена");  
    },  
    [onDelete]  
  );
```

```

return (
  <div className={classes.root}>
    <div className={classes.image}>
      <img src={urlToImage} alt={title} />
    </div>
    <div className={classes.body}>
      <div>
        <div className={classes.title}>{title}</div>
        <div className={classes.description}>{description}</div>
      </div>
      <ul>
        <li>
          Автор: <span>{author}</span>
        </li>
        <li>
          Дата публікації: {" "}
          <span>{new Date(createdAt).toLocaleDateString("pt-PT")}</span>
        </li>
        {saveDate && (
          <li>
            Дата збереження: {" "}
            <span>{new Date(saveDate).toLocaleString()}</span>
          </li>
        )}
      </ul>
      <div className={classes.controls}>
        <div>
          <a
            href={url}
            target="_blank"

```

```

    rel="noreferrer"
    className="waves-effect waves-light btn"
  >
    <i className="material-icons right">link</i>Перейти до оригіналу
  </a>
</div>
<div>
  {onSave && (
    <a
      href="#"
      className="waves-effect waves-light btn"
      onClick={handleSave}
    >
      <i className="material-icons right">save</i>Зберегти
    </a>
  )}
  {onDelete && (
    <a
      href="#"
      className="waves-effect waves-light btn"
      onClick={handleDelete}
    >
      <i className="material-icons right">delete</i>Видалити
    </a>
  )}
</div>
</div>
</div>
</div>
);

```

```
};
```

```
export default NewsItem;
```

A.8 Компонент погоды Weather.js

```
import React, { useEffect, useState } from "react";
import weatherServices from "modules/MainContainer/Weather/weatherServices";
import clsx from "clsx";
import classes from "./Weather.module.scss";

const Weather = (props) => {
  const [weatherData, setWeatherData] = useState(null);

  const [loading, setLoading] = useState(false);

  useEffect(() => {
    setLoading(true);

    const geoSuccess = async ({ coords: { latitude, longitude } }) => {
      const weather = await weatherServices.getCurrentWeather(
        latitude,
        longitude
      );
      const copyWeather = { ...weather };

      let { temp_max, temp_min, feels_like } = copyWeather.main;

      if (temp_max > 0) {
        temp_max = `+${temp_max.toFixed(0)}°`;
      }
    };
  }, []);
}
```

```

    }
    if (temp_min > 0) {
      temp_min = `+${temp_min.toFixed(0)}`;
    }
    if (feels_like > 0) {
      feels_like = `+${feels_like.toFixed(0)}`;
    }

    copyWeather.temp = { temp_max, temp_min, feels_like };
    setWeatherData(copyWeather);
    setLoading(false);
  };

  const geoFail = () => {
    setLoading(false);
  };

  navigator.geolocation.getCurrentPosition(geoSuccess, geoFail);
}, [setWeatherData]);

return loading ? (
  <div>Loading...</div>
) : (
  <div className={classes.root}>
    {weatherData ? (
      <div>
        <div className={classes.title}>
          Погода в місті <b>{weatherData.name}</b>
        </div>
        <ul className={classes.body}>

```

```

<li className={classes.icon}>
  <img
    src={`http://openweathermap.org/img/wn/${weatherData?.weather[0]?.icon}@2x.png`}
    alt="icon"
  />
</li>
<li className={clsx(classes.item, classes.temp)}>
  {weatherData.temp.temp_max === weatherData.temp.temp_min ? (
    <span>{weatherData.temp.temp_min} °C</span>
  ) : (
    <span>
      {weatherData.temp.temp_min}°C ... {
        {weatherData.temp.temp_max}
        °C
      }
    </span>
  )}
</li>
<li className={classes.item}>
  Відчувається як: <b>{weatherData.temp.feels_like} °C</b>
</li>
<li className={classes.item}>
  Вітер: <b>{weatherData.wind.speed} м/с</b>
</li>
<li className={classes.item}>
  Вологість: <b>{weatherData.main.humidity} %</b>
</li>
</ul>
</div>

```

```

): (
  <div>
    <div className="tac align-center mt-5">
      Будь ласка, дайте дозвіл цьому сайту до геоданих
    </div>
  </div>
)}
</div>
);
};

```

```
export default Weather;
```

A.9 Компонент карточки курсів валют ExchangeRate.js

```

import clsx from "clsx";
import errorToastr from "libs/toastr/errorToastr";
import exchangeRateServices from
"modules/MainContainer/ExchangeRate/exchangeRateServices";
import React, { useEffect, useState } from "react";
import classes from "./ExchangeRate.module.scss";

const ExchangeRate = (props) => {
  const [exchangeData, setExchangeData] = useState(null);

  useEffect(() => {
    try {
      (async () => {
        const { data } = await exchangeRateServices.getExchangeRates("5");
        setExchangeData(data);
      })();
    }
  });

```

```

    } catch (e) {
      errorToastr("Error", e.message);
    }
  }, []);

```

```

return exchangeData ? (
  <div className={classes.root}>
    <div className={classes.title}>Курс валют на сьогодні</div>
    <div className={classes.body}>
      <div className={clsx(classes.row)}>
        <div className={classes.item}>
          <b>Валюта</b>
        </div>
        <div className={classes.item}>
          <b>Купівля</b>
        </div>
        <div className={classes.item}>
          <b>Продаж</b>
        </div>
      </div>
      {exchangeData.map(({ ccy, buy, sale }) => (
        <div key={ccy} className={classes.row}>
          <div className={classes.item}>
            <b>{ccy}</b>
          </div>
          <div className={classes.item}>{Number(buy).toFixed(2)}</div>
          <div className={classes.item}>{Number(sale).toFixed(2)}</div>
        </div>
      ))}
    </div>

```

```

    </div>
  ) : (
    <div className="">Loading...</div>
  );
};

```

```
export default ExchangeRate;
```

A.10 Компонент карточки фото дня DayPhoto.js

```

// import { Materialbox } from "materialize-css";
import errorToastr from "libs/toastr/errorToastr";
import dayPhotoSevices from
"modules/MainContainer/DayPhoto/dayPhotoServices";
import React, { useEffect, useState } from "react";
import classes from "./DayPhoto.module.scss";
import M from "materialize-css";
import useCallbackRef from "hooks/useRefCallback";
const DayPhoto = () => {
  const [photoLink, setPhotoLink] = useState(null);

  useEffect(() => {
    (async () => {
      try {
        const {
          data: {
            urls: { small: photo },
          },
        } = await dayPhotoSevices.getDayPhoto();

```

```

    setPhotoLink(photo);
  } catch (e) {
    errorToastr("Error", e.message);
  }
})();
}, []);

const [imgNode, setRef] = useCallbackRef();

useEffect(() => {
  M.Materialbox.init(imgNode);
}, [imgNode]);

return photoLink ? (
  <div className={classes.root}>
    <div className={classes.title}>Фото дня</div>
    <div className={classes.dayPhoto}>
      <img
        className="materialboxed"
        ref={setRef}
        src={photoLink}
        alt="Фото дня"
      />
    </div>
  </div>
) : null;
};

export default DayPhoto;

```

A.11 Компонент сторінки нотаток Notes.js

```
import {
  deleteSavedNoteAction,
  saveNoteAction,
} from "modules/Notes/store/actions";
import React, { useCallback, useState } from "react";
import { connect } from "react-redux";
import { _v } from "utils/short";
import { Formik, Field, Form } from "formik";
import classes from "./Notes.module.scss";

const Notes = ({ noteList, saveNote, deleteNote }) => {
  const [initialVal, setInitialVal] = useState({ subject: "", body: "" });
  const handleSubmit = useCallback(
    (values, { resetForm }) => {
      saveNote(values);
      resetForm();
    },
    [saveNote]
  );
  return (
    <div className={classes.root}>
      <div className={classes.title}>Нотатки</div>
      <div className={classes.newNote}>
        <Formik initialValues={initialVal} onSubmit={handleSubmit}>
          ({ { handleReset } }) => (
            <Form>
              <div className={classes.fieldsContainer}>
                <Field
```

```

        name="subject"
        placeholder="Тема"
        className={classes.field}
        component="textarea"
      />
      <Field
        name="body"
        placeholder="Що бажаєте записати?"
        className={classes.field}
        rows={10}
        component="textarea"
      />
      <div className="jcsb">
        <button
          type="button"
          onClick={handleReset}
          className="waves-effect waves-light btn"
        >
          Очистити
        </button>
        <button
          type="submit"
          className="waves-effect waves-light btn"
        >
          Додати
        </button>
      </div>
    </div>
  </Form>
)}

```

```

    </Formik>
  </div>
  <div className="container">
    <div className={classes.noteContainer}>
      {_v(noteList).map((note) => {
        return (
          <div className={classes.note}>
            <div className={classes.noteTitle}>{note.subject}</div>
            <div className={classes.noteBody}>{note.body}</div>
            <div className="jcsb mt-3 align-center">
              <div className={classes.createdAt}>
                <b> Створено: </b>
                {new Date(note.createdAt).toLocaleDateString("pt-PT")}
              </div>
              <button
                type="button"
                className="waves-effect waves-light btn"
                onClick={() => deleteNote(note.createdAt)}
              >
                <i className="material-icons right">delete</i>
                Видалити
              </button>
            </div>
          </div>
        );
      })}
    </div>
  </div>
  </div>
);

```

```

};

const mapStateToProps = ({ notes: { notes } }) => ({
  noteList: notes,
});

export default connect(mapStateToProps, {
  saveNote: saveNoteAction,
  deleteNote: deleteSavedNoteAction,
})(Notes);

```

A.12 Компонент сторінки курсів валют ExchangePage.js

```

import React, { useCallback, useEffect, useRef, useState } from "react";
import classes from "./ExchangePage.module.scss";

import                exchangePageService                from
"modules/ExchangePage/exchangePageService";

import errorToastr from "libs/toastr/errorToastr";
import Chart from "chart.js/auto";
import { useLocalStorage } from "hooks/useLocalStorage";
import localStorageKeys from "constants/constants";
import { _v } from "utils/short";
import clsx from "clsx";

const mainCurrencyCode = ["EUR", "USD", "RUB", "PLN"];

const ignoreCodes = ["XPD", "XAU", "XPT", "XAG"];

const transformCoursesDataForCanvas = (coursesArray) => {
  return coursesArray.reduce((accum, currValue) => {
    currValue.forEach(({ exchangedate, cc, rate }) => {

```

```

    if (ignoreCodes.includes(cc)) {
      return accum;
    }
    if (accum[cc]) {
      accum[cc] = { ...accum[cc], [exchangedate]: rate };
    } else {
      accum[cc] = { [exchangedate]: rate };
    }
  });

  return accum;
}, {});
};

const ExchangePage = () => {
  const [mainCourses, setMainCourses] = useState({});
  const [allCourses, setAllCourses] = useState({});
  const [showingCourses, setShowingCourses] = useState({});
  const { setItem, getItem } = useLocalStorage();
  const canvasRef = useRef();
  const chartRef = useRef();

  useEffect(() => {
    if (mainCourses) {
      setShowingCourses(mainCourses);
    }
  }, [mainCourses]);

  useEffect(() => {
    const currentDate = new Date()

```

```

.toLocaleDateString("pt-PT")
.replaceAll("/", ".");

const coursesFromStorage = getItem(localStorageKeys.courses);

if (
  coursesFromStorage &&
  coursesFromStorage["EUR"] &&
  coursesFromStorage["EUR"][currentDate]
) {
  const listCopy = { ...coursesFromStorage };
  Object.keys(listCopy).forEach((currency) => {
    if (!mainCurrencyCode.includes(currency)) {
      delete listCopy[currency];
    }
  });
  setMainCourses(listCopy);
  setAllCourses(coursesFromStorage);

  return;
} else {
  (async () => {
    try {
      const datesForPrev30Days = Array(30)
        .fill(null)
        .map((_, i) => {
          const date = new Date();
          date.setDate(date.getDate() - i);
          return date.toLocaleDateString("en-ZA").replaceAll("/", "");
        });
    }
  });
}

```

```

const promiseArr = datesForPrev30Days.map((date) =>
  exchangePageService.getDataByDay(date)
);

const coursesForAllDates = await Promise.all(promiseArr);

const transformedData = transformCoursesDataForCanvas(
  coursesForAllDates
);

const listCopy = { ...transformedData };
Object.keys(listCopy).forEach((currency) => {
  if (!mainCurrencyCode.includes(currency)) {
    delete listCopy[currency];
  }
});
setMainCourses(listCopy);
setAllCourses(transformedData);

setItem(localStorageKeys.courses, transformedData);
} catch (e) {
  errorToastr("Помилка запиту", e.message);
}
})();
}, []);

const createCanvas = useCallback(() => {
  const canvas = canvasRef.current.getContext("2d");

```

```

if (chartRef.current) {
  chartRef.current.destroy();
}

const preparedDatasets = Object.keys(showingCourses).map((curr) => {
  const backgroundColor = `#${Math.floor(Math.random() *
16777215).toString(
    16
  )}`;
  return {
    label: curr,
    data: _v(showingCourses[curr]),
    backgroundColor,
    borderColor: backgroundColor,
  };
});

const dataForCanvas = {
  labels: Object.keys(showingCourses["EUR"]),
  datasets: preparedDatasets,
};

chartRef.current = new Chart(canvas, {
  type: "line",
  data: dataForCanvas,
  options: {},
});
}, [chartRef, canvasRef, showingCourses]);

```

```

const toggleShowingCourses = useCallback(() => {
  if (Object.keys(showingCourses).length === 4) {
    setShowingCourses(allCourses);
  } else {
    setShowingCourses(mainCourses);
  }
  createCanvas();
}, [createCanvas, showingCourses]);

useEffect(() => {
  if (canvasRef && showingCourses && showingCourses["EUR"]) {
    createCanvas();
  }
}, [showingCourses, createCanvas]);

const [exp, setExp] = useState(false);

useEffect(() => {
  if (canvasRef && chartRef.current) {
    if (exp) {
      canvasRef.current.height = 3000;
    } else {
      canvasRef.current.height = 1000;
    }
    chartRef.current.update();
  }
}, [exp]);

return (
  <div className={classes.root}>

```

```

<div className={classes.title}>Курси валют</div>
<div
  onClick={toggleShowingCourses}
  className="waves-effect waves-light btn tac mb-4 ml-4 "
>
  {_v(showingCourses).length === 4 ? "Більше валют" : "Основні
валюти"}
</div>
<div
  onClick={() => setExp((prev) => !prev)}
  className="waves-effect waves-light btn tac mb-4 ml-4 "
>
  {exp ? "Згорнути" : "Розгорнути"}
</div>
<div className={clsx(classes.canvasRoot, { [classes.expansive]: exp
})}>
  <canvas ref={canvasRef}>Нет поддержки канвас</canvas>
</div>
</div>
);
};

```

```
export default ExchangePage;
```

A.13 Компонент збережених новин SavedNews.js

```

import NewsItem from
"modules/MainContainer/NewsContainer/NewsItem/NewsItem";
import { deleteSavedNewsAction } from "modules/Saved/store/actions";
import React from "react";

```

```

import { connect } from "react-redux";
import { _v } from "utils/short";
import classes from "./Saved.module.scss";

const SavedNews = ({ newsList, onDeleteNews }) => {
  return (
    <div className={classes.root}>
      <div className={classes.title}>Збережені новини</div>
      <div className="container">
        <div className={classes.newsContainer}>
          {_v(newsList).map((news) =>
            news.urlToImage ? (
              <div key={news.publishedAt ?? news.title}>
                <NewsItem {...news} onDelete={onDeleteNews} />
              </div>
            ) : null
          )}
        </div>
      </div>
    </div>
  );
};

const mapStateToProps = ({ savedNews: { listMap } }) => ({
  newsList: listMap,
});

export default connect(mapStateToProps, {
  onDeleteNews: deleteSavedNewsAction,
})(SavedNews);

```

A.14 КОМПОНЕНТ ГОЛОВНОЇ СТОРІНКИ MainContainer.js

```

import React from "react";

import                      ExchangeRate                      from
"modules/MainContainer/ExchangeRate/ExchangeRate";

import Weather from "modules/MainContainer/Weather/Weather";
import classes from "./MainContainer.module.scss";
import DayPhoto from "modules/MainContainer/DayPhoto/DayPhoto";
import SearchBar from "modules/MainContainer/SearchBar/SearchBar";
import                      NewsContainer                      from
"modules/MainContainer/NewsContainer/NewsContainer";

const MainContainer = (props) => {
  return (
    <div className={classes.root}>
      <div className="jcsb container mt-4">
        <ExchangeRate />
        <Weather />
        <DayPhoto />
      </div>
      <div className="mt-5 mb-3">
        <SearchBar />
        <NewsContainer />
      </div>
    </div>
  );
};

export default MainContainer

```

ДОДАТОК Б
Слайди презентації

Національний університет «Запорізька політехніка»
Кафедра ПЗ

Випускна кваліфікаційна робота бакалавра на тему:
РОЗРОБКА WEBЗАСТОСУНКУ ДЛЯ ПЕРЕГЛЯДУ ТА
ВЗАЄМОДІЇ З ІНФОРМАЦІЙНИМ ПОРТАЛОМ

Виконав:
ст. гр. КНТ-137

А.Я. Боберський

Керівник:
к.т.н., доцент

О.М. Гладкова

Рисунок Б.1 – Слайд 1

МЕТА РОБОТИ



- **Мета роботи** – створення програмного застосунку для пошуку та перегляду новин з можливістю широкого налаштування фільтрів та збереження у локальне сховище.
- **Об’єкт дослідження** – програмні засоби для пошуку та перегляду новин.
- **Предмет дослідження** – вебзастосунки та веб-сайти для пошуку та перегляду новин.

Рисунок Б.1 – Слайд 2

**ДЛЯ ДОСЯГНЕННЯ ПОСТАВЛЕНОЇ МЕТИ НЕОБХІДНО
ВИКОННАТИ НАСТУПНІ ЗАДАЧІ**

- дослідити предметну область;
- виконати аналіз процесу розробки веб-застосунку;
- розробити структуру пояснювальної записки;
- виконати програмну реалізацію проекту;
- провести тестування програмного продукту.

Рисунок Б.3 – Слайд 3

ПОРІВНЯННЯ ІСНУЮЧИХ АНАЛОГІВ

Функціональні можливості	Назва програмних продуктів				
	Українська Правда	Gazeta.ua	Страна.ua	24 канал	Новий продукт
Зручний дизайн	+	+	-	+	+
Легкочитаємий текст	+	+	-	+	+
Пошук за ключовою фразою	+	+	+	+	+
Відсутність реклами	-	-	-	-	+
Зручна фільтрація статей	+-	-	-	+	+
Порівняння статей	-	-	-	-	+
Адаптивність до різних екранів пристроїв	+	-	+	+	+
Додаткові дані (погода, курси валют)	-	+	-	-	+
Посилання на джерело	+	-	-	+	+
Додатковий функціонал	-	-	-	+	+

Рисунок Б.4 – Слайд 4

ПОРІВНЯННЯ НАЙПОШИРЕНІШИХ ФРЕЙМВОРКІВ JAVASCRIPT

Фреймворк / бібліотека	Переваги	Недоліки
1	2	3
Angular	Велика спільнота Декларативний стиль коду Наявність готових рішень Компонентний підхід Повноцінний фреймворк	Великий обсяг результуючого коду. Стисла, незручна документація. Великий поріг входу.
React	Віртуальний DOM Велика спільнота швидкість рендерингу Велика кількість допоміжних інструментів та готових рішень	Є бібліотекою для рендеру. Розробники, іноді, мають занадто великий вибір, не має жорстко заданих правил
Vue	Зрозуміла документація Крихітний розмір. Vue.js може важити близько 20 КБ. Чудова інтеграція. Vue.js можна використовувати як для створення однострінкових додатків, так і для складніших веб- інтерфейсів додатків.	Ризик надмірної гнучкості. Іноді у Vue.js можуть виникати проблеми при інтеграції в величезні проекти, а досвіду про можливі рішення до сих пір немає. Не хватка ресурсів

Рисунок Б.5 – Слайд 5

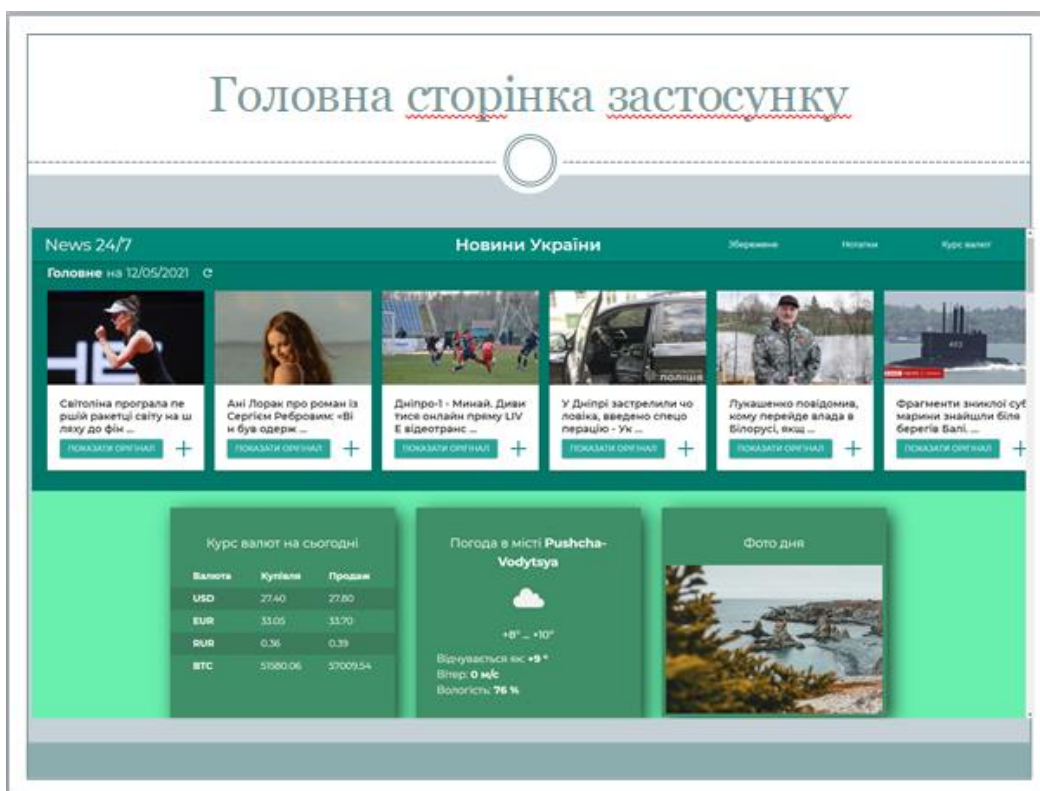


Рисунок Б.6 – Слайд 6

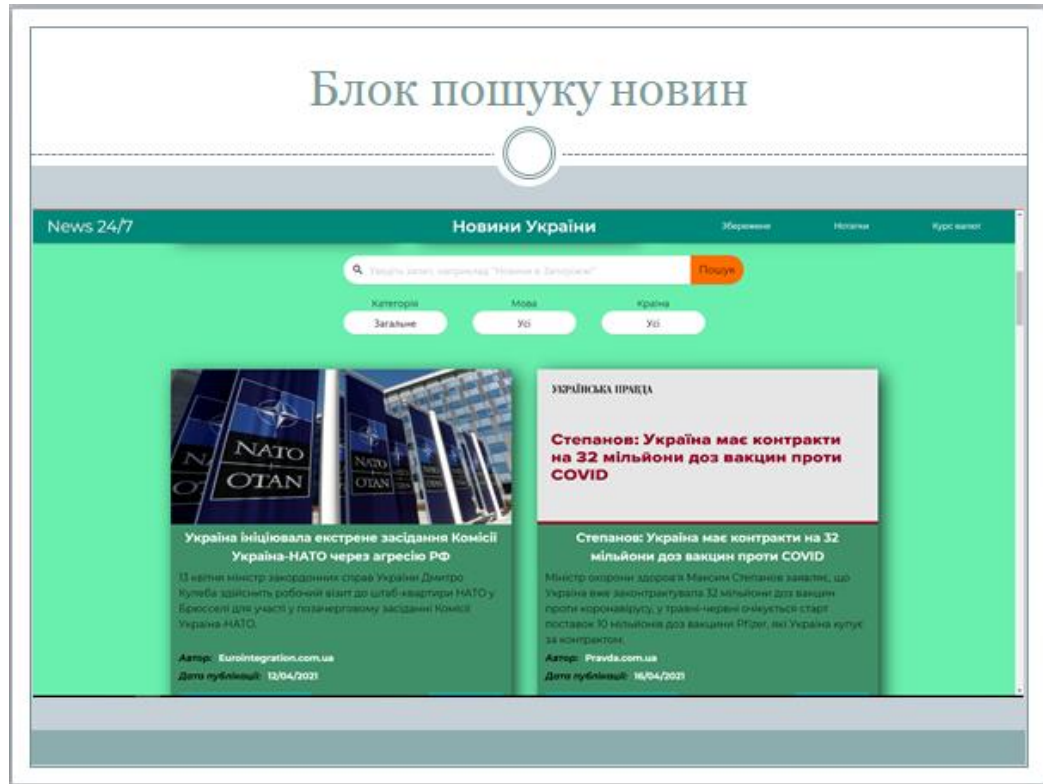


Рисунок Б.7 – Слайд 7

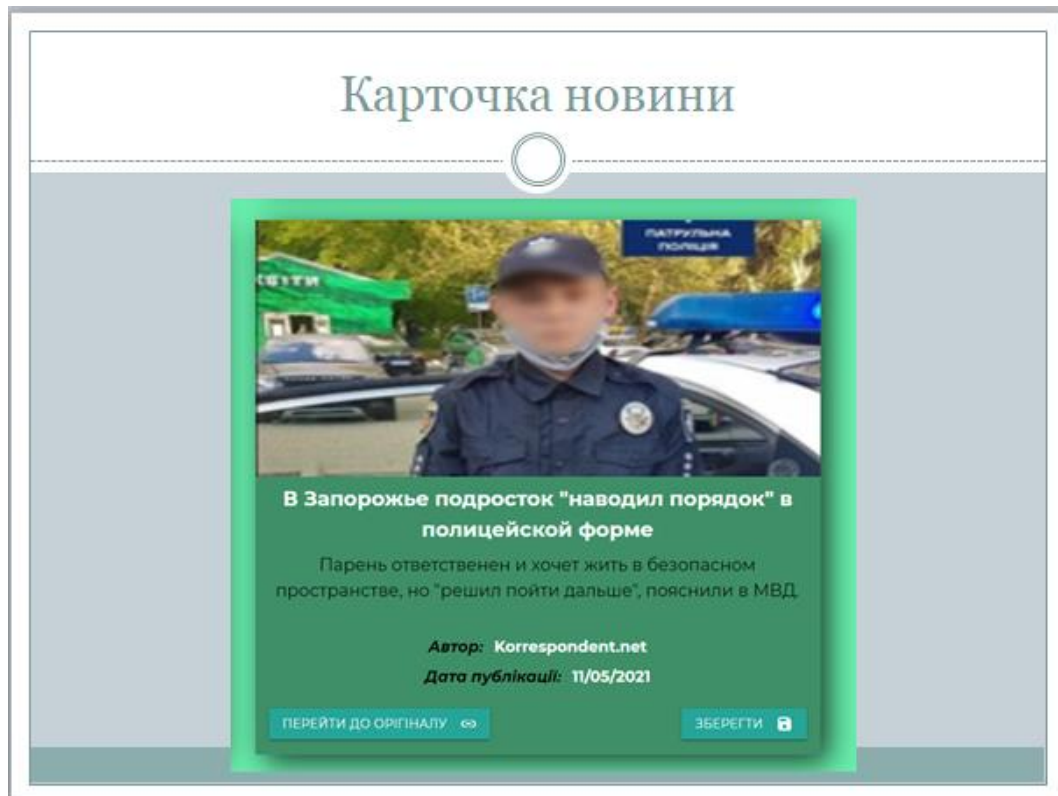


Рисунок Б.8 – Слайд 8

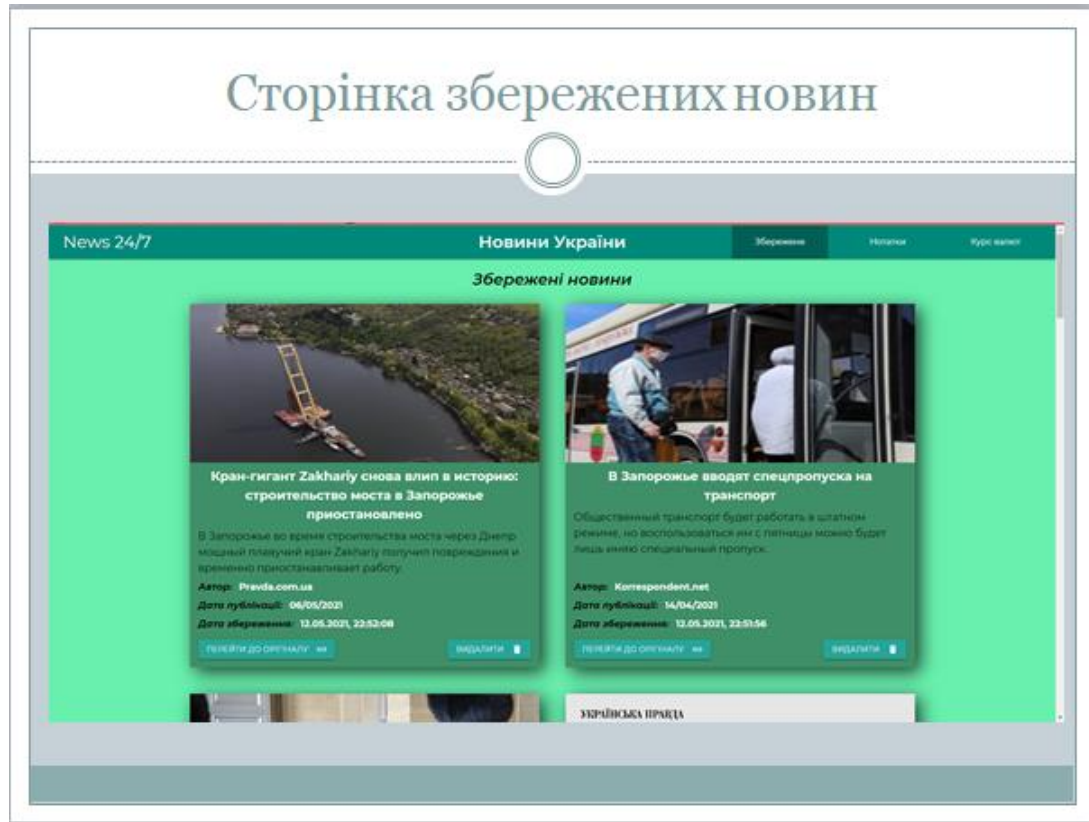


Рисунок Б.9 – Слайд 9

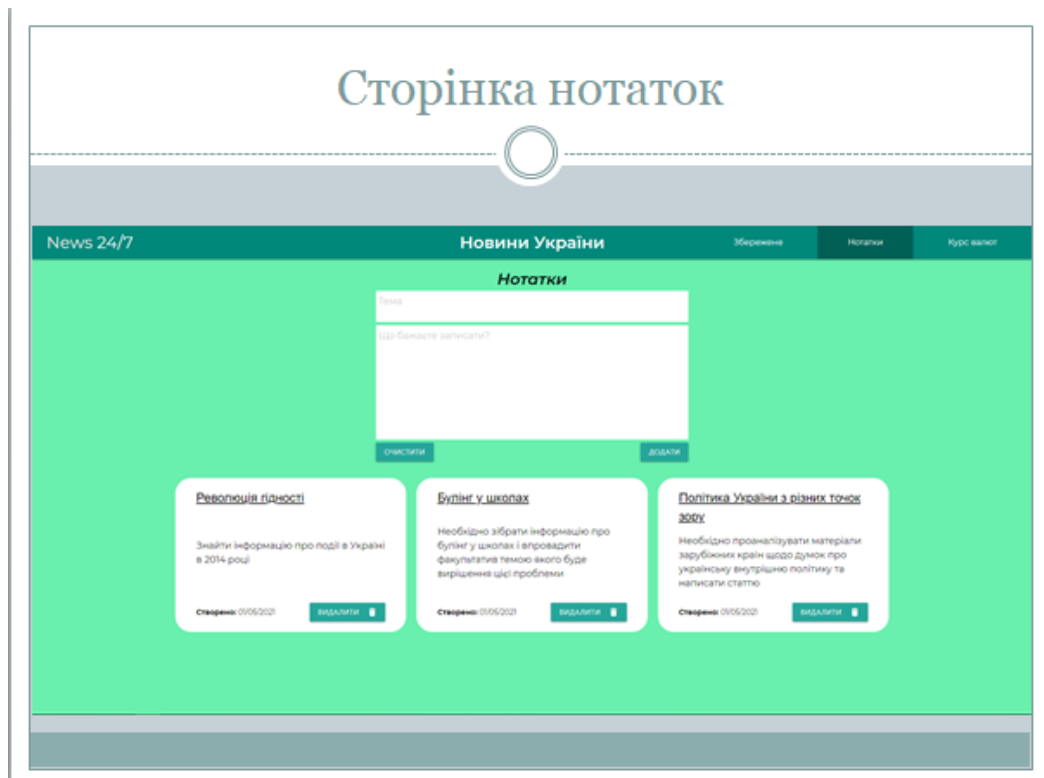


Рисунок Б.10 – Слайд 10

Сторінка курсів валют

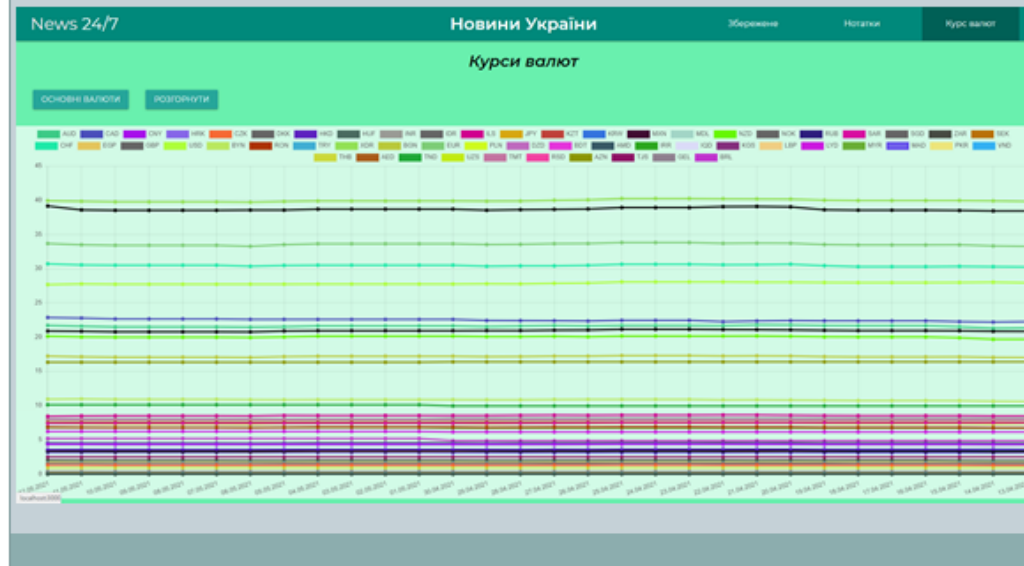


Рисунок Б.11 – Слайд 11

Висновки

- В ході виконання кваліфікаційної роботи бакалавра, було проведено огляд та аналіз існуючих систем для пошуку новин. Проведений аналіз показав, що майже усі існуючі програмні продукти мають недоліки. Тому було реалізовано новий програмний продукт, що позбавлений недоліків схожих сервісів та має нові модулі та розширені можливості.
- Складено технічне завдання, в якому зазначена мета розробки та її призначення, основні вимоги до програми, порядок її функціонування.
- Програма має дуже гарний і зрозумілий інтерфейс, який надасть легкості у роботі з масмедіа.
- Система може бути задіяна будь яким користувачем. Повністю дороблена та готова до змін у будь-який час, зручна у використанні та буде зрозуміла для кожного користувача, навіть якщо у нього немає навичок у використанні комп'ютеру.

Рисунок Б.12 – Слайд 12