МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних наук,
управління та адміністрування
Кафедра інформаційних технологій

# Кваліфікаційна робота бакалавра

на тему: Створення MVP- порталу новин

Виконав студент групи К-20і
спеціальності 122 Комп'ютерні науки
Скрипнік Михайло Сергійович

Керівник Штефан
Наталія Зінов'ївна

Консультант д.т.н., проф.
Казакова Надія Феліксівна

Рецензент д.т.н., професор
Мещеряков Володимир Іванович

Одеса 2022

**ЗМІСТ**

# ПЕРЕЛІК СКОРОЧЕНЬ ТА ТЕРМІНІВ

CSS – Cascading Style Sheets

HTML – HyperText Markup Language

MVC – Model-View-Controller

MVW – Model-View-Whatever

RTK Query – Redux Toolkit Query

UI – User Interface

Angular.js – фреймворк

C++ – мова програмування

JavaScript – мова програмування

MongoDB – система керування базами даних

Objective-C – мова програмування

Shadow DOM – тіньова модель документу

TypeScript – мова програмування

React.js – бібліотека

Vue.js – фреймворк

WebAssembly – мова програмування низького рівня

# ВСТУП

Зі швидким і винятковим зростанням технологій у всьому світі новинна індустрія зазнала багатьох змін. Більшість інформаційних агентств запустили онлайн-портал новин та запровадили електронні газети. Оскільки кількість людей, які користуються інтернетом, швидко збільшується з кожним днем, зростання онлайн-читачів та онлайн-порталів новин буде на висоті в найближчі роки.

Більшість індустрії новин і журналів виходять в Інтернет шляхом розробки та проектування послуг онлайн-порталів новин і введення електронних документів. У майбутньому, поки технологія розвиватиметься надзвичайно швидко, без сильної присутності в Інтернеті газети та журнали, безперечно, будуть боротися за існування. Найкращий спосіб – змінюватися, змінюватися з часом.

Розробивши веб-сайт новинного порталу, є можливість створити сильну присутність в Інтернеті, не витрачаючи занадто багато.

Загальна характеристика записки:

- кількість сторінок – 57;
- кількість рисунків – 21;
- кількість таблиць – 0;
- кількість посилань – 18

# 1 АНАЛІТИЧНА ЧАСТИНА

Є багато причини, чому онлайн чи цифровий перехід є важливим для газетної та журнальної індустрії або навіть для бізнесу. Навіщо для газети чи журналу потрібен онлайн-портал чи веб-сайт, в цей період інформаційно-технологічної революції це питання, яке не потребує відповіді. З кожним роком люди все рідше читають друковані газети чи журнали або отримують свіжі оновлення про події в усьому світі з програм, які вони завантажили на свій мобільний телефон, планшет чи ПК [1].

## 1.1 Опис предметної області

Зараз дуже сприятливий час для компаній, що займаються розробкою порталів новин. За останні роки спостерігається значне зростання онлайн-читання новин та інформаційного вмісту. Згідно з національним опитуванням читачів (Великобританія), у другому кварталі 2021 року «The Independent» мав загальну аудиторію 21,1 мільйона кожного місяця, що на 46% більше, ніж у другому кварталі 2020 року.

Дизайнери та розробники інтернет-порталів новин та веб-сайтів добре проведуть час зараз і в майбутньому. Згідно з опитуванням «World Press Trends» 2020 року, щонайменше 40 відсотків глобальних користувачів Інтернету читають газети в Інтернеті, а в більшості розвинених країн кількість читачів на цифрових платформах перевищила читацьку аудиторію друкованих. Газети та журнали, навіть підприємства, стукають у ваші розробників щодо допомоги.

Сектор розробки та дизайну новинних порталів в Європі має всі можливості повірити в те, що в майбутньому цей сектор отримає все більше і більше проектів. Факти та цифри зростання онлайн-читачів в Європі вказують на цю тенденцію. Нещодавнє опитування показало, що більшість із майже 50 мільйонів користувачів смартфонів в Європі проводять більшу частину свого

часу (72%) за серфінгом в мережі на своїх мобільних телефонах. Новини та розваги – це дві категорії, які шукають більшість споживачів [1].

Розробка веб-сайту порталу новин є важливою складовою успіху індустрії новин у сучасному світі високошвидкісного технологічного зростання з багатьох причин.

1. Майбутнє в Інтернеті.

Дизайн онлайн-порталу новин є важливим, оскільки майбутнє належить онлайн, а не друку. Стрімке зростання використання мобільних пристроїв для отримання новин та інформації протягом останніх років постійно збільшувало цифрову аудиторію газети. Згідно з дослідженнями та опитуваннями, у майбутньому ця тенденція буде посилюватися, і жодних ознак зниження не спостерігається.

2. Онлайн-портали швидкі.

У наш час люди хочуть знати речі негайно. Вони вже не готові чекати завтрашнього дня. На основі дослідження, опублікованого в Університеті Стірлінга, головною причиною поширення новин в Інтернеті стала безпосередність. На відміну від друкованих версій, онлайн-портали швидкі, актуальні та з оновленнями 24/7.

3. Люди хочуть налаштувати, тобто дизайн порталу новин надає читачам можливість налаштувати вміст і категорії відповідно до своїх інтересів і смаку. Друковані ЗМІ не дають користувачам такої свободи чи контролю. Ця свобода та контроль впливає на рівень прихильності читачів до порталу.

4. Простіший доступ – до онлайн-порталів або веб-сайтів легко отримати доступ з будь-якої точки світу. Читачу більше не доведеться хвилюватися, думаючи про те, що відбувається у його регіоні, поки вів перебуваєте за кордоном. Онлайн-портали новин миттєво дають вам оновлення. Ця функція легкого доступу спокушає читачів уникати друкованих ЗМІ.

5. Друк не безкоштовний.

Люди почали думати, що немає сенсу купувати друковану газету чи журнал, якщо вони вірять, що можуть отримати всі необхідні новини в Інтернеті безкоштовно. І це головна причина зниження тиражів друкованих ЗМІ у всьому світі.

6. Онлайн-портали мають більший вибір.

Це ще одна особливість дизайну онлайн-порталів, які приваблюють більше читачів, ніж друковані ЗМІ. Читачам пропонується широкий вибір категорій із спорту, культури, політики, розваг, стилю життя, освіти, здоров'я, подорожей тощо, які вони не отримують у друкованих версіях.

7. Онлайн у прямому ефірі

Прямі трансляції подій, виборів, спорту тощо перетягують читачів із друкованих ЗМІ на онлайн-портали. Дні, коли люди чекали наступного ранку, щоб дізнатися результати своїх улюблених матчів, минули [1].

### 1.2 Огляд аналогів та вимоги до проекту

Газети стали цифровими, оскільки кожна з них надає послуги електронного паперу та веб-сайтів новин. Цифрові медіа стали невід'ємною частиною сучасного суспільства та потенційним викликом для друкованих ЗМІ. Оскільки кількість людей, які користуються Інтернетом, збільшується з кожним днем, у найближчі роки ця проблема буде зростати.

Дизайн і розробка новинного порталу потребують більш ретельного продумування та планування в цю епоху цифрової новинної революції. Але тут у вас є варіант; тільки якщо ви хочете існувати і залишатися сильними на ринку онлайн-новин. Без вражаючого, захоплюючого, креативного, зручного та насиченого вмістом дизайну веб-сайт новинного порталу може зникнути найближчим часом.

Новини – це те, що завжди цікавило і цікавитиме людей, і не дивно, що новинні портали ломляться від відвідувачів, незважаючи на кризи та пандемії.

Будь-які новини завжди привертають увагу. Зазвичай, рейтинг формується за популярністю порталів, кількістю передплатників у соціальних мережах, трафіку головного сайту (статистика від SimilarWeb), а також умовною репутацією ЗМІ у читачів та за даними від незалежних джерел [2].

Для початку слід розглянути декілька аналогів для виявлення недоліків та від'ємностей. Першим інформаційним сайтом проаналізовано «CNN News» (рис. 1). CNN – найвідоміше інформаційне агентство світу, надає міжнародні новини, статті та аналітичні матеріали з Африки, Азіатсько-Тихоокеанського регіону, Європи, Латинської Америки, Близького Сходу, Південної Азії, США та Канади.

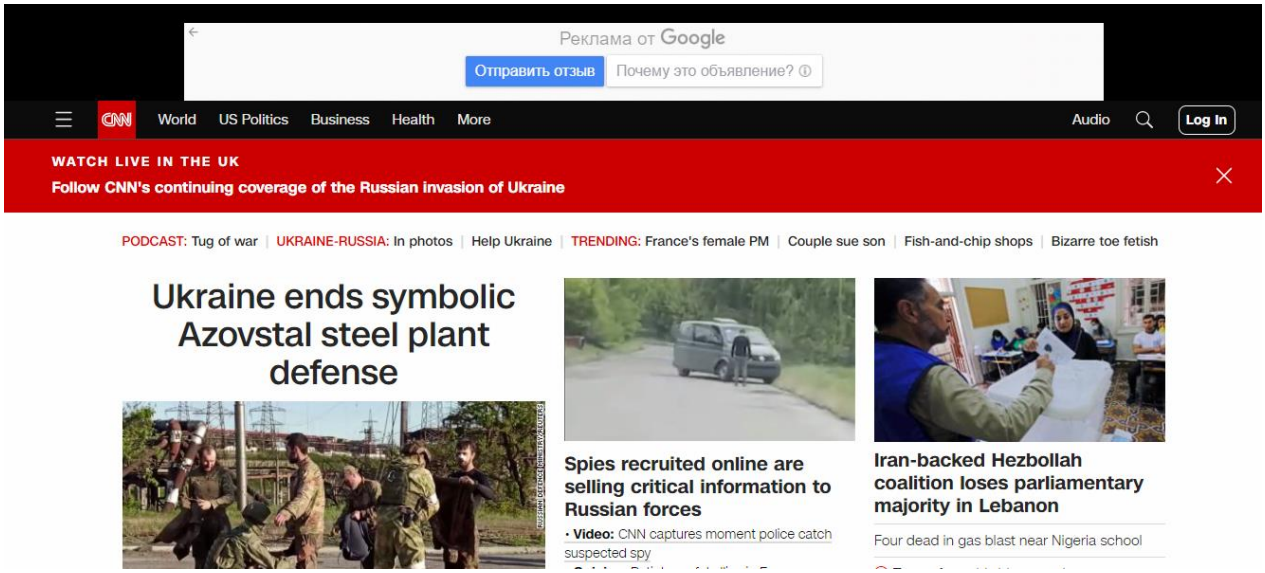



Рисунок 1 – Головна сторінка «CNN News»

Це один з найпопулярніших інформаційних платформ онлайн простору. Підтримка більшої кількості мов публікацій та охоплення найважніших новин робить цей новосний портал лідером у топі аналогів. Як і все інші лідери інтернет простору «CNN News» змушений підпорядковуватися Google, оскільки переважна більшість користувачів вперше відвідають його сторінку через відому пошукову систему [3].

В його механізм включені найважливіші аспекти:

- оригінальний контент;
- час публікації;
- періодичність публікації;
- слідуючи новітнім тенденціям;
- правильне форматування текстів (заголовки, ліди, середні заголовки, ключові слова та фрази).

Звичайно, головна мета – зробити кожну вашу публікацію корисною для справжніх користувачів, а не для роботів.

Другим аналогом було розглянуто портал новин «The Cipher Brief» (рис.2). Оскільки глобальний ландшафт безпеки ставав дедалі складнішим, «The Cipher Brief» став незамінним – забезпечуючи безпартійну платформу для експертів з уряду та бізнесу, щоб обмінюватися думками, навчатися один у одного та працювати разом, щоб знайти нових партнерів та рішень [2].

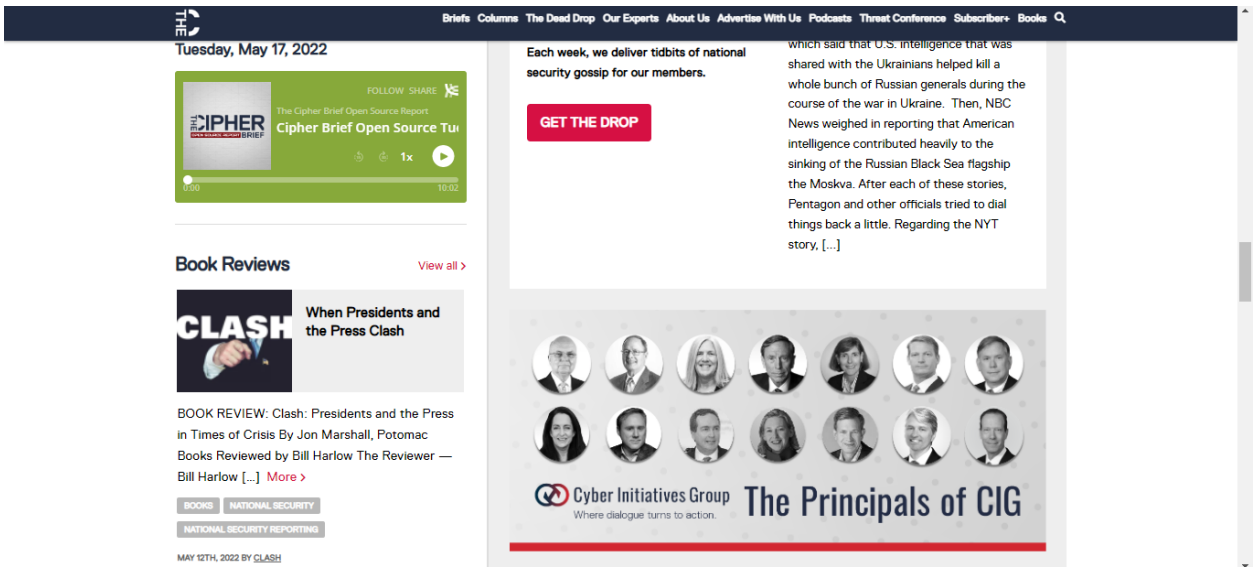



Рисунок 2 – Головна сторінка порталу «Cipher Brief»

«Cipher Brief» – це єдине ЗМІ, яке зосереджено виключно на тому, щоб підвищити рівень довіри щодо подій національної безпеки за допомогою новин, аналізу та інтерв'ю з керівниками уряду та приватного сектору. Там, де

інші інформаційні організації повідомляють читачам що сталося, «The Cipher Brief» розповідає, що це означає, співпрацюючи з мережею унікально кваліфікованих експертів, які допомагають надати контекст подій, які впливають на глобальну безпеку [4].

Останні роки суттєво вплинули на ринок онлайн-медіа. У березні-квітні відвідування сайтів зросло в середньому на 30-40%, а для окремих сайтів – аж удвічі. Такі "гірки" призвели до того, що за вісім місяців 2020-го змінилося три лідери рейтингу відвідування українських сайтів. Такими є результати дослідження Інституту масової інформації стосовно топ-50 українських інформаційних онлайн-медіа за рівнем відвідуваності (рис. 3).

| Місце | Сайт | Переглядів, млн |
|---|---|---|
| 1. | Pravda.com.ua | 39,88 |
| 2. | Strana.ua | 28,44 |
| 3. | 24tv.ua | 26,27 |
| 4. | Korrespondent.net | 26,19 |
| 5. | Rbc.ua | 25,30 |
| 6. | Obozrevatel.com | 24,05 |

Рисунок 3 – Статистика з українських порталів новин

Експерти IMI проаналізували дані сайту SimilarWeb. Цей сайт, своєю чергою, аналізує трафік понад 80 млн сайтів з усього світу й у відкритому доступі надає основну статистику цього трафіку. У цьому матеріалі наводяться результати дослідження трафіку 50 найвідвідуваніших українських онлайн-медіа [5].

Проаналізувавши українськи топові аналоги, можна відмітити наявність трафіку з пошуку для сайтів, який має платну основу. Тобто сайти для залучення трафіку з пошукових систем оплачують ті чи інші ключові слова, за

якими результати пошуку ведуть на цей сайт. Крім цього, всі топові сайти новин обов’язково мають посилання на youtube-репортажі, що теж збільшує кількість кліків на їх сторінки.

Після аналізу аналогів інформаційних сайтів новин було вирішено направити свій проект на новини місцевого характеру, так як крім соціальних мереж дуже мало інформації зібрано на одному інтернет просторі. Для наповнення майбутнього сайту інформацією слід ураховувати:

- промислові тенденції регіону;
- перевірити, чого не вистачає місцевій громаді;
- які теми найбільше обговорюються місцевими жителями в мережі (Facebook, дискусійні форуми, коментарі на інших сайтах порталів).

Ідеальний випадок коли у місті чи районі ще немає інформаційного порталу, це чудова можливість запустити його та задовольнити очікування та потреби місцевої громади.

# 2 ВИБІР ПРОГРАМНИХ ЗАСОБІВ РОЗРОБКИ

## 2.1 Аналіз фреймворків

У веб-розробці завжди розрізняється фронтенд і бекенд. Інтерфейс – це частина веб-сайту, яку користувачі бачать у своєму браузері. Бекенд, відповідно, знаходиться за кадром і зазвичай відповідає за бізнес-логіку веб-додатка. Інтерфейс складається з трьох дуже важливих елементів: HTML (створює структуру веб-сторінки), CSS (формує зовнішній вигляд елементів конструкції) та JavaScript – мова програмування, яка описує функціональність і відповідає за динамічні елементи на веб-сторінці.

Технічно всі три елементи інтерфейсу є лише рядками коду, які потім відтворюються браузером для відображення красивої веб-сторінки. Розробники витрачають багато годин на написання коду, щоб веб-сайт виглядав чудово. Однак є помічники, які можуть зменшити кількість необхідного коду та оптимізувати процес розробки в цілому. Це призводить до скорочення часу розробки та витрат на виведення продукту на ринок. Ці помічники називаються фреймворками.

Фреймворк програмного забезпечення (будь-то інтерфейс або бекенд) включає стандартизований попередньо написаний код, що робить розробку певної функціональності легшою та швидшою [6].

Можна написати код інтерфейсу будь-якою мовою програмування. Однак цей код потрібно конвертувати в JavaScript, оскільки браузери зараз можуть виконувати лише JS. Впровадження WebAssembly, однієї з основних тенденцій веб-розробки, незабаром може змінити цю ситуацію. Однак JavaScript все ще є оптимальним вибором для розробки інтерфейсу.

Вибір правильної технології для розробки додатків – це мудре рішення. JavaScript, оскільки сьогодні він забезпечує найширший функціонал для створення фронт-енда. Але дилема виникає при виборі середи розробки, а саме кращих фреймворків JavaScript, адже кожен наділений унікальними

особливостями. Як правило, спор будується навколо трьох фреймворків – Angular.js, React.js і Vue.js.

Facebook запустив React як бібліотеку JavaScript. Angular є фреймворком JavaScript на основі TypeScript, розроблений Google. Vue – це керований спільнотою фреймворк з відкритим кодом і платформа JavaScript, яка швидко розвивається.

### 2.1.1 Angular

Вперше фреймворк AngularJS був випущений Google у 2010 році. У 2016 році з’явився Angular 2, який був повним переписанням Angular JS. З тих пір нові версії часто з'являлися. На даний момент Angular 7 є останньою доступною версією.

Angular – це фреймворк MVW (Model-View-Whatever), який традиційно використовується як MVC (Model-View-Controller). Завдяки цьому додаток поділено на три взаємопов’язані компоненти. Це дозволяє розробникам Angular JS писати добре структурований код, що корисно для складних проектів.

Angular шаблони для створення компонентів є читабельними, оскільки вони здебільшого використовують стандартні теги HTML. Проста реалізація двостороннього зв'язування даних [7].

Двостороннє прив’язування даних означає, що будь-які зміни моделі впливають на подання. Навпаки, коли змінюється вигляд, одразу змінюється і модель. Angular дозволяє просте двостороннє прив’язування даних, що корисно для простих програм. Більш складні програми працюють швидше за допомогою одностороннього зв’язування даних, яке працює лише в одному напрямку (від перегляду до моделі чи від моделі до перегляду), залежно від потреб програмного забезпечення. Це дозволяє економити ресурси.

Angular має велику спільноту, яка розвивалася з моменту випуску AngularJS і стала сильнішою після випуску Angular 2. Фреймворк має близько

500 000 щотижневих завантажень на npm і понад 45 тисяч зірок на Github. Така популярність означає, що існує ряд рішень, сумісні з різними версіями Angular, а також можливість отримувати поради від досвідчених розробників і користувачів, не звертаючись до офіційної підтримки.

Недоліки Angular:

1. Не використовує Shadow DOM за замовчуванням, який вирішує проблему унікальних імен для елементів або ідентифікаторів сторінки, що може викликати у розробників серйозні проблеми, особливо коли мова йде про складні проекти.

Розробник може змінити стиль CSS для одного файлу, а інші файли також вплинуть на ті самі зміни. Shadow DOM дозволяє браузерам включати піддерево елементів DOM у візуалізацію документа, але не в основний документ DOM. Shadow DOM інкапсулює стилі, сценарії та вміст всередині спеціального елемента, щоб вони не заважали іншому вмісту програми.

2. Взагалі не використовує віртуальний DOM.

Віртуальний DOM дозволяє швидко змінювати будь-який елемент без необхідності відображати всю DOM. Цей підхід є тенденцією серед сучасних фреймворків веб-розробки, але, на жаль, він поки не підтримується жодною версією Angular. Натомість Angular 2 використовує односпрямований потік даних для виявлення змін у моделі та оновлює лише ті частини DOM, на які вплинули зміни моделі.

3. Використання TypeScript.

Будучи фреймворком JS, Angular підтримує використання чистого JavaScript. Однак ця структура була створена для використання з TypeScript, наднабором JavaScript, створеним Microsoft. TypeScript дозволяє розкрити справжню потужність Angular. Є один недолік – розробнику доведеться витратити деякий час на вивчення зміненого синтаксису.

4. Нижча швидкість візуалізації.

Обмежене використання тіньової DOM і відсутність віртуальної DOM призводить до зниження продуктивності. Швидкість відтворення/повторного

візуалізації перегляду нижча, ніж продуктивність інших фреймворків JavaScript на нашій діаграмі.

Angular – це монолітний фреймворк із важким кодом, який потрібно завантажити з сервера, перш ніж ви побачите веб-програму у своєму браузері. В результаті знижується швидкість і продуктивність [7].

### 2.1.2 Vue

Vue.js – популярна альтернатива Angular і React. Цей прогресивний фреймворк для створення інтерфейсу користувача набуває все більшої популярності. Переваги Vue:

1. Фреймворк MVC.

Так само, як і Angular, Vue є фреймворком MVC. Перевага цього очевидна – це дозволяє писати добре структурований код, що надзвичайно важливо при розробці складних додатків.

2. Легке рішення.

Однією з істотних переваг Vue є невеликий розмір фреймворка, оскільки він не включає багато функцій «з коробки», але функціональність легко розширюється за допомогою різноманітних рішень сторонніх розробників. Його часто порівнюють з Angular, який є монолітною структурою, яка має купу вбудованих функцій, які навряд чи будуть використовуватися у додатку взагалі. Звичайно, тремтіння дерева дозволяє усунути невикористаний код, але розмір фреймворка все ще більший у порівнянні з тим, що пропонує Vue.

Повнофункціональний проект Vue.js з Vuex і vue-router важить близько 30 КБ у форматі gzip. У той же час готова, скомпільована AOT програма, створена за допомогою angular-cli, має розмір ~130 Кб gzipped. Його компактний розмір і можливість включати сторонні модулі для розширення функціональності роблять Vue.js розумнішим вибором для тих, хто піклується про зменшення розміру, а отже, і про підвищення швидкості веб-програми.

3. Декларативні шаблони написані на HTML, що робить їх читабельними без знання інших мов програмування.

4. Віртуальний DOM.

Завдяки легшій реалізації віртуальної DOM програми, створені за допомогою Vue.js, мають найвищу продуктивність порівняно з іншими фреймворками інтерфейсу.

5. Двостороння прив'язка даних – Vue.js автоматично синхронізує всю модель з DOM.

6. Підтримується Laravel «з коробки».

Laravel є одним з найкращих серверних фреймворків PHP. Підтримка Vue дозволяє створювати веб-програми з використанням двох найпрогресивніших технологій без будь-яких додаткових інсталяцій.

7. Чистий JavaScript.

Vue.js використовує чистий JavaScript, усуваючи потребу розробникам або інженерам-тестувальникам вивчати будь-яку іншу мову програмування.

Завдяки приблизно 800 000 завантажень на тиждень Vue.js дає чудові результати. Понад 128 тисяч зірок Github демонструють його зростаючу популярність і також відображають погляди спільноти. Vue.js зараз використовується в усьому світі. Це означає, що рішення, сумісні з цим фреймворком JS, будуть ставати все більш доступними.

У порівнянні з Angular і React, Vue.js є найпростішою технологією інтерфейсу для вивчення та початку роботи. Крім того, ви можете почати додавати Vue.js до свого проекту крок за кроком [8].

Недоліки Vue: він менш популярний у порівнянні з Angular і React, обидва з яких вражають кількістю користувачів. Однак ви все ще можете розраховувати на підтримку інших членів спільноти.

Менше бібліотек для Vue.js оскільки є менше користувачів, було зроблено менше рішень для розширення функціональних можливостей фреймворку.

Цей фреймворк створив Еван Ю, колишній розробник Angular. Завдяки цьому він зрозумів слабкі сторони Angular і зміг створити альтернативу, яка б вирішувала ці проблеми, зберігаючи всі переваги Angular. Ця нова структура швидко стала надзвичайно популярною в його рідній країні Китаї. У результаті документація для деяких сторонніх бібліотек може бути доступна лише китайською мовою. Але офіційна документація повністю доступна англійською мовою [9].

### 2.1.3 React

React насправді є бібліотекою JavaScript, створеною для створення інтерфейсів користувача. Вона підтримується Facebook і Instagram і стала основною технологією для нескінченної стрічки в цих двох програмах. Як бібліотека JS, React має обмежену сферу використання, але в комплекті з іншими бібліотеками він стає потужним рішенням, що робить його головним конкурентом Angular [10].

Переваги React:

1. Компонентна модель.

React не використовує жодних шаблонів. Логіка компонента написана на JavaScript, що надає їй більшу гнучкість і дозволяє легко проходити через вашу програму великі обсяги даних, зберігаючи при цьому стан DOM. Хоча цей підхід використовується у всіх порівнюваних інтерфейсних фреймворках, React був першим, хто представив модель компонентів.

2. Віртуальний DOM.

Віртуальна DOM дозволяє створити спрощену копію DOM. Усі зміни, які необхідно внести, вносяться у віртуальний DOM. Пізніше дві DOM порівнюються, і, коли будуть виявлені відмінності, справжня DOM повторно відтворить лише змінену частину. Цей процес набагато швидший і ефективніший, ніж робота безпосередньо з DOM.

3. Одностороннє прив’язування даних.

Двостороннє прив’язування даних було перевагою для Angular, і одностороннє прив’язування даних React також може бути перевагою. Цей підхід змушує подання реагувати на будь-які зміни, внесені в модель, але зміни в самому поданні не можуть вплинути на модель. В результаті дані надходять лише в одному напрямку, що зменшує можливість будь-яких побічних ефектів.

4. Використання чистих функцій.

На відміну від Angular, React не вимагає використання класів. Інтерфейс вашого додатка можна створити за допомогою чистих функцій, що спрощує кодову базу.

5. Вбудована платформа мобільної розробки.

React Native – це партнер React, фреймворк, створений для розробки нативних мобільних додатків за допомогою JavaScript. З його допомогою програми можуть включати в себе власні будівельні блоки і виглядати точно так само, як програми, створені за допомогою Objective-C або Java. Великою перевагою є те, що потрібен лише один розробник, щоб створити рідний додаток для двох платформ, оскільки для цього потрібно лише знання JavaScript.

У порівнянні з гібридними мобільними додатками, створеними за допомогою Angular, програми React Native мають набагато вищу продуктивність і майже таку ж потужну, як нативні програми.

За підтримки двох найбільших світових компаній соціальних мереж, React має мільйони користувачів по всьому світу. Ця сильна спільнота означає, що користувачі мають доступ до швидкої допомоги від досвідчених розробників.

На Github React зібрав понад 122 тисячі зірок, а статистика використання npm показує, що фреймворк завантажується більше 5 мільйонів разів на тиждень [9].

Недоліки React:

1. Потрібні сторонні технології.

React – це бібліотека JavaScript, а не фреймворк. В результаті його можливості обмежені. Для розширення функціональності необхідно використовувати сторонні модулі та бібліотеки. Читання документації та вивчення використання цих технологій може зайняти деякий час.

2. Використання JSX.

React рекомендує використовувати JSX замість звичайного JavaScript і HTML. Насправді це JavaScript, але його було розширено синтаксисом XML. Творці стверджують, що JSX швидше, безпечніше і легше, ніж JS. Отже, щоб отримати максимум від React, вам потрібно буде вивчити цю модифікацію JavaScript [11].

## 2.2 Платформа Node.js

Node.js – це серверна платформа, обгорнута мовою JavaScript для створення масштабованих програм, керованих подіями. Це бентежить навіть досвідчених програмістів, оскільки традиційне середовище JavaScript завжди було на стороні клієнта – у браузері користувача або в додатку, який спілкується з сервером. JavaScript не розглядався, коли мова йде про сервер, який відповідає на запити клієнта, але саме це надає Node.js.

Node.js написаний не на JavaScript (він написаний на C++), але він використовує мову JavaScript як інтерпретаційну мову для обробки запиту/ відповіді на стороні сервера. Іншими словами, Node.js запускає автономні програми JavaScript. Перевага полягає в тому, що програмісти можуть використовувати свої поточні, хоч і на стороні клієнта, знання програмування і починати кодування з Node.js набагато легше.

Node.js має декілька атрибутів, які роблять його особливо привабливим для мережевого або Інтернет-програмування. Перша пов'язана з усіма

накладними витратами та упаковкою, які використовуються існуючими технологіями для розмови в Інтернеті.

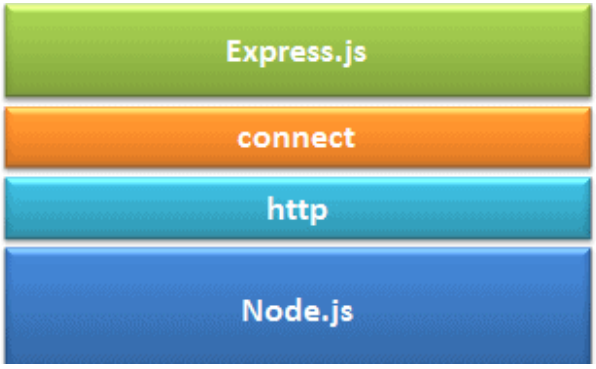
Другий привабливий атрибут Node.js пов'язаний з моделлю подій веб-програмування. Більшість існуючих технологій написані для отримання «великих прогалин» даних для кожного запиту та відповіді. Іншими словами, на сервер може бути надіслана ціла сторінка даних, навіть якщо є лише невеликі зміни. Ці технології оптимізовані для використання великих фрагментів даних із меншою кількістю подій. Node.js робить навпаки; він розроблений для роботи з більшою інтерактивністю - меншими шматками даних, що реагують на багато інших подій [12].

### 2.3 MongoDB та фреймворк Express

Express – це мінімальний і гнучкий фреймворк веб-додатків Node.js, який надає надійний набір функцій для розробки веб- та мобільних додатків. Це сприяє швидкому розвитку веб-додатків на основі Node. Нижче наведено деякі з основних функцій Express Framework:

- дозволяє налаштувати проміжне програмне забезпечення для відповіді на http-запити;
- визначає таблицю маршрутизації, яка використовується для виконання різних дій на основі методу http та url-адреси;
- дозволяє динамічно відображати сторінки html на основі передачі аргументів шаблонам [13].

Express.js заснований на модулі проміжного програмного забезпечення Node.js під назвою «connect» (рис. 4), який, у свою чергу, використовує модуль «http». Таким чином, будь-яке проміжне програмне забезпечення, засноване на підключенні, також працюватиме з Express.js.




Риуснок 4 – Ієрархія модулів для Express.js

Переваги Express.js:

- робить розробку веб-додатків node.js швидкою та легкою;
- легко налаштувати;
- дозволяє визначати маршрути вашої програми на основі методів http та url-адрес;
- включає різні модулі проміжного програмного забезпечення, які можна використовувати для виконання додаткових завдань на запит і відповідь;
- легко інтегрувати з різними механізмами шаблонів, такими як jade, vash, ejs тощо;
- дозволяє визначити проміжне програмне забезпечення обробки помилок;
- легко обслуговувати статичні файли та ресурси вашої програми.
- дозволяє створити сервер rest api;
- легко підключатися до таких баз даних, як mongodb, redis, mysql [14].

Node.js – це сервер, керований подіями, який має один потік, він в свою чергу керує всіма підключеннями до сервера. Він обробляє всі запити у зворотних викликах, не затримуючи їх потік (рис. 5).

Коли запит надходить на сервер, Express обробляє його у зворотний виклик, не блокуючи основний стек.

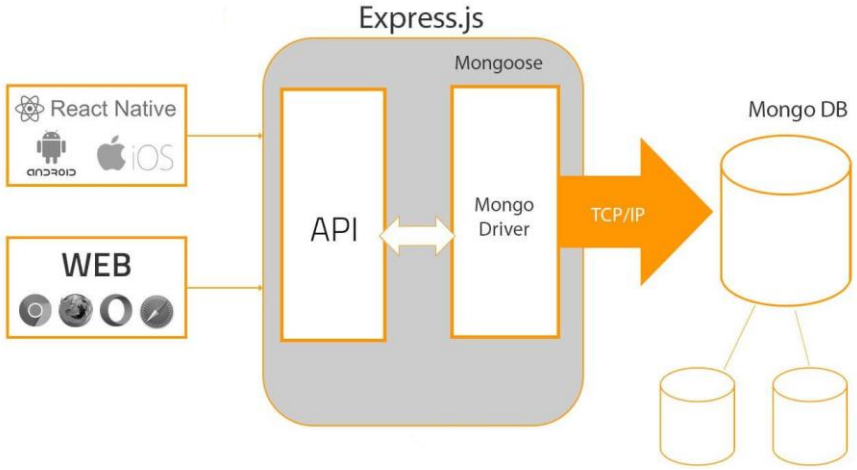



Рисунок 5 – Робота Express.js

Якщо потрібно отримати деякі дані з файлу або бази даних, то такий процес може зайняти багато часу (наприклад, через затримку читання диска). Середній сервер блокуватиме всі інші запити, доки ці дані не будуть доставлені.

Вузол JS, у свою чергу, реєструє логіку постобробки даних у зворотному виклику. Дані обробляються паралельно, тому сервер може приймати інші дані одночасно. В результаті Node.js може обробляти більшу кількість запитів ефективніше.

Mongoose – це надбудова драйвера Mongo, яка ідеально підходить до екосистеми Express. Головною перевагою MongoDB є те, що він орієнтований на JavaScript. Це означає, що драйвер передає всі дані, які він має, не витрачаючи часу на пристосування їх до середовища програмування. MongoDB розроблено для масштабованих серверів, керованих даними [14].

Кожен запит у MongoDB обробляється так само, як і у зворотному виклику. Це означає, що сервер не повинен чекати (блокувати запас), поки певні дані не будуть передані, і може одночасно обробляти інші запити. Це надзвичайно важливо, оскільки масштабована база даних може бути розташована на одному або кількох фізично віддалених серверах, тому отримання даних з них скорочує час.

Коли завершається весь процес, сервер отримує запит і реєструє обробку даних події. Потім подія надсилає запит на віддалений сервер Mongo. Після цього потрібен деякий час для повернення даних. Одночасно сервер обробляє інші запити. Коли дані надходять, драйвер Mongo викликає обробник запитів користувача. Обробник запитів користувача повертає дані клієнту.

Express.js прискорює процес розробки, оскільки використовується лише одна мова (JavaScript). Використання MongoDB також відіграє важливу роль, оскільки він орієнтований на JavaScript. А ReactNative робить можливим розробку мобільних додатків [15].

# 3 ПРОЕКТНА ЧАСТИНА

## 3.1 Вимоги до проекту

Найкращий дизайн порталу новин в Інтернеті має масу відмінних рис і характеристик. Якщо необхідно, щоб дизайн новинного порталу залучав більше читачів і охоплював більшу аудиторію, наведені нижче 11 функцій допоможуть у цьому:

1. Індивідуальний дизайн порталу новин.

Розвиток онлайн-порталу новин має надати читачам можливість налаштувати свою сторінку надії, надавши їм можливість вибирати розділи, зміст, теми, які їх цікавлять. Якщо дизайн не запропонує цей персоналізований досвід читача, відвідувачі, швидше за все, залишать портал у пошуках інших, які пропонують їм це.

2. Останні новини.

Це ще одна важлива відмінна риса розробки сайту чудового новинного порталу. Миттєві оновлення останніх новин із новинами, які цікавлять людей, є достатньо потужними, щоб портал затягував читачів цілий день. Якщо портал в першу чергу приносить читачам найголовніші та хвилюючі новини, то вони, відвідають даний портал новин у пошуках чогось, що їх сколихне.

3. Стиль має значення.

Розробка новинного порталу – це також творчість. Створення порталу чи веб-сайту зі стильним та актуальним дизайном – це абсолютний процес мистецтва та творчості. Розробнику необхідно знати аудиторію і створювати для неї дизайн.

4. Легкий для навігації дизайн.

Дизайн новинного порталу, у якому легко орієнтуватися, швидше за все, відвідують читачі, ніж ті, у яких немає їх. Більшість новинних веб-сайтів розміщують своє основне меню навігації відразу під заголовком і над вмістом. Деякі інші також використовують ліву бічну панель. Обидві ці позиції легко побачити відвідувачам.

5. Колірна схема.

У розробці новинного порталу читабельність є найважливішою характеристикою. Більшість веб-сайтів новинних порталів використовують темний текст на білому тлі. Контраст кольорів полегшує читабельність вмісту, тому схема «чорне по білому» найбільш підходить для новинних порталів та веб-сайтів [16].

6. Адаптивний дизайн новинного порталу.

Розробники порталу новин повинні звернути увагу на цю відмінну рису; адаптивний дизайн. Завдяки адаптивному дизайну новинного сайту ваш портал добре виглядає на екрані будь-якого пристрою: ноутбуків, планшетів і мобільних телефонів.

7. Часті оновлення новин.

Часте оновлення вмісту є важливим для існування новинного порталу. Без частих оновлень ваш сайт не приверне аудиторію. Частий вміст також впливає на результати пошуку в пошукових системах.

8. Рекламні місця.

У розробці новинного порталу монетарні характеристики так само важливі, як і функції вмісту та сервісу. Інтеграція реклами в правильній частині дизайну має вирішальне значення для грошової вигоди порталу новин. Необхідно мати можливість використовувати рекламу, не відвертаючи читачів від веб-сайту.

9. Дизайн новинного порталу на основі сітки.

Веб-сайти новинних порталів, як правило, створюються на основі сітки. Цей дизайн є популярним вибором не тільки завдяки гострому та вражаючому вигляду, який він створює, але й є одним із найефективніших способів керування та організації великої кількості контенту. Дизайн на основі сітки дозволяє розбити вміст на вичерпні блоки; таким чином залучати користувачів до читання вашого вмісту.

10. Максимізація вільного простору.

React бібліотеку сам по мірі необхідності, або використовує збірну солянку з готових рішень, бережно відображуваних по кількості зірки в безбрежному океані npm-модулей. Ant Design – це повноцінна дизайн-система, візуальна мову зі своїми принципами, стайлгайдами і бібліотекою компонентів. Основні особливості: UI корпоративного класу, розроблений для веб-прикладів [17].

Якщо програми MVP-версії повинні бути готові за короткий час, то у таких випадках перед стартом проекту роблять вибір у використанні готових компонентів бібліотеки, щоб мінімізувати витрати часу на опис власних велосипедів [18].

### 3.2 Структура проекту

З урахуванням того, що проект представляє собою MVP-версію, він буде мати спрощену структуру: головну сторінку та розділ для адміністрування (рис. 6).

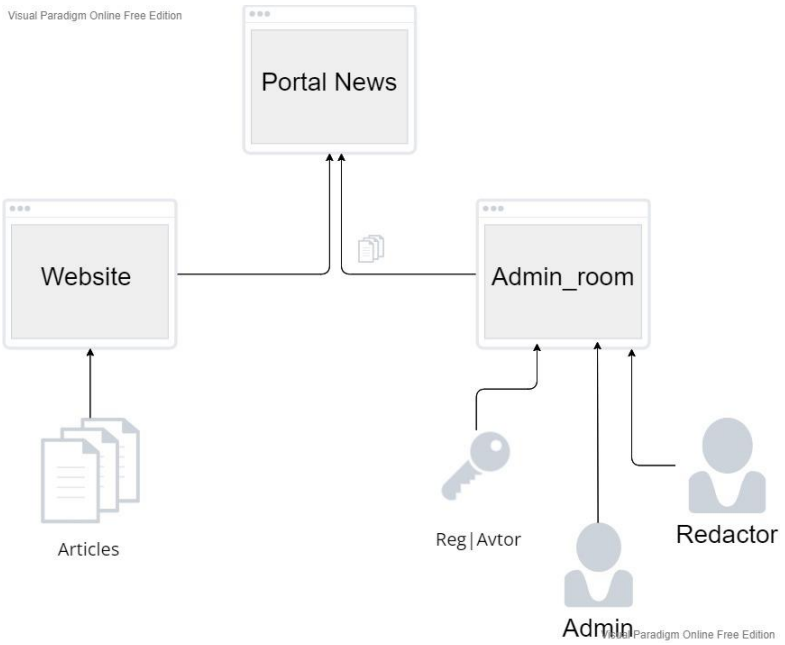



Рисунок 6 – Структура проекту

Опишемо більш детально кожний елемент для подальшого проектування UI. Головна сторінка буде включати стрічку новин та блок для перегляду обраної статті.

Розділ «Стрічка новин»:

1. Блок «Хедер»:

– зображення «логотип сайту»;
– текст «названа сайту»;
– текст «короткий опис сайту»;
– Background.

2. Блок «Список новин» складається зі списку блоків «Короткий опис статті» з посторінковим виводом. Відображає статті із значенням поля «Статус» – «Опубліковано».
3. Блок «Короткий опис статті»:

– зображення «логотип»;
– текст «назва» і короткий опис;
– дата публікації – заповнюється автоматично датою першої публікації.
– дія «Перехід до статті» здійснюється за натисканням на заголовок статті та відкриває розділ «Перегляд статті».

4. Блок «Футер» включає до себе посилання «Вхід для співробітників»(блок «Вхід до системи»).

До розділу «Перегляд статті» входить три блока:

1. Блок «Заголовок сайту».
2. Блок «Візуалізація статті», який відображає данні блоків відповідної статті у читабельному вигляді:

– загальна інформація по статті та секції статті.

3. Блок футера.

Для адмін розділу буде створено дві ролі: адміністратор та редактор. Адміністратор має доступ до:

– розділу «Список статей»;

– розділу «Налаштування сайту»;
– розділу «Користувачі».

Редактор працює у розділі «Список статей», де знаходяться всі функції керування контентом:

1. Блок «Загальна інформація по статті»:
– поле «Назва статті» є обов’язковим;
– поле «Короткий опис» є обов’язковим;
– поле «Логотип» відкриває провідник надаючи можливість вибрати файлу формату .jpg. Після вибору поряд з кнопкою з’являється назва файлу. Обов’язкове для заповнення.
2. Блок «Секції статті» містить вертикальний список згенерованих форм секцій. Кожна форма секції містить дію «Видалити». Між формами у роді розділювача виступає блок «Вставити форму». Блок «Вставити форму» також міститься на початку та на кінці форми.
3. Блок «Вставити форму»:
– поле «Тип елемента» є обов’язковим. В залежності вибору значення буде створений відповідний блок:
    a) значення «Зображення». У разі вибору буде створений блок «Форма заповнення даних елемента «Зображення»;
    b) значення «Цитата». У разі вибору буде створений блок «Форма заповнення даних елемента «Цитата»;
    c) значення «Абзац». У разі вибору буде створений блок «Форма заповнення даних елемента «Абзац»;
    d) значення «Заголовок». У разі вибору буде створений блок «Форма заповнення даних елемента «Заголовок».
– кнопка «Вставити» створює блок та вставляє його на місце відповідного блоку «Вставити форму»;

4. Блок «Форма заповнення даних елемента «Зображення»:
– поле «Зображення». Відкриває провідник надаючи можливість вибрати файлу формату .jpg. Після вибору поряд з кнопкою з’являється назва файлу. Обов’язкове для заповнення;
– поле «Посилання на першоджерело».
5. Блок «Форма заповнення даних елемента «Цитата»»
– поле «Текст». Обов’язкове;
– поле «Автор». Обов’язкове.
6. Блок «Форма заповнення даних елемента «Абзац» має поле «Текст». Обов’язкове.
7. Блок «Форма заповнення даних елемента «Заголовок» має поле «Тип заголовка». Обов’язкове. Є можливість обрати тип заголовку (B2..B4). Дія «Зберегти». Відправляє запит на збереження статті у БД та відкриває розділ «Список статей». Дія «Відміна» відкриває розділ «Список статей».

### 3.3 Діаграма варіантів використання

Діаграма варіантів використання UML є основною формою вимог до системи/програмного забезпечення для нової недостатньо розробленої програми. Випадки використання визначають очікувану поведінку (що), а не точний спосіб її здійснення (як).

Випадки використання після вказівки можуть бути позначені як текстовим, так і візуальним уявленням (тобто діаграма варіантів використання). Ключова концепція моделювання варіантів використання полягає в тому, що воно допомагає розробникам проектувати систему з точки зору кінцевого користувача. Це ефективна техніка для передачі поведінки системи в термінах користувача шляхом визначення всієї зовнішньої поведінки системи.

Побудуємо діаграму Use-Case для об’єкту розробки, яка наглядно демонструє ролі у системі та основні варіанти використання кожного актору (рис. 7).

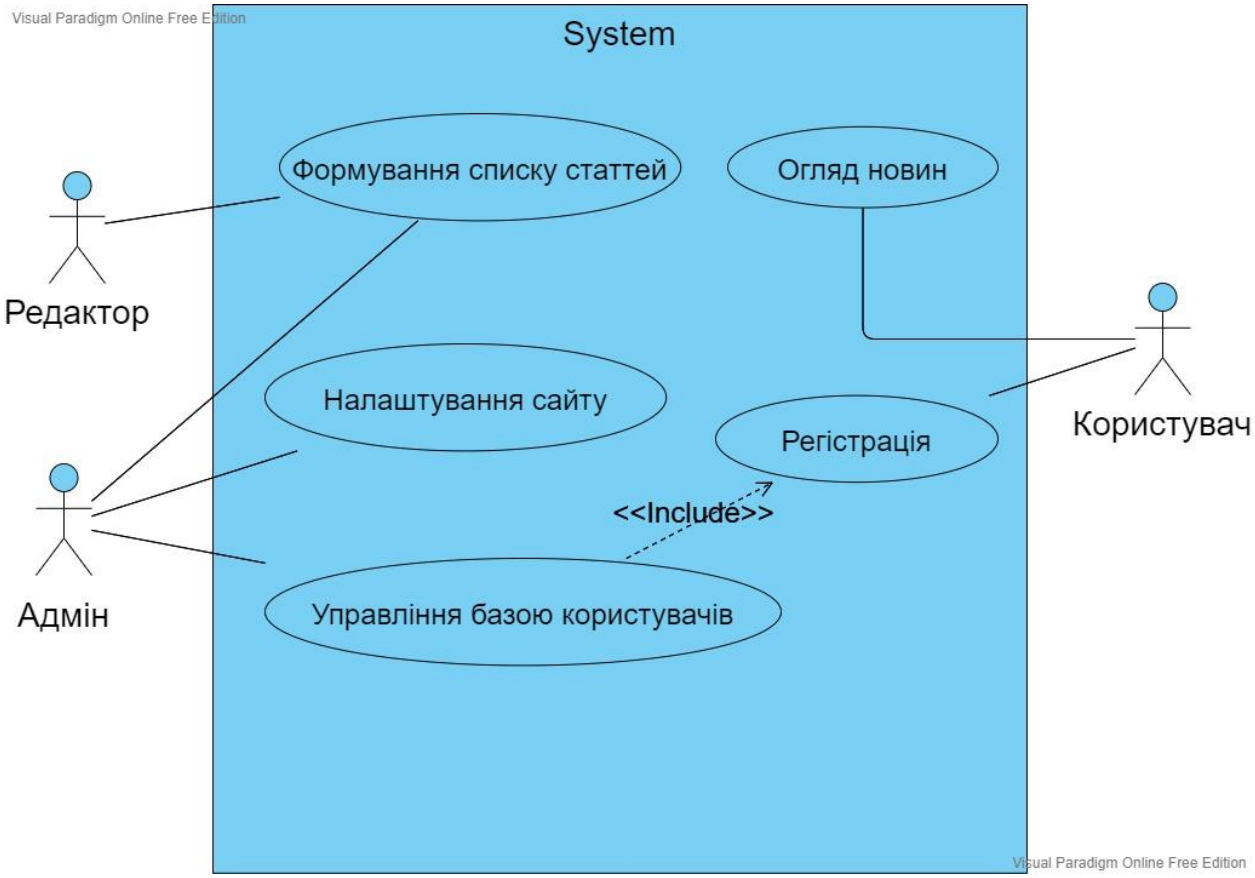



Рисунок 7 – Діаграма Use-Case

# 4 ОПИС РЕАЛІЗАЦІЇ ПРОЕКТУ

## 4.1 Інтерфейс програмного продукту

При проектуванні об’єкту розробки було прийнято рішення створити шаблон-каркас для завантаження новин враховуючи факт того, що навіть у південному регіоні нашої країни існують селищні ради, які потребують власного інформаційного веб-ресурсу. Структура та функціонал максимально спрощені задля збільшення попиту на програмний продукт.

Головна сторінка сайту на прикладі новин Одеси представлено на рисунку 8:

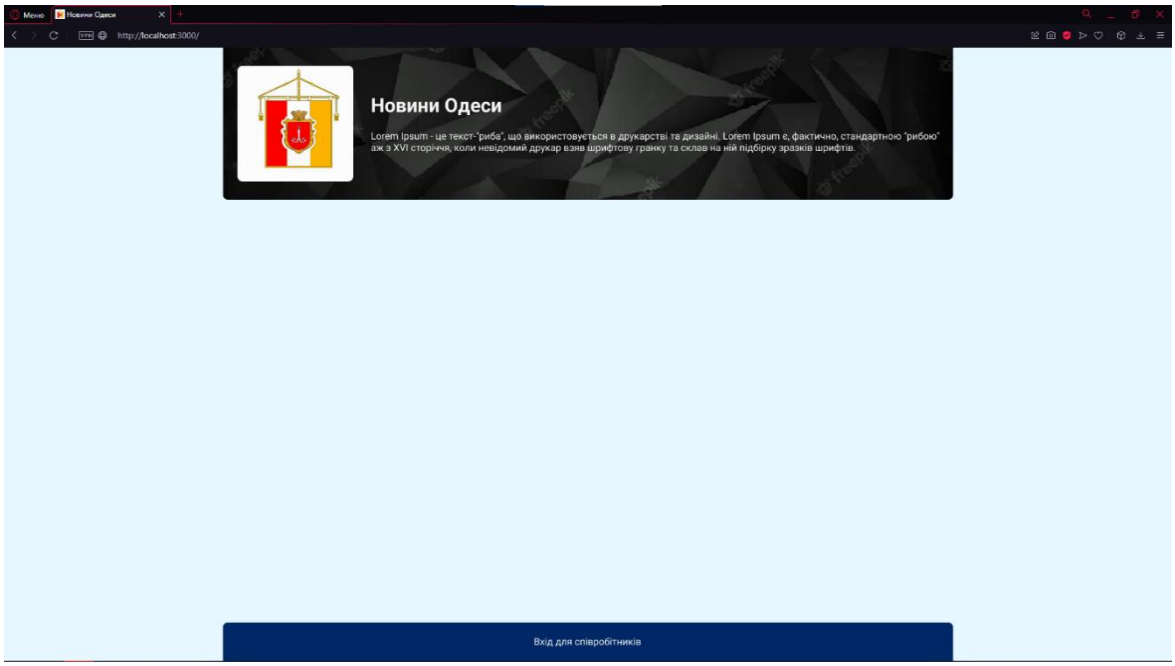



Рисунок 8 – Головна сторінка

За замовчуванням блок для розміщення новин пустий. Присутні тільки хедер, де розміщено назву та емблему (прапор) міста, та футер з посиланням на вхід до адмін-частини порталу. Для перегляду майбутньої інформації на сайті реєстрація не потрібна. Для співробітників найманих власником веб-ресурсу для забезпечення своєчасного написання та публікації останніх новин

порталу існує обов’язкова процедура створення нового облікового запису користувача.  Розглянемо цей процес більш детально.

Якщо співробітник порталу уперше хоче отримати доступ до адмін-панелі, необхідна реєстрація. Для цього слід перейти за посиланням та заповнити форму, яка представлена на рисунку 10:

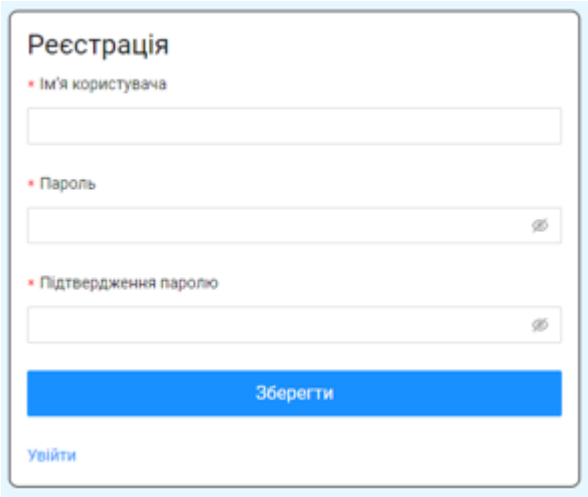



Рисунок 10 – Форма реєстрації користувача

Однофакторна аутентифікація (SFA) – метод автентифікації з використанням лише однієї категорії. Найбільш поширеним прикладом SFA є облікові дані, пов'язані із запровадженням імені користувача та звичайного пароля.

Це допоможе власнику сайту забезпечити безпеку інформаційного ресурсу від несанкціонованого доступу. Натомість адміністрація отримує можливість оперативно редагувати опубліковану на порталі інформацію.

Для поля з паролем наявні вимоги щодо мінімальної довжини пароля та його коректного повторного введення, це зроблено для забезпечення достатньої надійності пароля та впевненості у коректності його вводу. Якщо під час підтвердження паролю значення обох полів не співпадають, система виведе повідомлення як показано на рисунку 11.

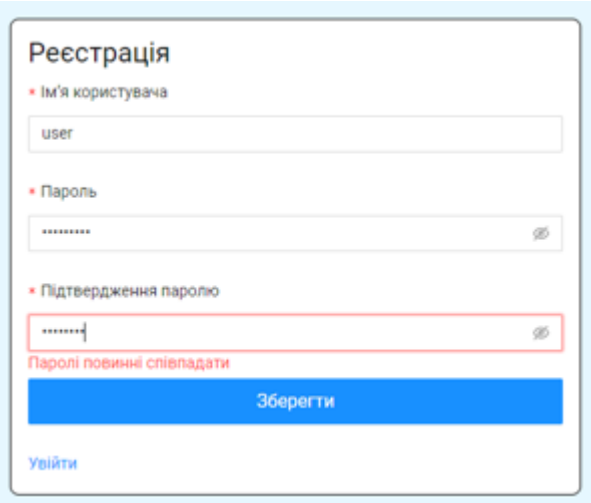



Рисунок 11 – Приклад помилкового вводу пароля

Крім цього, якщо користувач намагається зареєструватись під логіном, який вже існує у системі БД, відображається відповідне повідомлення (рис. 12).

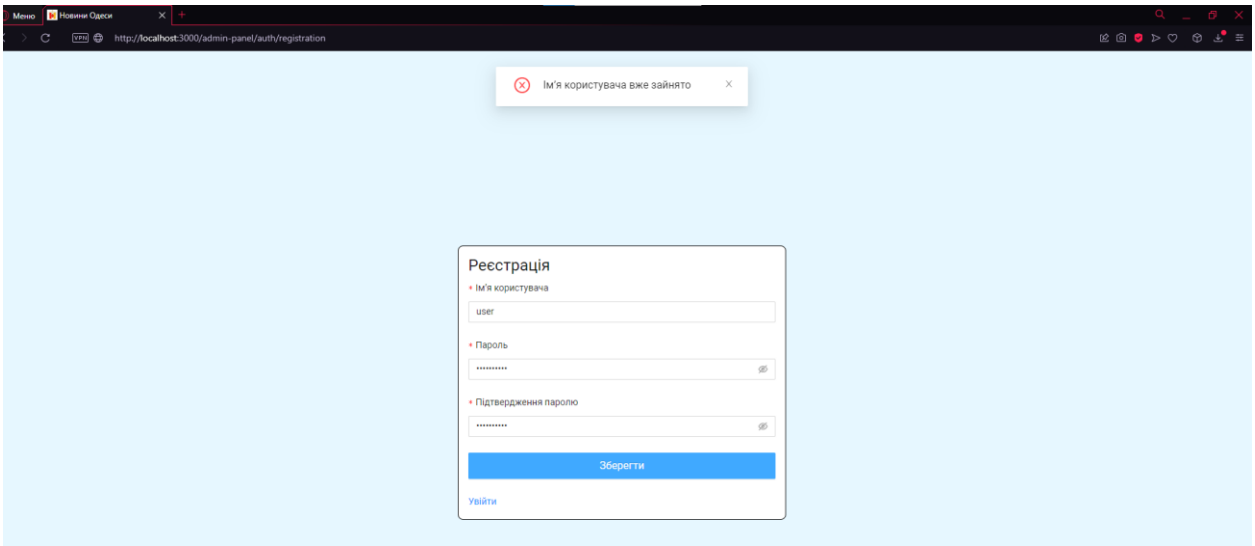



Рисунок 12 – Приклад перевірки існуючих даних у системі

Після успішного створення нового облікового запису співробітник повинен звернутися до людини виконуючої обов’язки адміністрування сайту

та передати їй використане під час реєстрації ім’я для подальшого активування аккаунту з вкладки «Користувачі» (рис. 13).

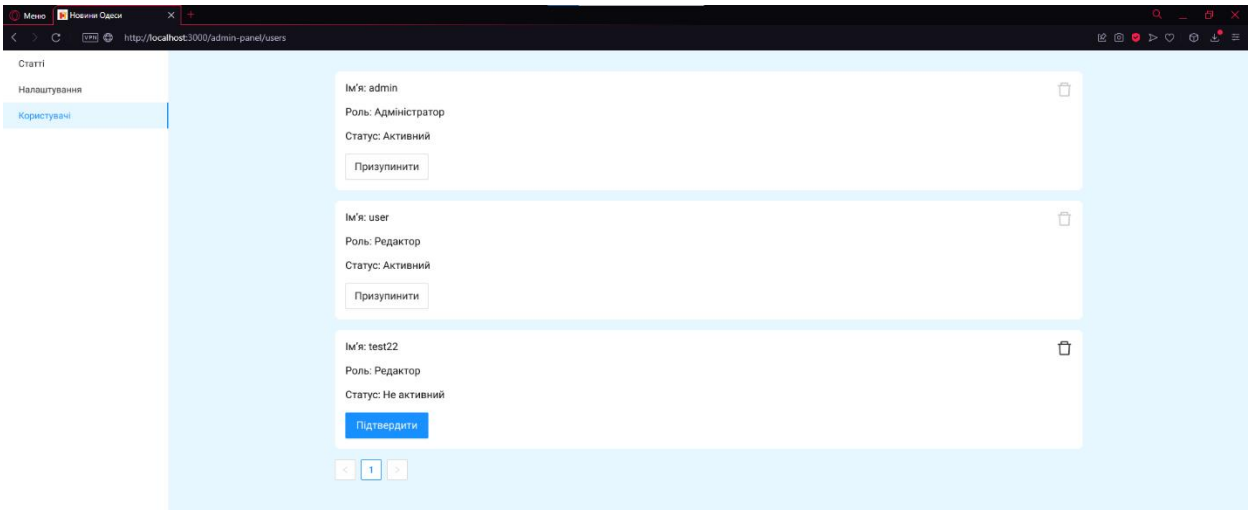



Рисунок 13 – Сторінка зі списком облікових записів

За аналогічним алгоритмом працює авторизація – процес, коли зареєстрований користувач хоче зайти у систему для отримання доступу до редагування даних порталу (рис. 14).

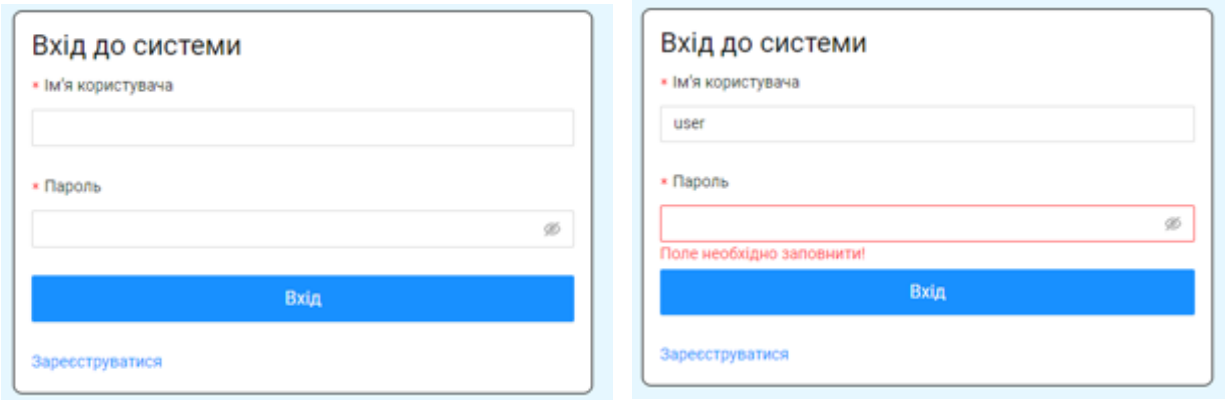



Рисунок 14 – Авторизація з перевіркою даних

Для зручності користувача для полів логін та пароль існує перевірка. У разі спроби авторизуватись під неіснуючими даними у системі користувачу

буде відображене повідомлення «Акаунт не знайдено» чи «Ім’я та пароль користувача не є дійсними» та пропозиція перейти до посилання на реєстрацію (рис. 15).

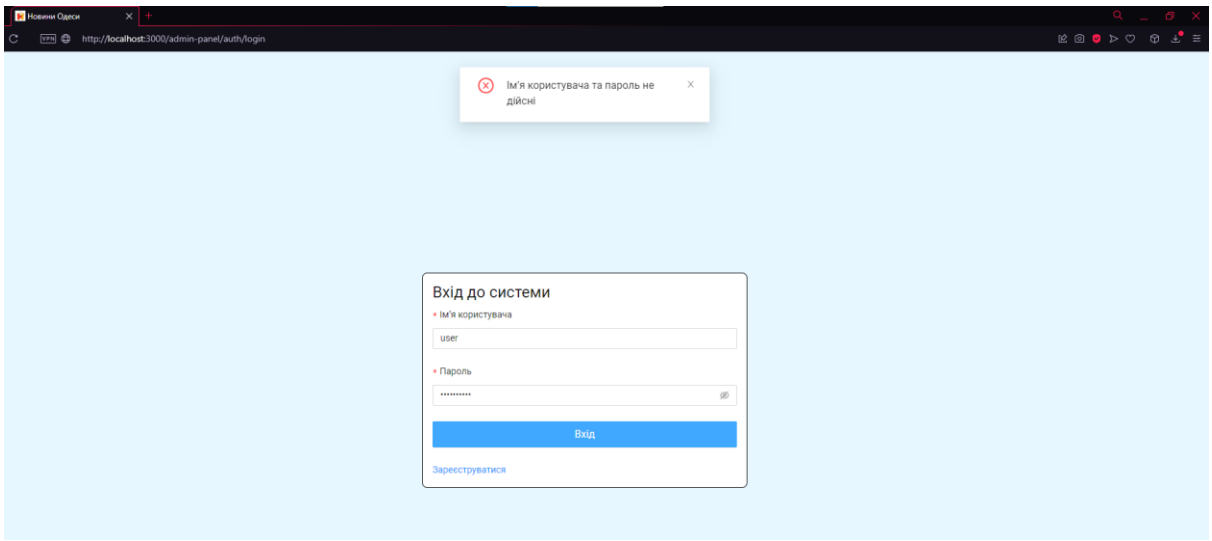



Рисунок 15 – Перевірка існування користувача

Для співробітників, які щойно створили собі нові облікові записи є одне обмеження: одразу після реєстрації аккаунт не є активним і авторизуватися використовуючи його не є можливим (рис. 16).

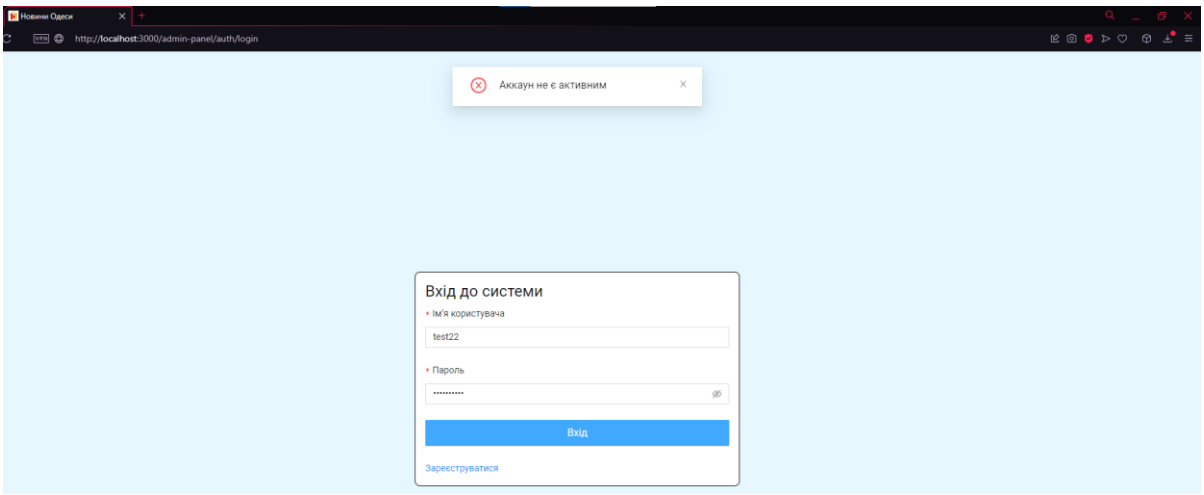



Рисунок 16 – Вхід до системи з неактивного акаунту

Далі розглянемо кабінет адміністратору (рис. 17).

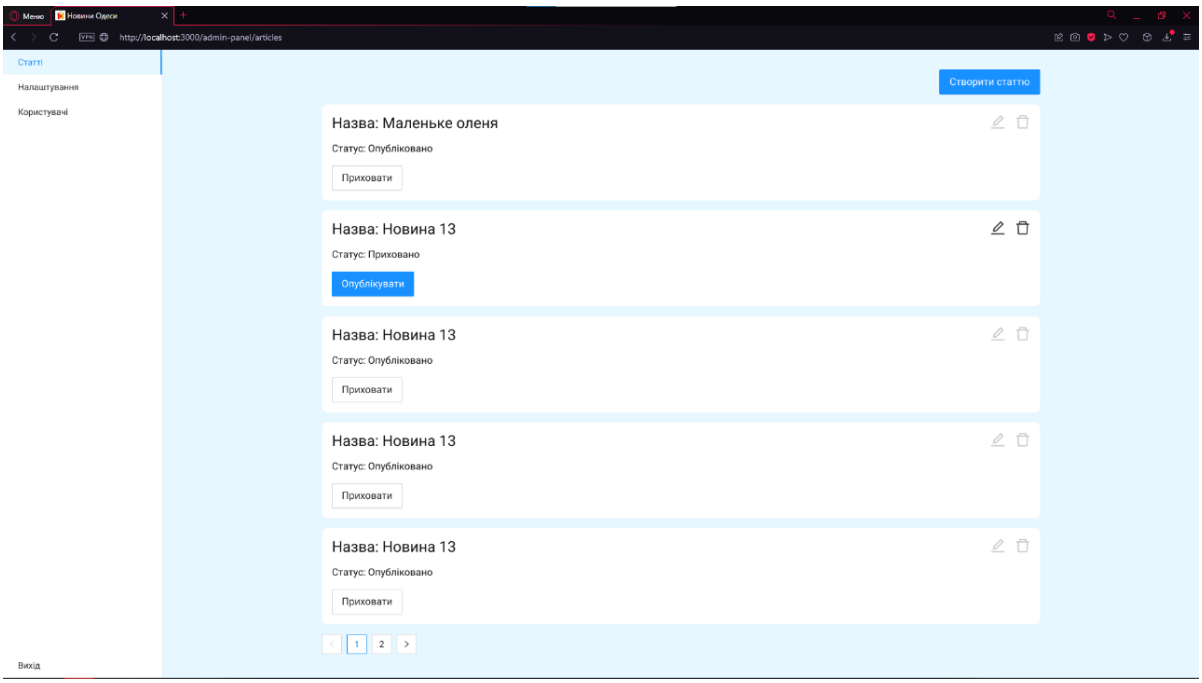



Рисунок 17 – Сторінка зі списком статей

За замовчуванням після входу у систему для користувача відображається список статей порталу. Крім цього, у його доступі для адміністратора є дві вкладки «Користувачі» та «Налаштування».

Перша вкладка необхідна для відстеження поступаючих запитів на активацію нових облікових записів для нових співробітників, та співробітників, що втратили доступ до своїх діючих облікових записів.

Також вона дозволяє деактивувати облікові записи, що більше не будуть використовуватись (у разі звільнення співробітника, зміни місця роботи, або втрати доступу).

Розглянемо процес створення статті (рис. 18). Як для адміністратора, так і для звичайного користувача для заповнення статті інформацією існує загальна форма:

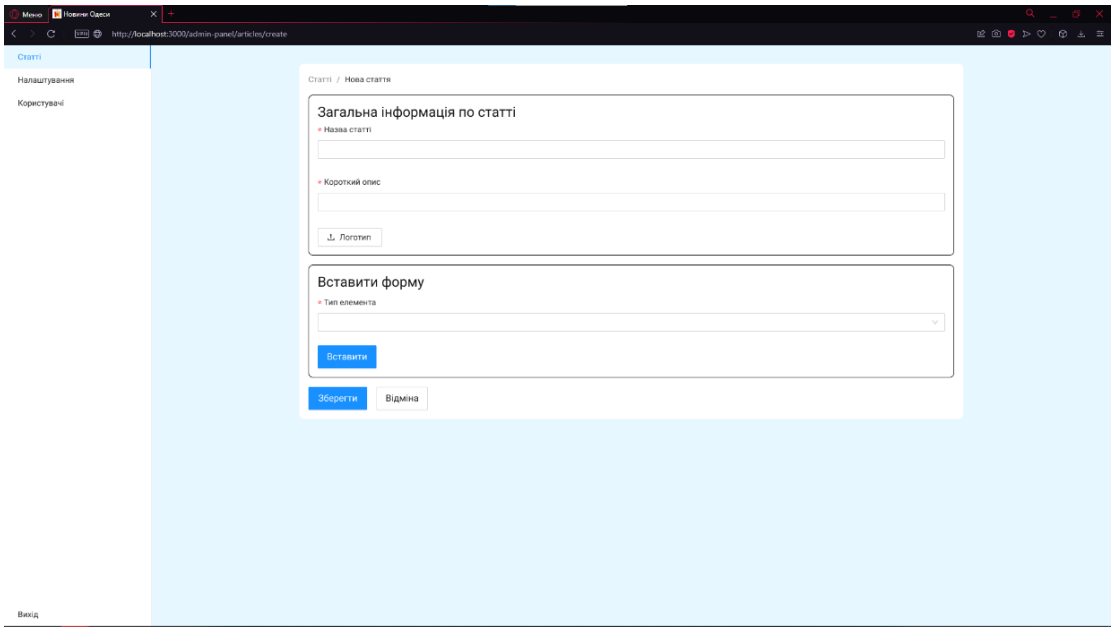



Рисунок 18 – Форма для нової статті

Крім базових елементів управління (назви та короткого опису) статті існують і додаткові поля з можливістю завантаження зображень. За необхідністю можна додати додаткові елементи до форми управління контентом (рис. 19).

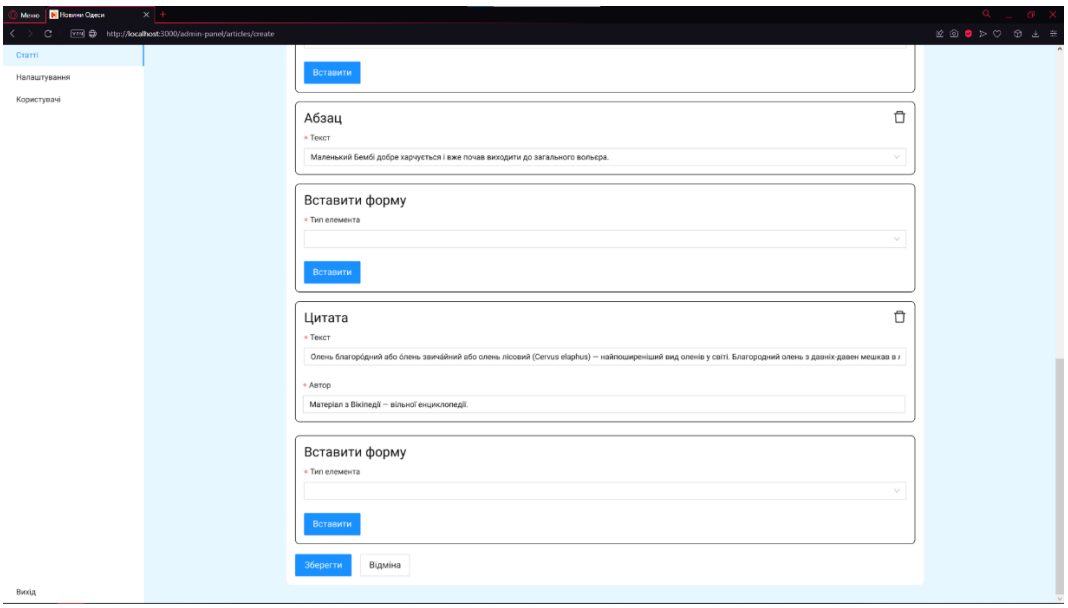



Рисунок 19 – Приклад додаткових елементів

Для кожної статті буде назначено статус «приховано» одразу після її збереження. Після збереження редактор може опублікувати статтю на сторінці зі списком статей. Редактор зможе змінювати її чи взагалі видалити статтю, але тільки у разі статусу «приховано». Аналогічно, якщо необхідно редагувати вже опубліковану статтю, її слід спочатку перевести у статус «приховано».

Для адміністратору відкриті функції з управління елементів сайту, а саме:

– поле «Логотип сайту». Відкриває провідник надаючи можливість вибрати файлу формату .jpg. Після вибору поряд з кнопкою з’являється назва файлу. Обов’язкове для заповнення;
– поле «Названа сайту». Обов’язкове для заповнення;
– поле «Короткий опис сайту»;
– поле «Фонове зображення». Відкриває провідник надаючи можливість вибрати файлу формату .jpg. Після вибору поряд з кнопкою з’являється назва файлу. Обов’язкове для заповнення.

Дія «Зберегти» зберігає зміни та обновлює данні блоку «Заголовок сайту».

Після збереження та публікації стаття з’являється на головній сторінці порталу у стрічці новин. Для переходу до повного опису достатньо клацнути по відповідному полі з назвою статті (рис. 20):

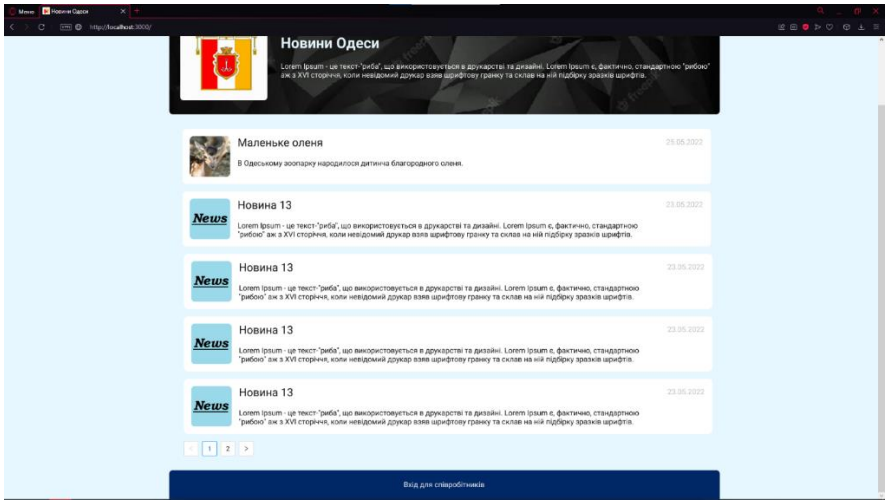



Рисунок 20 – Приклад стрічки новин

Розгорнуту статтю представлено на рисунку 21:

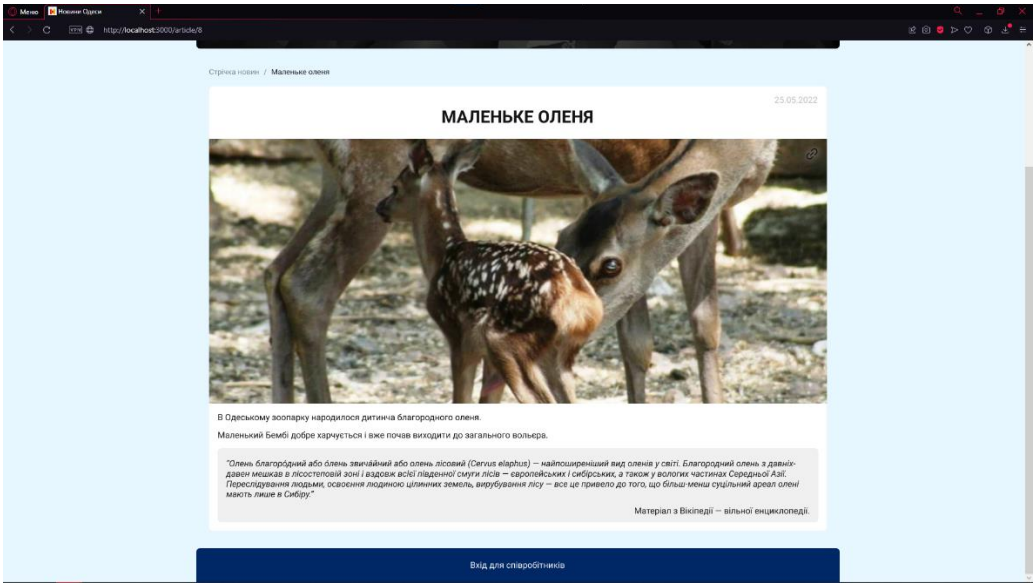



Рисунок 21 – Приклад опублікованої статті

## 4.2 Опис реалізації функціоналу

### 4.2.1 Хедер

файл «getHeader.js»

```
import sourceName from "./sourceName";
import Method from "/constants/Method";
import tags from "api/tags";

const getHeader = {
    queryFn (arg, store, extraOptions, baseQuery) {
        return baseQuery({
            url: `${sourceName}`,
            method: Method.GET,
        });
    },
    providesTags: [tags.HEADER],
};

export default getHeader;
```

Містить об’єкт конфігурації за яким бібліотека API генерує відповідний набір функцій для відправлення запитів до серверу. У даному випадку буде відправлен запит типу «GET» за ресурсом назва якого міститься у змінній sourceName. Отриманий результат буде помічений спеціальним тегом tags.HEADER, що дозволить за необхідності обновити отримані данні.

файл «header.js»

```
import {useGetHeaderQuery} from "../../api/siteApi";
import {ContentContainerLayout} from "../../layouts";
import styles from '/styles/Header.module.scss';
import classNames from "classnames";
import {Title} from "../../components";
import Text from "../../components/Text/Text";

function Header() {

    const {
        data
    } = useGetHeaderQuery();

    if (!data) return null;

    return (
        <ContentContainerLayout
            Container={(props) => (
                <header
                    {...props}
                    className={classNames([props.className,
styles.container])}
                    style={{background:  `url(${data.background})
center/cover`}}
                />
            )}
        >
            <div className={styles.logo_container}>
                <img   className={styles.logo}   src={data.logo}
alt="logo"/>
            </div>
            <div className={styles.text_container}>
                <Title
className={styles.text}>{data.title}</Title>
                <Text
className={styles.text}>{data.description}</Text>
            </div>
        </ContentContainerLayout>
    );}
```

```
export default Header;
```

Містить функціональний компонент Header, що є відповідальним за відображення заголовку сайту. Компонент використовує функцію useGetHeaderQuery згенеровану бібліотекою API за конфігураційним об’єктом описаним в файлі getHeader.js, що надає йому актуальні данні з бази даних. Отримані текстові данні відображаються за допомогою проксі компонент створених на базі відповідних компонент Ant Design.

Функція «getServerSideProps»

```
export const getServerSideProps = wrapper.getServerSideProps(
    (store) => async (context) => {
        const arg = {
            page: 1,
            perPage: 5,
        };
        store.dispatch(siteApi.endpoints.getHeader.initiate());

store.dispatch(siteApi.endpoints.getArticles.initiate(arg));

        await
Promise.all(api.util.getRunningOperationPromises());

        return {
            props: {},
        };
    }
);
```

Забезпечує Next всіма необхідними даними для генерації сторінки зі списком статей на стороні серверу. У даному випадку відбувається відправка запросів на бекенд сайту використовуючи функції getHeader та getArticles. До getArticles передається об’єкт з полями page та perPage, що мають значення 1 та 5 відповідно. Таким чином використовуючи getArticles Next отримує першу сторінку зі останніми опублікованими статтями у кількості 5 штук.

Функція getServerSideProps повертає пустий об’єкт props оскільки данні отримані з використанням згенерованих бібліотекою api функції типу

useQuery під час процесу відображення компоненти будуть надані з кеша api попередньо заповненого завдяки функції getServerSideProps.

### 4.2.2 Бекенд

Авторизація:

```
module.exports.login = async function (req, res) {
     try {
          if (!req.body.name) throw new Error('Будьласка, введіть ім’я користувача');
          if (!req.body.password) throw new Error('Будьласка, введіть пароль');

          const name = sanitize(req.body.name).trim();
          const password = sanitize(req.body.password).trim();

          if (name.length > 50) throw new Error('Максимальна довжина ім’я користувача 50 символів');
          if (password.length > 50) throw new Error('Максимальна довжина паролю 50 символів');

          const dbo = getDBO(req);
          const user = await dbo.collection(Collection.USER).findOne({name});

          if (!user) {
               // Not Found
               res.status(404).json({
                    message: "Аккаунт не знайдено.",
               });
               return;
          }

          if (user.status !== AccountStatus.ACTIVE) {
               // Inactive
               res.status(401).json({
                    message: "Аккаун не є активним",
               });
               return;
          }

          const passwordResult = bcrypt.compareSync(password, user.password);

          if (!passwordResult) {
               // Unauthorized
               res.status(401).json({
```

```
                    message: "Ім’я користувача та пароль не
дійсні",
               });
               return;
          }

          createCookie(res, user);
          res.status(200).json(User.toJson(req, user));
     } catch (error) {
          errorHandler(res, error);
     }
};
```

Описана функція відповідає за авторизацію користувача. Відбувається пошук у базі даних користувача за наданим іменем, перевірка чи є аккаунт активованим та перевірка відповідності паролей.

У разі некоректності отриманих даних сервер відправляє відповідне повідомлення, що буде відображене користувачу на фронтенді. У разі проходження всіх перевірок буде викликано функцію createCookie, для створення cookie з id аккаунта та терміном валідності в 9 годин, та відправка користувачу даних акаунту.

Реєстрація

```
module.exports.register = async function (req, res) {
     try {
          const dbo = getDBO(req);

          const name = sanitize(req.body.name)?.toLowerCase();
          const alreadyCreatedUser = await
dbo.collection(Collection.USER).findOne({name});

          if (alreadyCreatedUser) {
               res.status(403).json({ message: "Ім’я користувача
вже зайнято" });
               return;
          }

          const user = new User(req);
          const result = await
dbo.collection(Collection.USER).insertOne(user);
          const newUser = await
dbo.collection(Collection.USER).findOne({_id:
result.insertedId});
```

```
            res.status(200).json(User.toJson(req, newUser));
        } catch (error) {
            errorHandler(res, error);
        }
};
```

Описана функція відповідає за створення нового облікового запису користувача. Після проходження перевірки на унікальність отриманого ім’я облікового запису конструктор класу User створює нового користувача зі статусом «Не активний», у ролі «Редактор».

Далі функція зберігає нового користувача у БД. У разі існування облікового запису з таким же іменем сервер відправляє відповідне повідомлення, що буде відображене користувачу на фронтенді.

Вихід з панелі адміністрування:

```
module.exports.logout = function (req, res) {
    res.cookie('jwt', null);
    res.status(200).json();
}
```

Описана функція видаляє cookie авторизації, що унеможливлює подальше використання панелі адміністрування користувачем до моменту наступної авторизації.

Отримання даних користувача.

```
module.exports.me = async function (req, res) {
    res.status(200).json(User.toJson(req, req.user));
}
```

Описана функція повертає персональні дані користувача. Використовується для запитів попередньо опрацьованих проміжним програмним забезпеченням (middleware) з авторизації за стратегією «jwt» згенерованого наступним викликом методу бібліотеки passport:

passport.authenticate('jwt', { session: false })

### 4.2.3 Адмін-панель

Файл AuthPage.js:

```
import {Switch, Route, Redirect, useRouteMatch} from 'react-
router-dom';
import {LoginPage, RegistrationPage} from 'pages';

function AuthPage() {
    const {path, url} = useRouteMatch();

    return (
        <Switch>
            <Route path={`${path}/login`}
component={LoginPage}/>
            <Route path={`${path}/registration`}
component={RegistrationPage}/>
            <Route path={`${path}`}>
                <Redirect to={`${url}/login`}/>
            </Route>
        </Switch>
    );
}

export default AuthPage;
```

Функціональна компонента AuthPage містить систему навігації між вкладеними сторінками LoginPage та RegistrationPage, що містять відповідні модулі. AuthPage здійснює навігаціє відносно адреси за якою вона була візуалізована, що отримується за допомогою функції useRouteMatch.

Файл AuthRegister.js

```
import {CreateUserForm} from 'forms';
import {AuthLayout} from 'layouts';
import {Block} from "components";
import {useCreateUserMutation} from "api/authApi";
import {useEffect} from "react";
import {useHistory} from "react-router-dom";
import {useServerSideError} from "hooks";

function AuthRegister() {

const history = useHistory();
```

```
const [
trigger,
state,
] = useCreateUserMutation();

useServerSideError(state);

useEffect(
() => {
if (state.isSuccess) {
history.push('/auth/login');
}
},
[history, state.isSuccess],
);

return (
<AuthLayout>
<Block title={'Реєстрація'}>
<CreateUserForm onSubmit={trigger}
isLoading={state.isLoading}/>
</Block>
</AuthLayout>
);
}

export default AuthRegister;
```

Містить компоненту, що відображає форму CreateUserForm, та забезпечує відправку її даних до серверу за допомогою функції useCreateUserMutation. У разі успішного завершення операції створення нового облікового запису переадресує користувача на сторінку авторизації. У разі отримання від серверу повідомлення виводить його на екран завдяки функції useServerSideError.

Файл CreateUserForm.js:

```
import {Button, Form, Input} from "antd";
import {NavLink} from "react-router-dom";

function CreateUserForm(props) {
    const {
        onSubmit,
        isLoading,
    } = props;
    return (
```

```
        <Form
            layout={'vertical'}
            initialValues={{ remember: true }}
            onFinish={onSubmit}
            autoComplete="off"
        >
            <Form.Item
                validateFirst={true}
                label="Ім’я користувача"
                name="name"
                rules={[{ required: true, message: 'Поле
необхідно заповнити!' }]}
            >
                <Input />
            </Form.Item>
            <Form.Item
                validateFirst={true}
                label="Пароль"
                name="password"
                rules={[{ required: true, message: 'Поле
необхідно заповнити!' }, {min: 8, message: 'Мінімальна довжина 8
символів'}]}
            >
                <Input.Password />
            </Form.Item>
            <Form.Item
                validateFirst={true}
                label="Підтвердження паролю"
                name="passwordConfirmation"
                rules={[
                    {required: true, message: 'Поле необхідно
заповнити!' },
                    {min: 8, message: 'Мінімальна довжина 8
символів'},
                    (form) => {
                        const values =
form.getFieldValue('password');
                        return {
                            validator(_, value) {
                                if (value === values) return
Promise.resolve();
                                return Promise.reject('Паролі
повинні співпадати');
                            }
                        };
                    }
                ]}
            >
                <Input.Password />
            </Form.Item>
            <Form.Item>
```

```
                <Button type="primary" htmlType="submit"
size={'large'} block loading={isLoading}>
                    Зберегти
                </Button>
            </Form.Item>
            <NavLink to={'/auth/login'}>Увійти</NavLink>
        </Form>
    );
}

export default CreateUserForm;
```

Містить компоненту, що відображає форму створення нового облікового запису. Форма містить поля «Ім’я користувача», «Пароль» та «Підтвердження паролю». Всі поля є обов’язковими для заповнення, що забезпечується валідацією на стороні фронтенду.

Також на стороні фронтенду реалізовані валідатори мінімальної довжини пароля, та відповідності введених значень полів «Пароль» та «Підтвердження паролю».

Файл createUser.js:

```
import sourceName from "./sourceName";
import Method from "constants/Method";

const createUser = {
    queryFn (arg, store, extraOptions, baseQuery) {
        return baseQuery({
            url: `${sourceName}/register`,
            method: Method.POST,
            body: arg,
        });
    },
};

export default createUser;
```

Містить об’єкт конфігурації за яким бібліотека API генерує відповідний набір функцій для відправлення запитів до серверу. У даному випадку буде відправлен запит типу «POST» за ресурсом назва якого міститься у змінній sourceName, та з отриманим при виклику значенням body. На базі цього об’єкту буде згенеровано функцію useCreateUserMutation.

Файл useServerSideError:

```
import {useEffect} from "react";
import {notification} from "antd";

function useServerSideError(state) {
    useEffect(
        () => {
            if (state.error?.data?.message) {
                notification.error({
                    message: state.error.data.message,
                    placement: 'top',
                });
            }
        },
        [state.error],
    );
}

export default useServerSideError;
```

Містить функцію, що отримує поле state зі згенерованої API функції, та виводить на екран повідомлення, використовуючи метод notification.error, отримане від серверу у разі виникнення помилки.

Файл AuthLogin.js

```
import {LoginForm} from 'forms';
import {AuthLayout} from 'layouts';
import {Block} from "components";
import {useLoginMutation} from "api/authApi";
import {useEffect} from "react";
import {useHistory} from "react-router-dom";
import {useServerSideError} from "hooks";

function AuthLogin() {

    const history = useHistory();

    const [
        trigger,
        state,
    ] = useLoginMutation();

    useServerSideError(state);

    useEffect(
        () => {
```

```
            if (state.isSuccess) {
                history.push('/');
            }
        },
        [history, state.isSuccess],
    );

    return (
        <AuthLayout>
            <Block title={'Вхід до системи'}>
                <LoginForm onSubmit={trigger}
isLoading={state.isLoading}/>
            </Block>
        </AuthLayout>
    );
}

export default AuthLogin;
```

Містить компоненту, що відображає форму LoginForm, та забезпечує відправку її даних до серверу за допомогою функції useLoginMutation. У разі успішного завершення операції авторизації сервер повертає cookie з id облікового запиту користувача та данні облікового запису, користувач переадресовується на головну сторінку адмін-панелі.

У разі отримання від серверу повідомлення виводить його на екран завдяки функції useServerSideError.

Файл LoginForm.js

```
import {Button, Form, Input} from "antd";
import {NavLink} from "react-router-dom";

function LoginForm(props) {
    const {
        onSubmit, isLoading,} = props;
    return (
        <Form
            layout={'vertical'}
            initialValues={{ remember: true }}
            onFinish={onSubmit}
            autoComplete="off"
        >
            <Form.Item
                validateFirst={true}
                label="Ім’я користувача"
```

```
                name="name"
                rules={[{ required: true, message: 'Поле
необхідно заповнити!' }]}
            >
                <Input />
            </Form.Item>
            <Form.Item
                validateFirst={true}
                label="Пароль"
                name="password"
                rules={[{ required: true, message: 'Поле
необхідно заповнити!' }, {min: 8, message: 'Мінімальна довжина 8
символів'}]}
            >
                <Input.Password />
            </Form.Item>
            <Form.Item>
                <Button type="primary" htmlType="submit"
size={'large'} block loading={isLoading}>
                    Вхід
                </Button>
            </Form.Item>
            <NavLink
to={'/auth/registration'}>Зареєструватися</NavLink>
        </Form>
    );
}
export default LoginForm;
```

Містить компоненту, що відображає форму авторизації. Форма містить поля «Ім’я користувача» та «Пароль». Всі поля є обов’язковими для заповнення, що забезпечується валідацією на стороні фронтенду. Також на стороні фронтенду реалізовані валідатори мінімальної довжини пароля. У разі успішного завершення валідації полів форми її данні будуть передані до функції onSubmit.

Також, компонента приймає параметер isLoading, що використовується для своєчасного надання користувачу відклику на натиск кнопки «Вхід»: користувач бачить іконку, що сигналізує про очікування відповіді від серверу.

Файл login.js

```
import sourceName from "./sourceName";
import tags from "api/tags";
import Method from "constants/Method";

const login = {
```

```
    queryFn (arg, store, extraOptions, baseQuery) {
        return baseQuery({
            url: `${sourceName}/login`,
            method: Method.POST,
            body: arg,
        });
    },
    invalidateTags: [tags.ACCOUNT],
};

export default login;
```

Містить об’єкт конфігурації за яким бібліотека API генерує відповідний набір функцій для відправлення запитів до серверу. У даному випадку буде відправлен запит типу «POST» за ресурсом назва якого міститься у змінній sourceName, та з отриманим при виклику значенням body. На базі цього об’єкту буде згенеровано функцію useLoginMutation.

# ВИСНОВКИ

У зв’язку з багатьма причинами збільшення популярності і попиту на інтернет-портали новин, серед яких важливий вклад роблять швидкість розповсюдження новин, швидкість їх донесення до цільового читача незалежно від його місця перебування та часу доби та надана користувачу можливість впливати на зміст власної стрічки новин згідно його вподобань, вибрана предметна область є актуальною з точки зору сфери розробки програмного забезпечення та затребуваною як зі сторони інформаційних ресурсів, так і їх користувачів.

В результаті огляду аналогів у вигляді відомих інформаційних порталів та статистики їх відвідування було прийняте рішення направити проект на задоволення потреб місцевої громади шляхом створення сайту для новин місцевого характеру.

Для забезпечення порталу сучасним дизайном та зручним інтерфейсом з впізнаваємою та очікуваною поведінкою, а також достатньою швидкодією за умов MVP проекту були обрані засобі програмної розробки, що здатні виконати ці умови.

На стадії проектування був сформований список з функціональних вимог до проекту, детально описані ключові компоненти структури майбутнього проекту  та побудована діаграма варіантів використання.

Під час опису реалізації проекту був детально описано інтерфейс програмного продукту з наданням необхідних зображень і коментарів до них, що у купі виконують роль інструкції для майбутніх працівників порталу. Окрім того, був надано програмний код з детальними коментарями щодо особливостей його роботи та зв’язками між файлами на які він був поділений.

Проект було успішно реалізовано з виконанням виставлених до нього умов та з можливістю його подальшого використання та вдосконалення.

## ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ


1. Why Do You Need To Develop An Online News Portal In 2022?(Updated). URL: https://arulmjoseph.com/need-develop-online-news-portal-2017. (дата звернення 11.04.2022)
2. Топ 45 популярних новинних сайтів світу. URL: https://marketer.ua/ua/top-45-popular-news-sites-in-the-world/ (дата звернення 11.04.2022)
3. Офіційна стронінка. URL: https://edition.cnn.com/ (дата звернення 14.04.2022)
4. Офіційна стронінка. URL: https://www.thecipherbrief.com/(дата звернення 14.04.2022)
5. Рейтинг топсайтів України. URL: https://imi.org.ua/monitorings/rejtyng-top-sajtiv-ukrayiny-i34992 (дата звернення 17.04.2022)
6. Front-end JavaScript Frameworks Showdown: Vue vs. React vs. Angular. URL: https://clockwise.software/blog/angular-vs-react-vs-vue/ (дата звернення 17.04.2022)
7. Angular vs React vs Vue 2022. URL: https://athemes.com/guides/angular-vs-react-vs-vue/ (дата звернення 26.04.2022)
8. The Progressive JavaScript Framework. URL: https://vuejs.org/ (дата звернення 26.04.2022)
9. Vue. URL: https://clockwise.software/blog/angular-vs-react-vs-vue/121. (дата звернення 11.05.2022)
10. Angular vs React vs Vue.js: Which is the Best Choice for 2022? URL: https://javascript.plainenglish.io/angular-vs-react-vs-vue-js-which-is-the-best-choice-for-2022-5ef83f2257ab (дата звернення 11.05.2022)
11. Офіційна сторінка. URL: https://reactnative.dev/ (дата звернення 11.05.2022)

12. Що таке node.js. URL: https://uk.theastrologypage.com/node-js (дата звернення 20.05.2022)
13. Node.js – Express Framework. URL: https://www.tutorialspoint.com/nodejs/nodejs_express_framework.htm (дата звернення 20.05.2022)
14. Express.js. URL: https://www.tutorialsteacher.com/nodejs/expressjs (дата звернення 11.04.2022)
15. Express.js Mobile App Development: Pros and Cons for Developers. URL: https://apiko.com/blog/express-mobile-app-development/ (дата звернення 20.05.2022)
16. 12 Must-have Features Of A Best Online News Portal Design. URL: https://arulmjoseph.com/12-must-features-best-online-news-portal-design (дата звернення 24.05.2022)
17. How to create a news portal website? URL: https://www.4media.com/article/146,how-to-create-a-news-portal-website (дата звернення 24.05.2022)
18. Что мы знаем об Ant Design. URL: https://habr.com/ru/company/simbirsoft/blog/416925/ (дата звернення 24.05.2022)