МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет аеронавігації, електроніки та телекомунікацій

Кафедра авіаційних компю'терно-інтегрованих комплексів

ДОПУСТИТИ ДО ЗАХИСТУ

Завідувач випускової кафедри

_____Віктор СИНЄГЛАЗОВ

«19» __червня__2023р.

КВАЛІФІКАЦІЙНА РОБОТА

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ

«БАКАЛАВР»

Спеціальність 151 «Автоматизація та компю'терно-інтегровані технології»

Освітньо-професійна програма «Комп'ютерно-інтегровані технологічні процеси і виробництва»

Тема: Рекомендаційна система оптимального вибору безпілотного літального апарата із заданими властивостями

Виконавець: студент групи КП-403 Наволоков Олександр Олександрович

Керівник: доктор технічних наук, професор Синєглазов Віктор Михайлович

Нормоконтролер: _____ Філяшкін М.К

(підпис)

Київ 2023

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет аеронавігації, електроніки та телекомунікацій

Кафедра авіаційних комп'ютерно-інтегрованих комплексів

Освітній ступінь: бакалавр

Спеціальність 151 «Автоматизація та комп'ютерно-інтегровані технології»

Освітньо-професійна програма «Комп'ютерно-інтегровані технологічні процеси і виробництва»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Віктор СИНЄГЛАЗОВ

" 19 "__червня__2023 р.

ЗАВДАННЯ

на виконання кваліфікаційної роботи студента

Наволоков Олександр Олександрович

1. Тема проекту (роботи): " Рекомендаційна система оптимального вибору безпілотного літального апарата із заданими властивостями "

2. Термін виконання проекту (роботи): з _10.03.2023_р. до _14.06.2023_ р.

3. Вихідні данні до проекту (роботи): орієнтуватися в параметрах пошуку безпілотних літальних апаратів.

4. Зміст пояснювальної записки (перелік питань, що підлягають розробці): 1. Постановка задачі на створення рекомендаційної системи вибору безпілотного літального апарату із заданими властивостями; 2. Опис існуючих програмних рішень, що використовуються як прототип при створені рекомендаційної системи вибору безпілотного літального апарату; 3. Засоби розробки; 4. Опис програмної реалізації; 5. Методика роботи користувача.

5. Перелік обов'язкового графічного матеріалу: Титульна сторінка, Актуальність роботи, Мета та основні завдання роботи, Аналіз існуючих програмних рішень, Ключові моменти, діаграма варіантів використання, діаграма розміщення, діаграми компонентів, Засоби розробки, Інтерфейс головної сторінки, Результат роботи програми, Висновки.

6. Календарний план-графік:

№ пор.	Завдання	Термін виконання	Відмітка про виконання
1	Підбір літератури	10.03.2023 - 15.03.2023	виконано
2	Постановка задачі на створення рекомендаційної системи вибору БПЛА із заданими властивостями	16.03.2023 – 26.03.2023	виконано
3	Опис існуючих програмних рішень, що використовуються як прототип при створені рекомендаційної системи	27.03.2023 – 02.04.2023	виконано
4	Засоби розробки рекомендаційної системи вибору БПЛА із заданими властивостями	03.04.2023 – 18.04.2023	виконано
5	Опис програмної реалізації рекомендаційної системи вибору БПЛА із заданими властивостями	19.04.2023 – 03.05.2023	виконано
6	Методика роботи користувача у рекомендаційній системі вибору БПЛА із заданими властивостями	04.05.2023 – 25.05.2023	виконано
7	Формування висновків щодо виконаної роботи	26.05.2023 – 31.05.2023	виконано
8	Оформлення пояснювальної записки	01.06.2023 – 06.06.2023	виконано
9	Створення презентації	07.06.2023 – 14.06.2023	виконано

7. Дата видачі завдання: «10» березня 2023 р.

Керівник дипломної роботи _____ Синєглазов В. М.
(підпис керівника) (П.І.Б.)

Завдання прийняв до виконання _____ Наволоков О. О.
(підпис випускника) (П.І.Б.)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи «Рекомендаційна система оптимального вибору безпілотного літального апарата із заданими властивостями» має 125 аркушів, вона містить 2 додатки. В роботі наведено 50 рисунків та 2 таблиці. Включає перелік посилань на використані джерела з 21 найменуванням.

Метою даної дипломної роботи є створення рекомендаційної системи вибору безпілотного літального апарату із заданими властивостями. Дана система повинна організовувати ефективний процес пошуку, перегляду БПЛА, а також фільтрації та сортування знайдених БПЛА за певними критеріями та параметрами. Це досягається шляхом створення програмних засобів з надійного пошуку та перегляду даних БПЛА.

Головними критеріями для створення даної системи – стала можливість підтримки їхніх функцій актуальним версіями мови програмування JavaSrcipt та бібліотеки React, що призначена для розробки інтерфейсів.

В роботі було спроектовано систему (веб-сервіс) для пошуку та перегляду БПЛА із заданими властивостями.

Дану систему можливо використовувати як модуль та/або шаблон (прототип) для побудови більш масштабних систем даного типу.

Ключові слова: веб-сервіс, веб-сайт, система, авторизований доступ, безпілотний літальний апарат.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

БПЛА — безпілотний літальний апарат.

HTML — HyperText Markup Language.

CSS — Cascading Style Sheets.

UML — Unified Modeling Language.

JS – JavaScript

UI – User Interface (інтерфейс користувача).

БД – база даних.

СУБД – система управління базами даних.

MVC — Model-View-Controller.

QA — quality assurance.

ВСТУП

У наш час інформаційних технологій, важко знайти в комп'ютерному світі людину, яка хоч би на інтуїтивному рівні не розуміла, що таке інформаційна довідкова чи пошукова системи та бази даних.

У найширшому сенсі інформаційна система є програмним комплексом, функції якого полягають в підтримці надійного зберігання інформації в пам'яті пристрою, виконанні специфічних для даного застосування перетворень інформації і обчислень, наданні користувачам зручного і легко освоюваного інтерфейсу. Також в таких системах потрібно забезпечувати стандартизовані набори даних та параметрів, для кращого зберігання, обробки та видачі результаті.

У зв’язку з війною в Україні потрібно приймати різноманітні швидкі та якісні рішення для забезпечення перемоги на всіх фронтах. Одним із найкращих інструментів боротьби із окупантами – використання БПЛА, які корисні не тільки під час розвідки, але й самому бою.

Процесі підбору БПЛА для певного виду діяльності можуть виникнути складності з їх пошуком. На даний момент в Україні немає якісного та надійного сервісу для пошуку та вибору БПЛА. В зв’язку з тим, виникає необхідність створення надійної системи для підбору БПЛА за певними параметрами та характеристиками – для прискорення та покращення вибору.

Для вирішення цих проблем створюється система (веб-сервіс), що дозволить здійнсювати якісний та надійний пошук БПЛА із заданими властивостями.

Отже, актуальність дипломної роботи зумовлена кількома чинниками:

— Відсутністю в Україні якісного та надійного сервісу для пошуку та вибору БПЛА.

— Активного використання БПЛА у війні проти окупантів.

— Мета роботи: організація ефективного процесу пошуку та перегляду БПЛА із заданими властивостями.

Завдання:

— Проаналізувати графічний інтерфейс та функціонал сервісів, які можуть виступити прототипом при створенні рекомендаційної системи пошуку БПЛА із заданими параметрами.

— Сформувати вимоги до системи, що створюється.

— Спроектувати та розробити веб-сервіс пошуку та перегляду БПЛА із заданими властивостями.

— Протестувати продукт.

РОЗДІЛ 1
ПОСТАНОВКА ЗАДАЧІ ПОСТАНОВКА ЗАДАЧІ НА СТВОРЕННЯ РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ ВИБОРУ БЕЗПІЛОТНОГО ЛІТАЛЬНОГО АПАРАТУ ІЗ ЗАДАНИМИ ВЛАСТИВОСТЯМИ

Мета роботи даного дипломного проекту – моделювання, проектування та створення рекомендаційної системи вибору безпілотного літального апарату із заданими властивостями. Вона досягається шляхом створення програмних засобів пошуку та перегляду БПЛА, що попереднього записуються в БД даних систем (веб-сервісів).

Призначенням даного програмного продукту – отримання зручного, легкого та чіткого графічного інтерфейсу користувача, а також відповідного функціоналу пошуку та перегляду БПЛА із заданими властивостями. А також надання можливості додавання нових БПЛА до БД даної системи – з метою отримання можливості перегляду її за допомогою функціоналу даної системи.

При виконанні даної дипломної потрібно здійснити моделювання та проектування системи (веб-сервісу) за допомогою діаграми варіантів використання, діаграми розміщення та діаграм компонентів. На основі вищезгаданої інформації – розробити та створити:

— Графічну частину продукту – за допомогою мови розмітки гіпертексту HTML та спеціальної мови стилю сторінок CSS, мов об’єктно-орієнтованого підходу JavaScript, а також бібліотек графічних інтерфейсів React.js, Material UI, Chart.js та React-swipeable-views.

— Функціональну частину продукту – за допомогою мов об’єктно-орієнтованого підходу JavaScript, а також різноманітних бібліотек для належної роботи функціоналу серверної частини сайту та взаємодії з БД даного сайту.

Даний програмний продукт повинен підтримувати наступні функції:

— Можливість переглядати всі наявні на сайті БПЛА. Дані БПЛА заносяться на сайт його адміністратором.

— Можливість здійснювати пошук потрібного БПЛА за допомогою певних критеріїв та параметрів.

— Можливість сортувати БПЛА, на головній сторінці сайту, за певним параметром чи критерієм. Наприклад, за алфавітом.

— Можливість авторизації, для адміністратору сайту, для переходу в адмін панель (сторінку). За допомогою даної сторінки будуть додаватися нові БПЛА в БД сайту, для їх подальшого виведення на головну сторінку сайту.

— Можливість для додавання нового БПЛА в БД сайту, з метою їх подальшого виведення на головну сторінку сайту.

— Можливість перегляду інформації про кожен БПЛА окремо.

— Можливість переходу на офіційне джерело певного БПЛА в глобальній мережі Інтернет.

РОЗДІЛ 2
ОПИС ІСНУЮЧИХ ПРОГРАМНИХ РІШЕНЬ, ЩО ВИКОРИСТОВУЮТЬСЯ ЯК ПРОТОТИП ПРИ СТВОРЕНІ РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ ВИБОРУ БЕЗПІЛОТНОГО ЛІТАЛЬНОГО АПАРАТУ

Рекомендаційна система оптимального вибору безпілотного літального апарата із заданими властивостями – повинна бути створена за допомогою інструментів та підтримувати функціонал, що приведено в розділі 1 даного проекту. Дана система буде представлена у вигляді веб-сервісу (веб-сайту) пошуку та перегляду БПЛА із заданими властивостями.

Ретельний аналіз інформаційних ресурсів глобальної мережі Інтернет показав – відсутність ідентичних або хоча б схожих систем вибору БПЛА. Це призводить, в свою чергу, до неможливості якісного аналізу даної системи на предмет дублювання графічного інтерфейсу чи функціоналу. Також важко точно зрозуміти переваги та недоліки даної системи, в порівнянні з можливими іншими.

В даному розділі буде описано веб-сервіси (веб-сайти) графічний інтерфейс чи/та функціонал яких використовувався прототипом для створення рекомендаційної системи вибору безпілотного літального апарату із заданими властивостями.

Так, як ця система є унікальною та немає схожих аналогів – тому вважаю, що створення даної системи є актуальним.

2.1. Інтернет-магазин Rozetka

Розетка (англ. Rozetka. стилізовано, як ROZETKA) — український інтернет-магазин та маркетплейс, що з'явився 2005 року. Станом на серпень 2020 року сайт посідає 7 місце серед найвідвідуваніших в Україні.

Даний сервіс надає можливість робити зручні та швидкі замовлення та покупки товарів з різних куточків України. Інтерфейс головної сторінки веб-сайту даного інтернет-магазину зображено на рис. 2.1.

Даний сервіс має наступні переваги в використані:

— Зручний графічний інтерфейс користувача.

— Наявність поділу товарів на категорії для кращого пошуку потрібного. Наприклад, холодильники чи пральні машинки віднесено в категорію побутові електроприлади, а шарфи та брюки – в категорію одяг.

— Можливість пошуку різних товарів за допомогою пошукового рядку, який допомогаю знайти потрібний товар за ключовим словом чи фразою.

— Можливість пошуку різних товарів за списком фільтрів із відповідними параметрами.

— Можливість сортування знайдених товарів за певними критеріями.

— Можливість перегляду детальної інформації про товар та його характеристики.

В даному дипломному проекті було взято за основу принципи й інструменти для перегляду та пошуку даних такя, як на веб-сайті інтернет-магазину Rozetka.

«Рис. 2.1. Інтерфейс головної сторінки веб-сайту інтернет-магазину Rozetka»

2.2. Сервіс E-Katalog

E-Katalog (Е-Каталог) — сервіс порівняння товарів та їх характеристик, що належить торговельній системі Nadavi.

Даний сервіс надає можливість робити зручне та швидке порівняння товарів, що продають різноманітні інтернет-магазини, а також характеристик даних товарів. Інтерфейс головної сторінки сервісу даного інтернет-магазину зображено на рис. 2.2.

Даний сервіс має наступні переваги в використані:

— Можливість підбору товару за необхідними параметрами.

— Можливість якісного порівняння товарів за їх характеристиками.

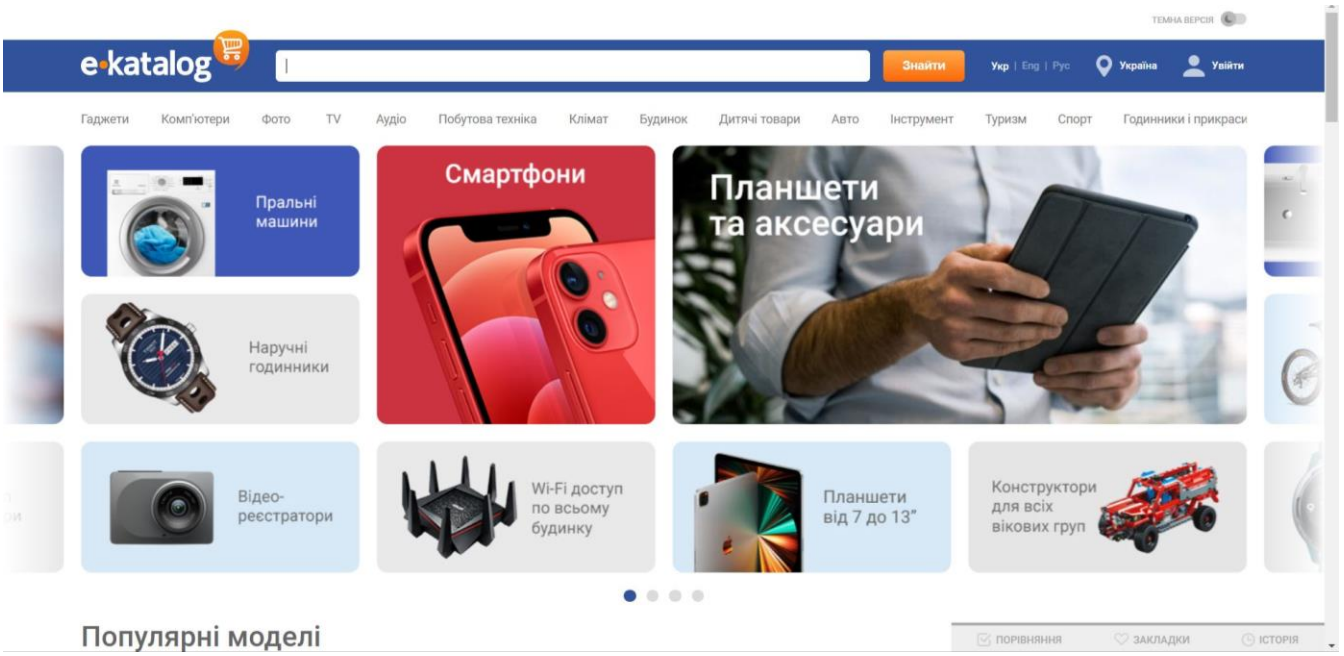
— Можливість перегляду відгуків про товари, що залишили покупці.

— Можливість написати свій відгук про певний товар.

— Наявність функції пошуку аксесуарів для товарів.

— Наявність можливості ознайомитися з рейтингами товарів і брендів.

В даному дипломному проекті було взято подібну ідею згруповування всіх даних про БПЛА та зручне відображення їх основних характеристик. Також, для зручності, на кожний БПЛА є посилання на оригінальний ресурс, де є можливість додатково почитати інформацію про даний БПЛА, або ж придбати його.



«Рис. 2.2. Інтерфейс головної сторінки сервісу E-Katalog»

Висновки за розділом 2

Проаналізовано веб-сайти, прототипи яких використовуються при розробці рекомендаційної системи вибору безпілотного літального апарату із заданими властивостями. Встановлено характерні для цих сайтів переваги. Переваги даного функціоналу було використано в даному дипломному проекті при створені та проектуванні веб-сайту пошуку та перегляду БПЛА із заданими властивостями.

РОЗДІЛ 3
ЗАСОБИ РОЗРОБКИ

В даному дипломному проекті були використанні наступні інструменти для розробки:

— Мова гіпертекстової розмітки HTML.

— Мова стилей таблиць CSS.

— Об'єктно-орієнтована прототипна мова програмування JavaScript.

— React. Це JavaScript-бібліотека (з відкритим кодом) для розробки графічних користувацьких інтерфейсів.

— Та інші бібліотеки, що потрібні для належної роботи майбутнього веб-сервісу (веб-сайту). Більш детально описані в даному розділі.

Незважаючи на те, що даний проект буде мати невеликий та немасштабний функціонал, в ньому буде використано принципи «клієнт-серверної» розробки, з подальшою публікацією даного сервісу (а саме серверної частини) в глобальній мережі Інтернет.

Архітектура клієнт-серверної розробки – це є одним із архітектурних шаблонів програмного забезпечення та є домінуючою концепцією у створенні розподілених мережних застосунків і передбачає взаємодію та обмін даними між ними [3]. Її головний принцип полягає в розподіленні будь-якого сервісу чи системи на компоненти, що працюють як автономні системи та взаємодіють між собою. Такими компонентами є:

— Клієнти, що використовують певні функції системи. Ці функції називаються сервісами. Сервіси надаються спеціальними серверами, що підключені до глобальної мережі Інтернет та працюють із нею.

— Сервери, що надають інформацію та послуги (сервіси). Для їх отримання клієнт, а саме клієнтська частина системи (сервісу), повинен належним чином звернутися до серверної частини для отримання певних результатів.

— Глобальна мережа інтернет, що забезпечує взаємодію між клієнтами та серверами.

Перевагами даної архітектури є можливість підтримки паралельного функціонування клієнтів та незалежно один від одного. Також і сервери – мають можливість взаємонезалежної діяльності.

Дана архітектура – є найкращим рішенням при проектуванні та розробці веб-сервісів та веб-систем. Для створення клієнтської частини використовується методологія «front-end», а для серверної – методологія «back-end».

В даному проекті окремі мови, технології чи бібліотеки можуть відповідати, як за «front-end», так і за «back-end». Тому, немає сенсу виносити кожну із методології в окремий розділ дипломної роботи. В даному підрозділі буде описано принципи за якими працюють дані методології.

Розробка «front-end» — це розробка, що зосереджена на елементах, які бачить користувач та з якими взаємодіє [4-7]. Суть даної методології (принципу) в наступному: надання користувачу можливості отримання зручного графічного інтерфейсу, для відображення даних та отримання до них доступу. Тобто, «front-end» потрібен для зручності роботи користувача. Так, як без наявності останнього – всі дані будуть відображатися у вигляді незрозумілого програного коду, а доступ до них буде довгим та громіздким.

Саме завдяки методиці «front-end» виникає можливість легкості та зручності при роботі з веб-програмами та веб-сайтами.

При розробці клієнтської частини даного продукту потрібно реалізувати наступний функціонал:

— Можливість правильного відображення на екранах різних пристроїв. Тобто, розширення екрану неповинна впливати на якість продукту.

— Можливість якісної роботи з різними операційними системами.

— Можливість роботи та належного відображення сайту в більшості відомих веб-браузерах. Такими є – Mozilla Firefox, Internet Explorer, Chrome, Microsoft Edge, Opera.

Далі потрібно розглянути наступну методологію, що буде використана в рамках даного дипломного проекту для розробки серверної частини. Це методологія «back-end».

Розробка «back-end» — це набір апаратно-програмних засобів, за допомогою яких реалізована логіка роботи сайту [4-7]. Тобто, це функціонал, який відбувається поза браузером користувача (функціонал не видно користувачу) та який реалізовує взаємодію з ним. Це означає, що серверна частина проекту – отримує запити користувача, обробляє їх та видає відповідні результати.

Ця методологія фокусується саме на тому, як працює та функціонує веб-сервіс. Для цього розробник функцій веб-серверу повинен створити спеціальний програмний код, що наддасть користувачу можливість отримувати дані та послуги з будь-якої точки Земної кулі, при наявності доступу в глобальну мережу Інтернет.

3.1. Мова HTML

Мова HTML (HyperText Markup Language — мова розмітки гіпертексту) — стандартизована мова розмітки документів для перегляду веб-сторінок у браузері. Браузери отримують HTML документ від сервера за протоколами HTTP/HTTPS або відкривають з локального диска, далі інтерпретують код в інтерфейс, який відображатиметься на екрані монітора [4-7].

Документи, які створені на основі мови HTML спеціальним клієнтським програмним забезпеченням – веб- браузерами. Головні цілі веб-браузерів це:

— Надання доступу до документів, які відформатовано у зручному для користувача.

— Надання користувачу зручного інтерфейсу для відображення веб-сторінок та передачі введених користувачем даних на веб-сервер.

Документи, що створені за допомогою мови HTML, як правило, мають розширення типу «*.html» та розмічені HTML-тегами. Веб-браузер перетворює дані теги та їх вміст в зручний веб-документ, та відображає їх користувачу. Це відбувається за рахунок того, що браузер розуміє принципи функціонування цих тегів та як їх належно обробляти. Відповідно, теги в веб-браузерах не відображаються.

Типовий HTML-документ містить наступну структуру:

— HTML-теги. Описують структуру, зовнішній вигляд та функції вмісту веб-документу.

— Вміст веб-документу. Тобто, текстова інформація даного документу.

HTML-документ, як і будь-який інший, складається з певних функціональних частин, основні з яких заголовок документу та його тіло. Header (заголовок) HTML- документу містить його назву, а також службову інформацію, що описує його вміст. Body (тіло) HTML- документу містить в собі основну інформацію, а саме – вміст документу.

Приклад структури документу:

```
<html>
    <head>
        <title> This is HTML TITLE </title>
    </head>
    <body>
        <h1> This is HTML HEADER </h1>
    </body>
</html>
```

3.2. Мова CSS

Мова CSS (Cascading Style Sheets, укр. Каскадні таблиці стилів) — це спеціальна мова (мова стилів), за допомогою якої описують вигляду документів (як і де відображати елементи веб-сторінки), написаних мовами розмітки даних. Найчастіше CSS використовується для документів, котрі розмічені мовою HTML, XHTML та XML [4-7].

CSS дозволяє усунути проблему дублювання програмного коду, що відповідає за графічний дизайн однотипних елементів. Розробник дизайну створює певний елемент та придумає йому назву чи ім’я. Далі, він здійснює опис всіх потрібних властивостей даного елементу, наприклад – кольору, шрифту тексту та його розміру. В результаті – створений елемент використовується в інших частинах проекту. Це

відбувається не за рахунок дублювання коду, а за допомогою спеціальних посилань на даний елемент.

Тобто, CSS – надає можливість зручного та швидкого створення дизайну веб-сторінок, чим стандартні способи, яким володіє мова HTML.

3.3. Мова JavaScript

JavaScript – динамічна, об'єктно-орієнтована прототипна мова програмування, що використовується для створення сценаріїв веб-сторінок, що надає можливість на боці клієнта (пристрої кінцевого користувача) взаємодіяти з користувачем, керувати браузером, асинхронно обмінюватися даними з сервером, змінювати структуру та зовнішній вигляд веб-сторінки [4-7, 12].

Дана мова – є одним з найкращих інструментів для створення скриптів та сценарії будь-якого веб-сервісу (веб-сайту), що працює за принципом «клієнт-серверної» архітектури». JavaScript можливо використовувати як на стороні веб-клієнта, так і на стороні веб-серверу.

Програмний код, написаний мовою JavaScript, вбудовується в код HTML-сторінки (HTML-контейнерів). Далі – дана сторінка здійснює зв’язок із веб-сервером (відбувається при її завантажені). Наступним кроком йде зміна значень атрибутів HTML-контейнерів і властивостей середовища відображення веб-документу. Це відбувається за рахунок роботи скриптів написаних мовою JavaScript.

Важливо особливістю вищеописаного процесу є те, що при такому підході HTML-сторінка не перезавантажується. Тобто, зміни, що відбуваються в документі браузера – не викликають перевантаження сторінки.

JavaScript використовує об’єктну модель документу для надійного керування веб-сторінками, що завантажуються на клієнтській стороні. Дана модель презентує кожен HTML-контейнер, що знаходяться в веб-документі, певним спеціальним об’єктом. Кожен з таких об’єктів володіє своїми певними параметрами – властивостями, методами та подіями.

Об’єктна модель – описує взаємодію між об’єктами, методами, властивостями та подіями, що присутні та працюють в програмному забезпеченні браузера. Тобто, це принцип взаємодії між веб-сторінками та браузером.

3.4. Бібліотека React

В даному дипломному проекті будуть створені та описані вимоги до графічного дизайну веб-сайту, що буде створюватися (дані вимоги описані в розділі 4). Для належного виконання даних вимог потрібно використовувати механізми та інструменти, що дозволять створити красивий, елегантний та зручний графічний інтерфейс веб-сайту. Створення таких інтерфейсів надає бібліотека React.

React (старі назви: React.js, ReactJS) — відкрита JavaScript бібліотека для створення інтерфейсів користувача, яка покликана вирішувати проблеми часткового оновлення вмісту веб-сторінки, з якими стикаються в розробці односторінкових застосунків [4-7, 12].

Дана бібліотека надає розробнику можливість створювати інтерактивні веб-сервіси, що використовують дані, які можуть змінюватися без перезавантаження сторінки. Завдяки своїй швидкості та простоті використання вона стала дуже популярною в масштабних комерційних веб-проектах. Проте, вона обмежена лише використанням при розробці та створені графічного інтерфейсу користувача.

Бібліотека React побудована за допомогою архітектури модель-вид-контролер (MVC). Дана архітектура описана в наступному в одному із підрозділів.

3.5. Інші платформи та бібліотеки проекту

В даному дипломному проекті, окрім бібліотеки React, використовуються ще й інші бібліотеки, що відповідають за його клієнтський та серверний функціонал. Назву та функціонал кожної із них описано нижче в даному підрозділі (див. рис. 3.1

Однією із найголовніших бібліотек в даному списку – є Node.js.

Таблиця 3.1

Бібліотеки, що використовуються в межах дипломного проекту

Node.js	Програмна платформа для запуску та збірки javascript коду.
Express	Бібліотека для створення, налаштування сервера, а також його конфігурації та комунікації із базою даних та клієнтською частиною додатку. Вона надає клієнту всі статичні файли, що потрібні для роботи, через мережу інтернет по протоколу HTTP.
Material UI	Одна з найбільш популярних бібліотек компонентів React, що потрібна для створення графічного інтерфейсу користувача. Компоненти та всі шаблонні елементи створені відповідно до принципів Google Material UI, але водночас вони є компонентами React.
Config	Бібліотека для полегшення взаємодії із конфігураційними файлами веб-сервісу, що проектується та створюється
Express-validator	Бібліотека, яка надає різні інструменти та middlewares, для валідації отриманих даних від клієнта та виведення відповідних помилок на екран веб-браузера. Middleware – це проміжне програмного забезпечення, що складається з агентів, які є посередниками між різними компонентами додатків, та, зазвичай, використовується в розподілених застосунках.
Jsonwebtoken (JWT)	Бібліотека, яка надає можливість реалізувати безпечну авторизацію за допомогою згенерованих токенів для кожного клієнта і конкретної сесії. Є однією із найбільш популярних популярних бібліотек при вирішені проблем безпечної авторизації.
Mongoose	Бібліотека для підключення СУБД MongoDB. Дана СУБД буде використовуватися в рамках даного дипломного проекту (буде описано в розділі 3.7).

Multer	Бібліотека для опрацювання отриманих файлових даних від клієнта. Надає спеціальний middleware для перехвату файлів і їх збереження у вказаній папці.
Bcrypt.js	Бібліотека для кодування і декодування даних. Потрібна для кодування паролю користувача, що передається глобальною мережею Інтернет на серверну частину веб-сервісу. Дана бібліотека потрібна для того, щоб хакери не змогли на пряму бачити пароль в БД.
Axios	Бібліотека для більш простішого посилання запитів на сервер від клієнта. Використовується замість внутрішньої функції fetch().
Chart.js	Бібліотека для конфігурації та відображення різноманітних графіків і діаграм. Через нею буде відображатися функція, що схематично зображує графічні числові дані про БПЛА
React-swipeable-views	Бібліотека для відображення стрічки із зображень. Надає функціонал для керування «свайпом» курсору або пальця (якщо сайт буде відкритий на телефоні).
Concurrently	Бібліотека для одночасного запуску декількох процесів. В межах даного проекту вона потрібна для запуску одночасно і серверної програми, і клієнтської.
Nodemon	Бібліотека для автоматичного перезапуску серверної програми, за умови, якщо було змінено якийсь файл у проекті. Використовується для зручності при розробці програми.

3.6. Архітектура MVC

Даний проект працює за принципом архітектури MVC.

Архітектура MVC (MVC, Модель–представлення–контролер, англ. Model-view-controller) — архітектурний шаблон, який використовується під час проектування та

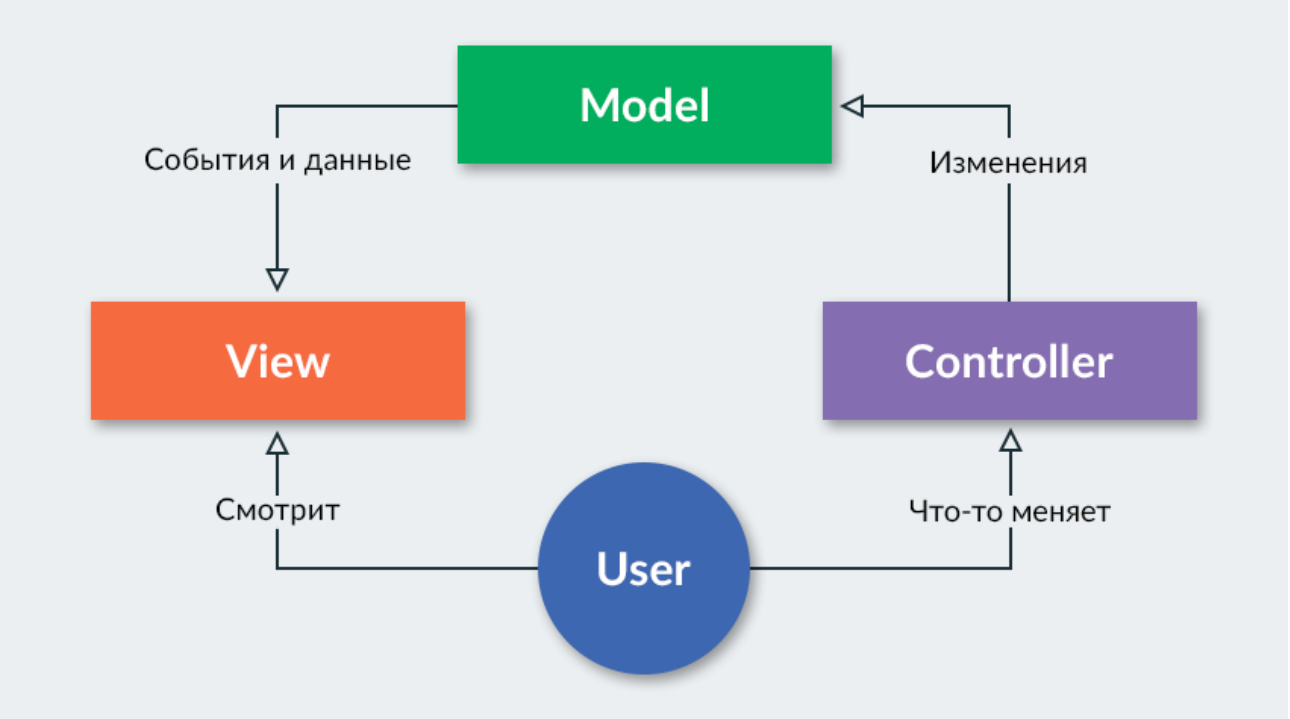
розробки програмного забезпечення [8-11]. Ця архітектура потрібна для відокремлення логіки функціоналу веб-сервісу від графічного інтерфейсу користувача. Дана мета досягається за рахунок поділу програмного продукту на три параметри: модель, вигляд та контролер.

Модель виконує функції керування поведінкою програмного продукту та даними, які циркулюють в ньому. Даний параметр здійснює реагування на запити отримання інформації та її модифікації. Як результат – відбувається оповіщення користувача про внесені зміни. Це досягається за допомогою окремих файлів чи цілих баз даних.

Вигляд виконує функції зручного відображення інтерфейсу користувача програмного продукту. Це досягається за рахунок спеціальних графічних елементів, наприклад, форм, клавіш, текстових полір, блоків та ін.

Контролер виконує функції отримання уведених користувачем даних, а також виконує виклики об’єктів моделі. В результаті він видає певні дії на виконання програмним продуктом.

Загальна структура архітектури MVC зображена нижче (див. рис. 3.1).



«Рис. 3.1. Структура архітектури MVC»

3.7. База даних Mongo

Веб-сервіс, що розробляється, буде містити наступні дані, що потрібно надійно зберігати та обробляти [4, 11, 13]:

— Облікові дані адміністраторів веб-сайту, у вигляді логінів та паролів.

— Інформація БПЛА та параметри що їх супроводжують, а саме – назву (ім’я), модель, тип двигуна, розмах крил та ін. Більш детально всі параметри описано в розділах 4.3.2 та 4.3.3.

Для збереження та обробки цих даних використовувалась БД Mongo. А для зручної роботи з цією БД було використано MongoDB.

MongoDB — документо-орієнтована система керування базами даних (СКБД) з відкритим вихідним кодом, яка не потребує опису схеми таблиць [14-16]. Вона є прикладом NoSQL-системи, тобто використовує не опис таблиць даних, а JSON-подібні документи та моделі. Тобто, дана СУБД підтримує зберігання документів в JSON-подібному форматі. Також вона має наступні переваги використання:

— Гнучка мова для формування запитів.

— Можливість створювати індекси для різних збережених атрибутів.

— Забезпечує якісне зберігання великих бінарних об'єктів.

— Має функцію логування (журналювання) операцій зі зміни і додавання даних в БД.

— Підтримує реплікацію даних. Реплікація – це принцип розподілення даних між вузлами, що дозволяє зберігати копії цих даних на різних вузлах мережі.

— Надає якісний функціонал для побудови відмовостійких систем.

3.8. Середовище розробки

Для створення веб-сервісу з пошуку та перегляду БПЛА із заданими властивостями – використовувалось середовище розробки Visual Studio Code версії 1.77.3, що створене компанією Microsoft, а для перегляду проміжних та кінцевих результатів – веб-браузер CCleaner версії 112.0.

Visual Studio Code, який також зазвичай називають VS Code — це редактор вихідного коду, створений Microsoft із Electron Framework для Windows, Linux і

macOS [17]. Даний редактор дозволяє створення, налагодження, підсвічування синтаксу коду. Також перевагою його використання є наявність функцій інтелектуального завершення коду та його рефакторинг.

Дане середовище було використано з наступних причин:

— Підтримка більшості мов програмування, таких як – C, C#, C++, Fortran, Go, Java, JavaScript, Node.js, Python, Rust.

— Дозволяє проектувати та створювати (розробляти) веб-додатки на базі Node.js.

— Підтримує функцію IntelliSense для JavaScript, TypeScript, JSON, CSS і HTML, а також підтримує налагодження Node.js. IntelliSense – це технологія, що розроблена компанією Microsoft, яка дозволяє дописувати назву функції при введені початкових букв. Дана технологія покращує та пришвидшує розробку програмних продуктів.

Висновки за розділом 3

Розглянуто та описано основні засоби розробки, що використовуються в процесі проектування та створення рекомендаційної системи вибору БПЛА із заданими властивостями, а саме – веб-сервісу (веб-сайту) з пошуку та перегляду БПЛА із заданими властивостями. Такими засобами є мова розмітки гіпертексту HTML, мова стилів таблиць CSS, мови програмування JavaScript, бібліотека React, що використовується для розробки графічних інтерфейсів користувача. Також було використано інші бібліотеки, які покращили розробку функціоналу клієнтської та серверної частин програмного забезпечення.

Розглянуто та описано принцип архітектури MVC, за допомогою якої функціонувати дана система (веб-сервіс), а також розібрано принцип збереження облікових даних та даних про БПЛА в БД веб-сайту, що проектується та створюється.

РОЗДІЛ 4
ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

Програмний продукт, а саме веб-сайт (веб-сервіс) для пошуку та перегляду БПЛА із заданими властивостями було створено відповідно до основних етапів розробки програмного забезпечення, а саме [11, 13]:

1. Аналіз вимог до програмного забезпечення.
2. Проектування програмного забезпечення.
3. Розробка програмного забезпечення.
4. Тестування програмного забезпечення.
5. Впровадження та супровід.

В даному розділі буде детально описано та розібрано етапи проектування та розробки, які вважаються найбільш складними та найбільш масштабними при створені програмного продукту.

4.1. Аналіз вимог до програмного забезпечення

Аналіз вимог до програмного забезпечення — це одна із головних частин розробки програмного забезпечення, яка складається з [11, 13]:

— Збору вимог до програмного продукту.

— Систематизації вимог до програмного продукту.

— Виявлення взаємозв’язків між компонентами продукту.

— Створення документації до програмного продукту.

Результатом даного етапу – є описання технічного завдання на створення програмного продукту. Короткий варіант технічного завдання описано в даному розділі.

4.1.1. Глосарій

В даному підрозділі описано глосарій (словник) з термінами, шо використовуються в процесі створення, впровадження та використання сайту (див. рис. 4.1).

Таблиця 4.1

Терміни, що є ключовими в технічному завданні та їх опис

Сайт (Веб-сайт чи веб-сервіс)	Інформаційна система, сукупність веб-сторінок та залежного вмісту, доступних у глобальній мережі Інтернет та які об’єднані за змістом та навігацією, за допомогою доменного імені. Доступ до вмісту сайту користувачі отримують за допомогою набору взаємопов’язаних HTML-сторінок.
HTML-сторінка	Це файли, що є носіями інформації в глобальній мережі Інтернет. Файл, а також його вміст згенеровані особливим способом, що може переглядатися. Такі файли можуть відкриватися за допомогою веб-браузера, як частина єдиного сайту.
Гіперпосилання	Активний (виділений кольором) елемент HTML-сторінки (текст, зображення чи кнопка на веб-сторінці), що задається спеціальним HTML-тегом.
HTML-тег	Це іменована мітка (керуючий код), що призначений для форматування HTML-сторінки.
Браузер (веб-браузер)	Прикладне клієнтське програмне забезпечення, що потрібне для перегляду вмісту HTML-сторінок.
HTML-форма	Це частина HTML-сторінки, що є набором елементів, (текстових полів, випадаючих списків тощо) та призначена для взаємодії з користувачем сайту. За допомогою цих форм користувач може ввести будь-яку інформацію та відправити її для обробки на сервер.

Поле форми	Шаблонний та структурний елемент, що містить типову інформацію. Такою інформацією може бути текст, дата, числові значення тощо.
Користувач	Особа, що використовує доступний йому функціонал сайту, при цьому в даний сайт не вноситься зміни, тобто, він доступний будь-якому іншому користувачу. Для належного використання потрібний доступ в глобальну мережу Інтернет
Адміністратор	Персона, яка здійснює та використовує функції сайту з найвищими доступними привілеями, які надаються користувачам.

4.1.2. Предмет розробки

Предмет розробки – це веб-сайт (веб-сервіс), що створюється з метою надання користувачам функцій полегшеного та покращеного пошуку та перегляду всіх наявних на даному сайті БПЛА. Також є можливість записувати нові БПЛА та їх характеристики адміністратором даного сайту.

Відповідно до описаного вище, стає зрозумілим, що основне призначення веб-сайту (веб-сервісу) – це:

— Можливість пошуку та перегляду всіх наявних на сайтів БПЛА.

— Можливість пошуку одно чи декількох БПЛА завдяки пошуковому рядку. Даний рядок допомагає здійснювати пошук за рахунок використання ключового слова чи фрази.

— Можливість пошуку одно чи декількох завдяки списку фільтрів пошуку. Даний список дозволяє вибрати певні параметри, зі списку доступних, за якими буде здійснюватися пошук БПЛА.

— Можливість зміни «теми» (фону) сайту, в світлих чи темних тонах.

— Можливість сортувати БПЛА, що виведені на головній сторінці сайту, за певним критерієм. Наприклад, за алфавітом.

— Можливість переглянути інформацію про параметри та характеристики конкретного БПЛА.

— Можливість перейти, за допомогою даного сайту, на офіційне посилання даного БПЛА в глобальній мережі Інтернет.

— Можливість додавати нові БПЛА до БД даного веб-сайту. Цю функцію може використовувати лише адміністратор сайту.

4.1.3. Вимоги до графічного дизайну

При розробці та створені веб-сайту повинно бути передбачена можливість вибору двох «тем» (фонів). Користувач зможе на свій власний смак та своїм бажанням вибрати одну із доступних «тем». Вони будуть мати наступні кольори:

Світла «тема»:

— Використання білого та синього кольорів для фону сайту.

— Використання синього, червоного та чорного кольорів для відповідних клавіш, іконок та графіків сайту.

— Використання білого кольору для рядків вводу текстового поля.

— Використання чорного кольору для текстового наповнення сайту.

Темна «тема»:

— Використання чорного та темно-сірого кольорів для фону сайту.

— Використання синього, червоного та чорного кольорів для відповідних клавіш, іконок та графіків сайту.

— Використання чорного кольору для рядків вводу текстового поля.

— Використання білого кольору для текстового наповнення сайту.

Дизайн сайту повинен бути виконаний за допомогою мови CSS та JavaScript, а саме використання наступних компонентів:

— «React.js» – бібліотека для створення красивих та елегантних графічних інтерфейсів.

— «Material UI» - бібліотека компонентів React для створення графічних інтерфейсів. Компоненти та всі шаблонні елементи створюються відповідно до

принципів Google Material UI, але водночас вони є компонентами бібліотеки «React.js».

— «Chart.js» – бібліотека для конфігурації і відображення різноманітних графіків і діаграм. За допомогою неї буде здійснено графічне відображення числових даних про БПЛА.

— «react-swipeable-views» – бібліотека для відображення стрічки із зображень. Надає функціонал для керування «свайпом» курсору або пальця (якщо сайт буде відкритий на телефоні).

Веб-сайт повинен коректно відображатись таких браузерах, як Opera 98.0.4759.39, Google Chrome 102.0.5005.63 та CCleaner Browser 112.0.

Потрібно створити структуру сайту, що містить наступні елементи:

— Адресний рядок – для введення доменного імені сайту.

— Пошуковий рядок – для пошуку БПЛА за ключовим словом чи фразою.

— Блок (лівий сайдбар) відображення списку фільтрів з клавішами для застосування та скасування фільтрів.

— Клавішу для перемикання «тем» сайту.

— Header. Блок, в якому відображаються карточки БПЛА. Кожна карточка має зображення, назву, короткий опис та клавішу для детального перегляду. Тобто, для переходу на сторінку конкретного БПЛА – потрібно натиснути на цю клавішу чи на зображення БПЛА.

— Пагінатор. Для відображення кількості сторінок сайту та кількості БПЛА, що розміщаються на одній сторінці.

— Відображення, на кожній окремій сторінці, детальної інформації про БПЛА, а також клавіша для переходу за посиланням. Воно переносить на офіційний ресурс БПЛА в глобальній мережі Інтернет.

— Форма авторизації в адмін панелі.

— Форма для створення нового БПЛА та його запису в БД сайту.

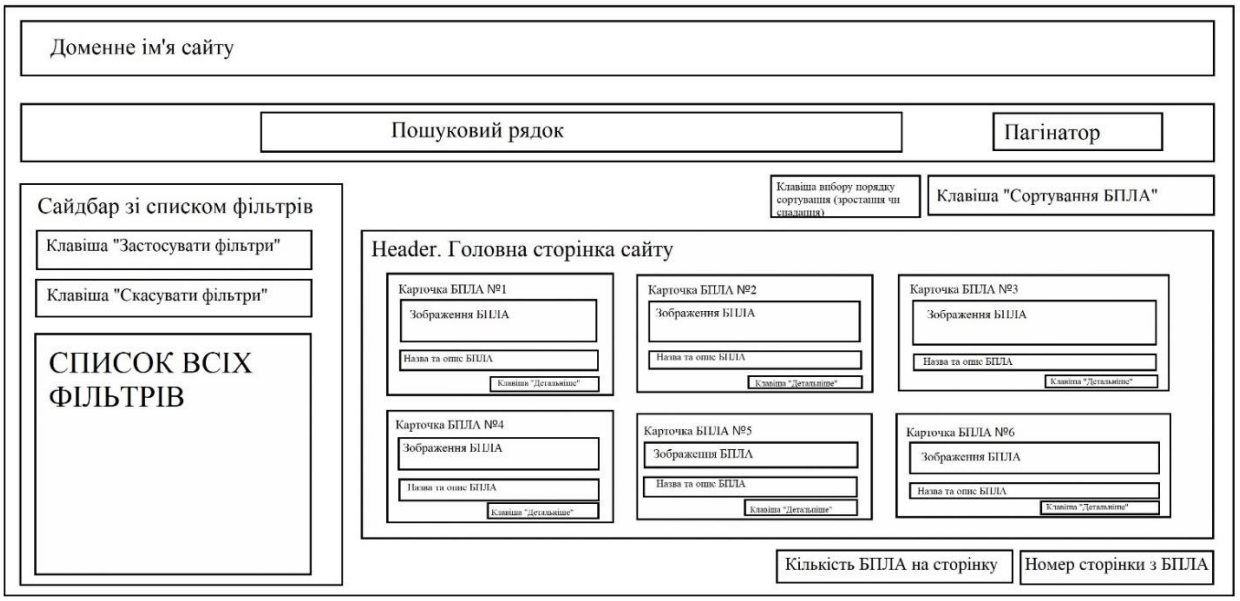
Нижче було створено графічну схему наступних сторінок веб-сайту:

— Головної сторінки веб-сайту (див. рис. 4.1).

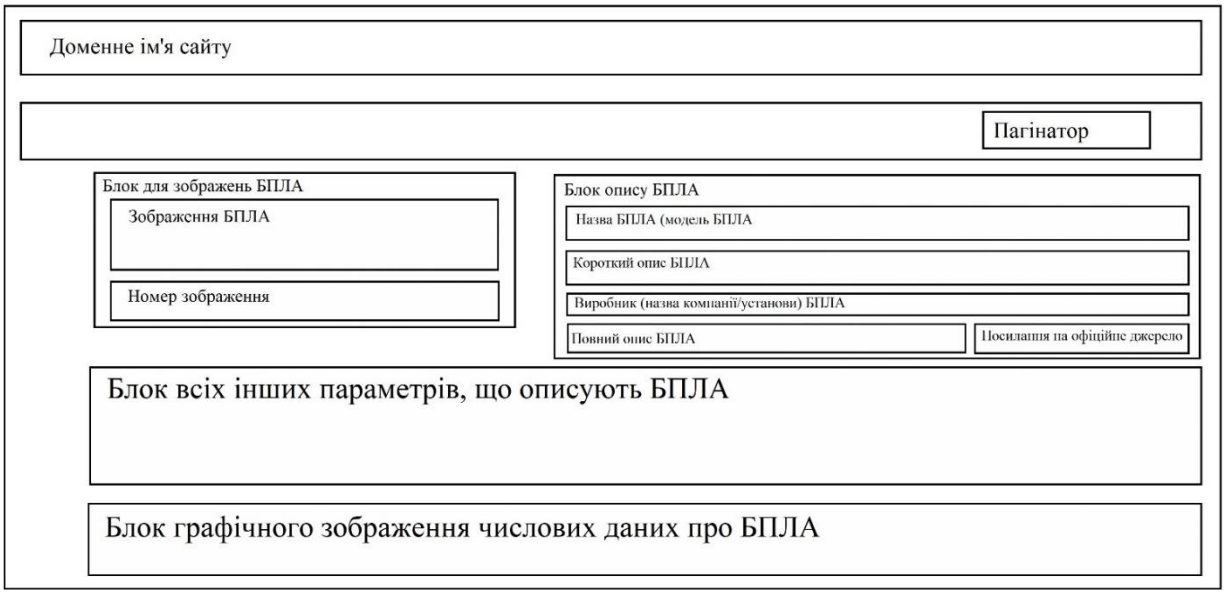
— Сторінки з детальним зображення кожного БПЛА (див. рис. 4.2).

— Сторінки авторизації (входу) в адмін панель (див. рис. 4.3).

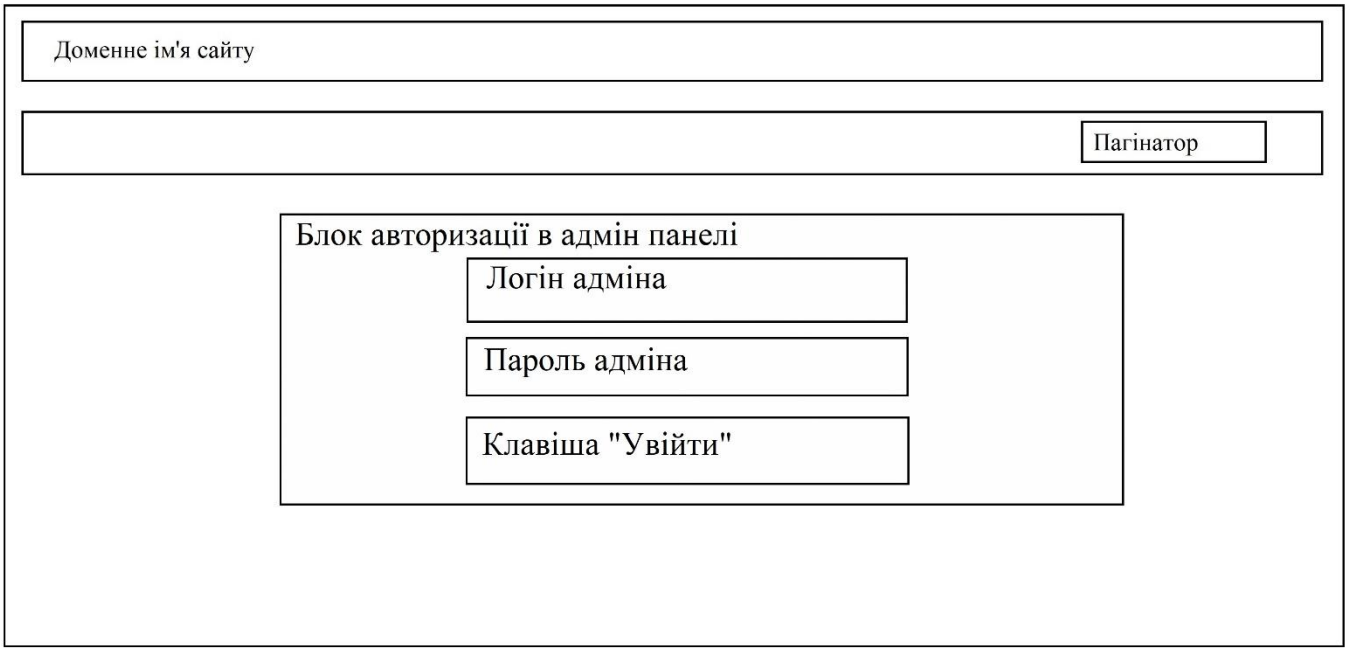
— Сторінки адмін панелі для створення нового БПЛА та його запису в БД сайту (див. рис. 4.4 - 4.7).



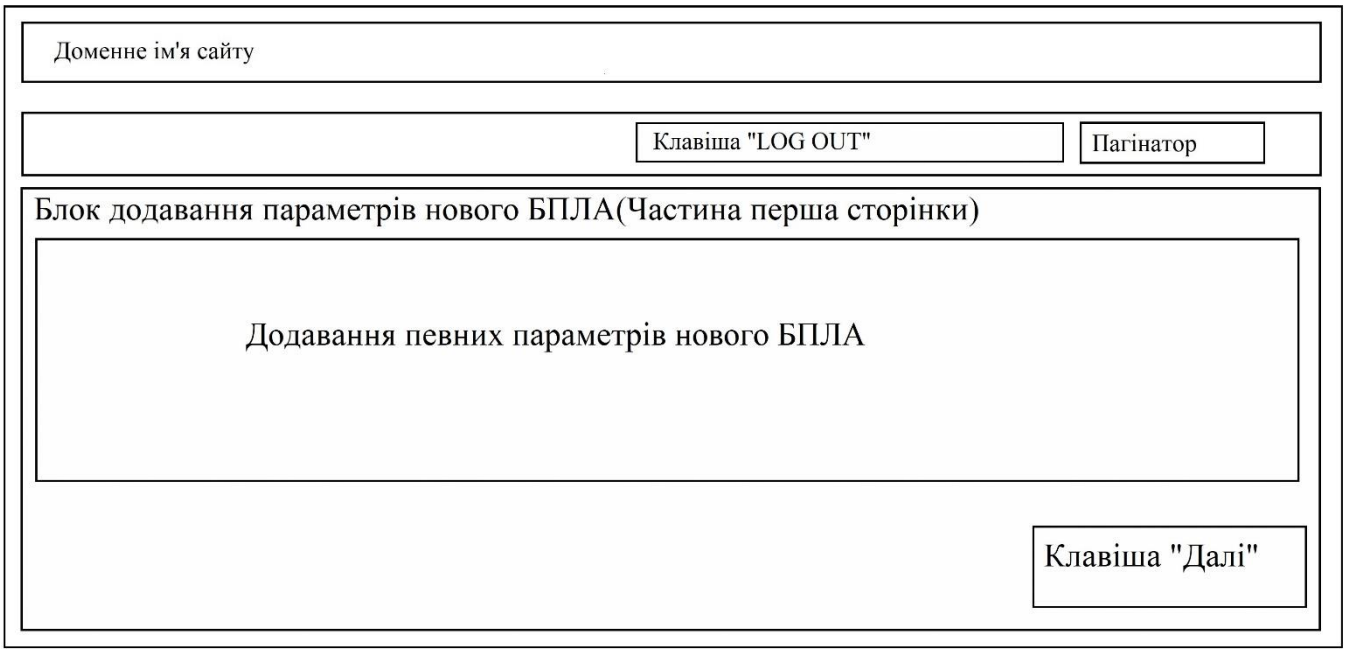
«Рис. 4.1. Графічна схема шаблону головної сторінки веб-сайту»



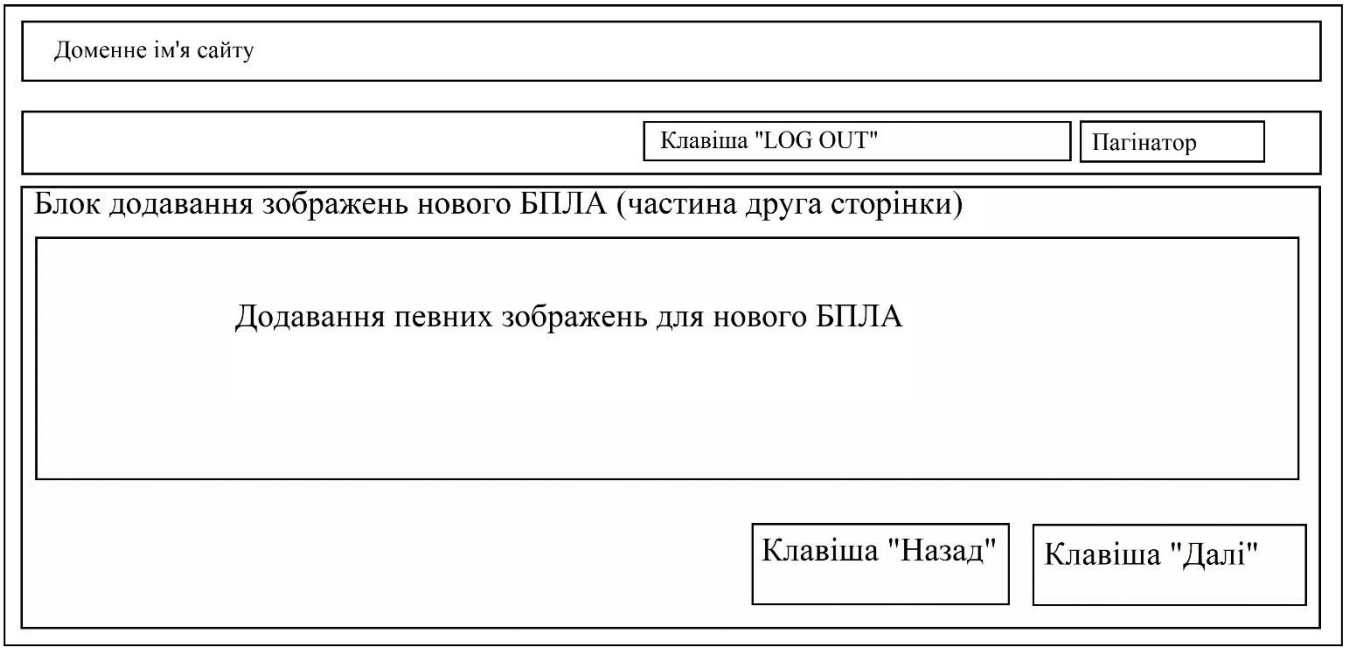
«Рис. 4.2. Графічна схема шаблону сторінки з детальним зображення кожного БПЛА»



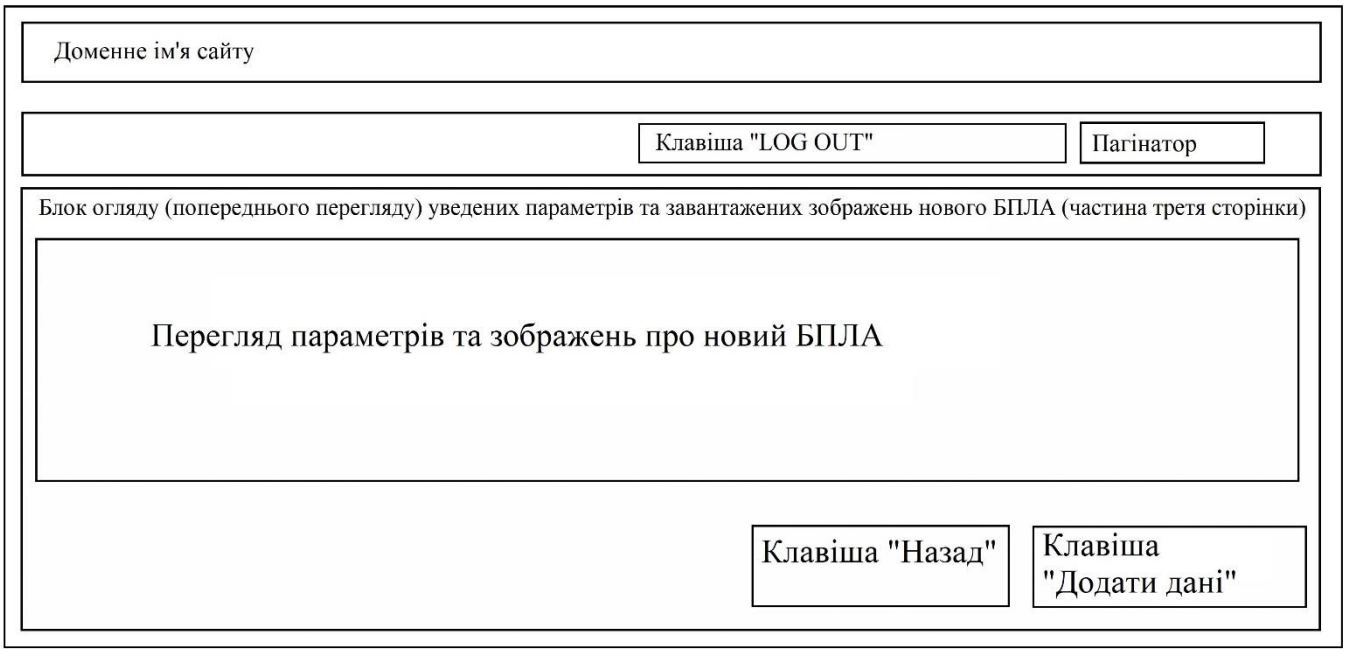
«Рис. 4.3. Графічна схема шаблону сторінки авторизації для переходу в адмін панель веб-сайту»



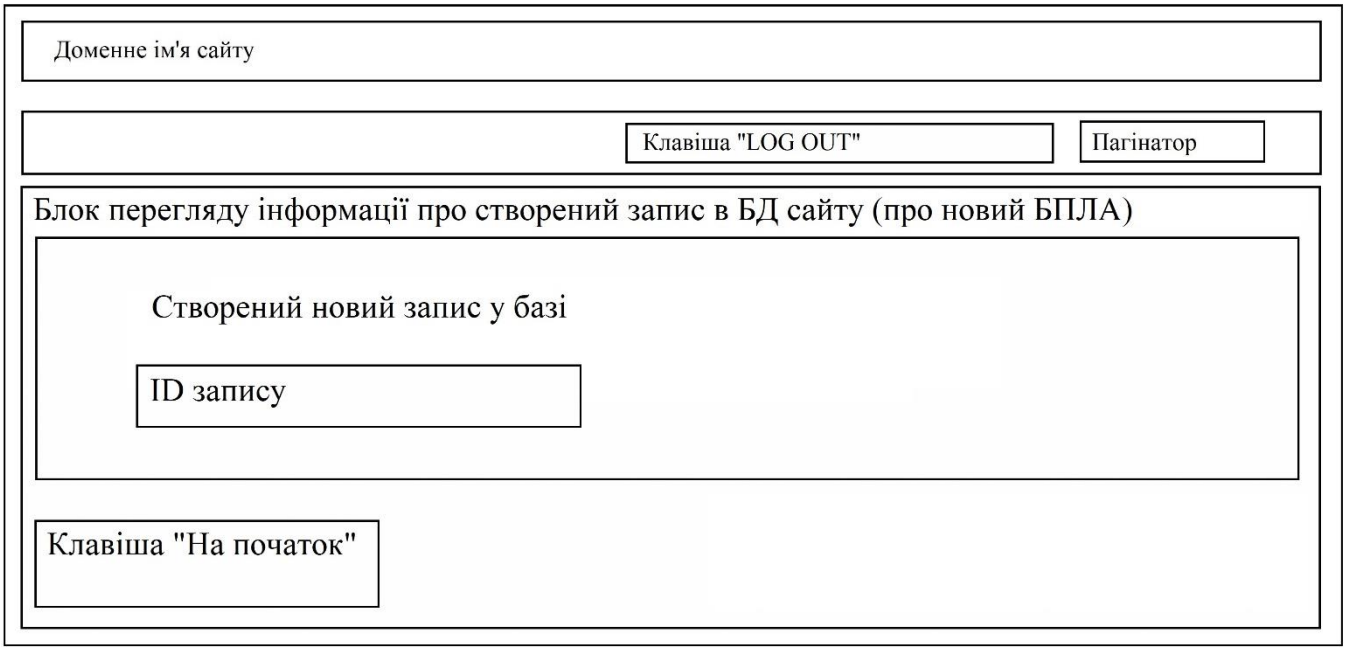
«Рис. 4.4. Графічна схема шаблону першої частини сторінки адмін панелі для створення нового БПЛА та його запису в БД сайту»



«Рис. 4.5. Графічна схема шаблону другої частини сторінки адмін панелі для створення нового БПЛА та його запису в БД сайту»



«Рис. 4.6. Графічна схема шаблону першої частини сторінки адмін панелі для створення нового БПЛА та його запису в БД сайту»



«Рис. 4.7. Графічна схема шаблону міні сторінки із повідомленням сторінки про успішне створення та публікацію нового БПЛА в БД сайту»

Доступ до сторінок авторизації (входу) в адмін панель та, безпосередньо, сторінки адмін панелі для створення нового БПЛА та його запису в БД сайту – має лише адміністратор даного веб-сайту.

4.1.4. Вимоги до функціональності сайту

Єдиною мовою, на якій функціонує, сайт – є українська. Веб-сайт (веб-сервіс) призначений для пошуку та перегляду БПЛА із заданими властивостями, а також має можливість запису в БД сайту нових БПЛА (адміністратором даного сайту). Даний сайт чи його прототип може використовуватися різних держустановах України, в тому числі у військових відомствах України. Тому структура сайту, а також весь функціонал, повині бути на державній мові. Необхідності забезпечення можливості надання інформації користувачеві іншими мовами немає.

Сайт повинен надати користувачам можливість:

— Переглядати всі наявні на сайті БПЛА. Дані БПЛА заносяться на сайт його адміністратором.

— Здійснювати пошук потрібного БПЛА за допомогою пошукового рядку. Для цього в цей рядок потрібно ввести ключове слово чи фразу для належного пошуку.

— Здійснювати пошук потрібного БПЛА за допомогою фільтрів пошуку. Для цього в цей потрібно вибрати потрібні параметри зі всіх можливих, що присутні в списку фільтрів.

— Змінювати «тему» (фон) сайту. На вибір будуть доступні дві «теми» – в світлих та темних тонах.

— Мати функцію «пагінатора» – для виведення на екран сайту лише певної кількості БПЛА, а не обов’язково всіх разом. А також відображати номер сторінки з БПЛА (на головній сторінці веб-сайту).

— Мати функцію сортування БПЛА, на головній сторінці сайту, за певним параметром чи критерієм. Наприкад, за алфавітом.

— Мати форму авторизації для переходу в адмін панель, за допомогою якої будуть додаватися нові БПЛА в БД сайту, з метою їх подальшого виведення на головну сторінку сайту.

— Мати адмін панель для додавання нового БПЛА в БД сайту, з метою їх подальшого виведення на головну сторінку сайту.

— Мати можливість перегляду інформації про кожен БПЛА окремо.

— Мати можливість переходу на офіційне джерело певного БПЛА в глобальній мережі Інтернет.

4.1.5. Вимоги до вмісту сайту

Потрібно створити наступні сторінки веб-сайту:

— Головна сторінка веб-сайту, де здійснюється відображення всіх наявних для пошуку БПЛА.

— Сторінка опису обраного БПЛА – на даній сторінці відбувається детальний опис конкретно вибраного БПЛА користувачем сайту.

— Адмін сторінка з формою авторизації. За допомогою неї адміністратор веб-сайту може авторизуватися, з метою подальшого додавання нових БПЛА в БД сайту.

— Сторінка для додавання нового БПЛА до БД веб-сайту.

— Головна сторінка веб-сайту повинна відображати список всіх наявних, в БД даного сайту, БПЛА. Відображення відбувається у вигляді карточок. Кожна карточка складається з зображення БПЛА, його моделі, назви та короткого опису про нього. Також на головній сторінці розміщується:

— Пошуковий рядок, для пошуку БПЛА за певним слово, реченням чи їх частиною. В даному випадку – це назва БПЛА, його модель, короткий та довгий опис.

— Список фільтрів з можливість вибору одного чи декількох параметрів пошуку БПЛА. Наприклад, тип двигуна чи країна виробник. Більш детально всі параметри фільтрів будуть описані в розділі 4.3.2.

— Можливість сортування, всіх чи тільки вибраних, БПЛА в порядку зменшення чи збільшення за певними параметрами. Наприклад, сортування за назвою в порядку зростання. Тобто, від першої букви алфавіту та до останньої.

— Перемикач для зміни «теми» (фону веб-сайту). Даний сайт буде мати світлу та темну «теми».

— «Пагінатор» – функція, що дозволяє вибрати кількість елементів БПЛА, які максимально можливо виводити на одну сторінку, та номер сторінки, на якій розміщені БПЛА.

Сторінка опису обраного БПЛА повинна містити зображення певного БПЛА, посилання на офіційне джерело БПЛА в глобальній мережі Інтернет, ім’я БПЛА, модель, опис та інші параметри. Список всіх параметрів описано в розділах 4.3.2, 4.3.3 та розділі 5.

Адмін сторінка з формою авторизації адміністратора веб-сайту повинна містити форму для уведення облікових даних адміністратора, а сам:

— Логіну адміністратора, у вигляді електронної пошти.

— Паролю адміністратора.

— Клавішу для підтвердження авторизації.

Сторінка для додавання нового БПЛА до БД веб-сайту повинна містити форму для додавання параметрів БПЛА та створення, на основі цих параметрів, нової

карточки БПЛА. Тобто, запис БПЛА в БД веб-сайту, для подальшого виведення на екран головної сторінки даного сайту. Ця форма буде складатися з трьох частин:

— Перша частина. Введення параметрів БПЛА. Наприклад, його назву, модель, короткий опис чи тип двигуна. Список параметрів більш детально описано в розділах 4.3.2 та 4.3.3. Також міститься клавіша для переходу на другу частину сторінки.

— Друга частина. Надає змогу завантажити одне чи декілька зображень даного БПЛА. Перше зображення, зі списку всіх, буде виводитися на головній сторінці веб-сайту, а інші – на сторінці перегляду конкретного БПЛА. Також дана сторінка буде мати клавішу для повернення до попереднього етапу (першої частини) та переходу до наступного етапу (третьої частини).

— Третя частина. Надає змогу зробити попередній огляд всієї інформації про БПЛА, що увів адміністратор веб-сайту. Також наявні клавіши для повернення на попердній етап (друга частина) та для підтвердження створення та публікації нового БПЛА.

— Також наявна міні сторінка, на якій зображено відповідне повідомлення про успішне створення та публікацію нового БПЛА, а також ID даного БПЛА в БД веб-сайту. Щоб продовжувати надалі створювати та публікувати інші БПЛА – присутня клавіша про перехід для створення нового БПЛА. Ця клавіша переносить адміністратора на першу частину сторінки.

На кожному із етапів сторінки для додавання нового БПЛА – присутня клавіша, що дозволяє розлогінитись адміністратору з даної сторінки. Тобто, перейти на головну сторінку веб-сайту.

Детальне графічне зображення даних сторінок наведено в розділі 4.1.3.

4.2. Проектування програмного забезпечення

Проектування програмного забезпечення — це один із основних та важливих процесів в створенні програмних продуктів, особливо масштабних комерційних та державних проектів. За допомогою нього здійснюється:

– проектування компонентів, інтерфейсів та архітектури системи;

– проектування характеристик, графічної та функціональної частини програмного продукту;

– проектування кінцевого результату.

Одним із найкращих інструментів в процесі проектування –є моделі в нотації мови графічного опису для об'єктного моделювання UML [11, 18]. Дана мова використовується для проектування програмного забезпечення та систем, проектування та моделювання бізнес-процесів. Також дана мова використовується для створення та документування результатів моделювання.

Потрібно пам'ятати, що UML не є мовою програмування. Але вона дозволяє, на основі побудованих схем та діаграм, побудувати (згенерувати робочий код, який можливо в подальшому використовувати в кодах програмних продуктів.

Для проектування веб-сайту з пошуку та перегляду БПЛА із заданими властивостями було використано наступні діаграми:

– діаграму варіантів використання;

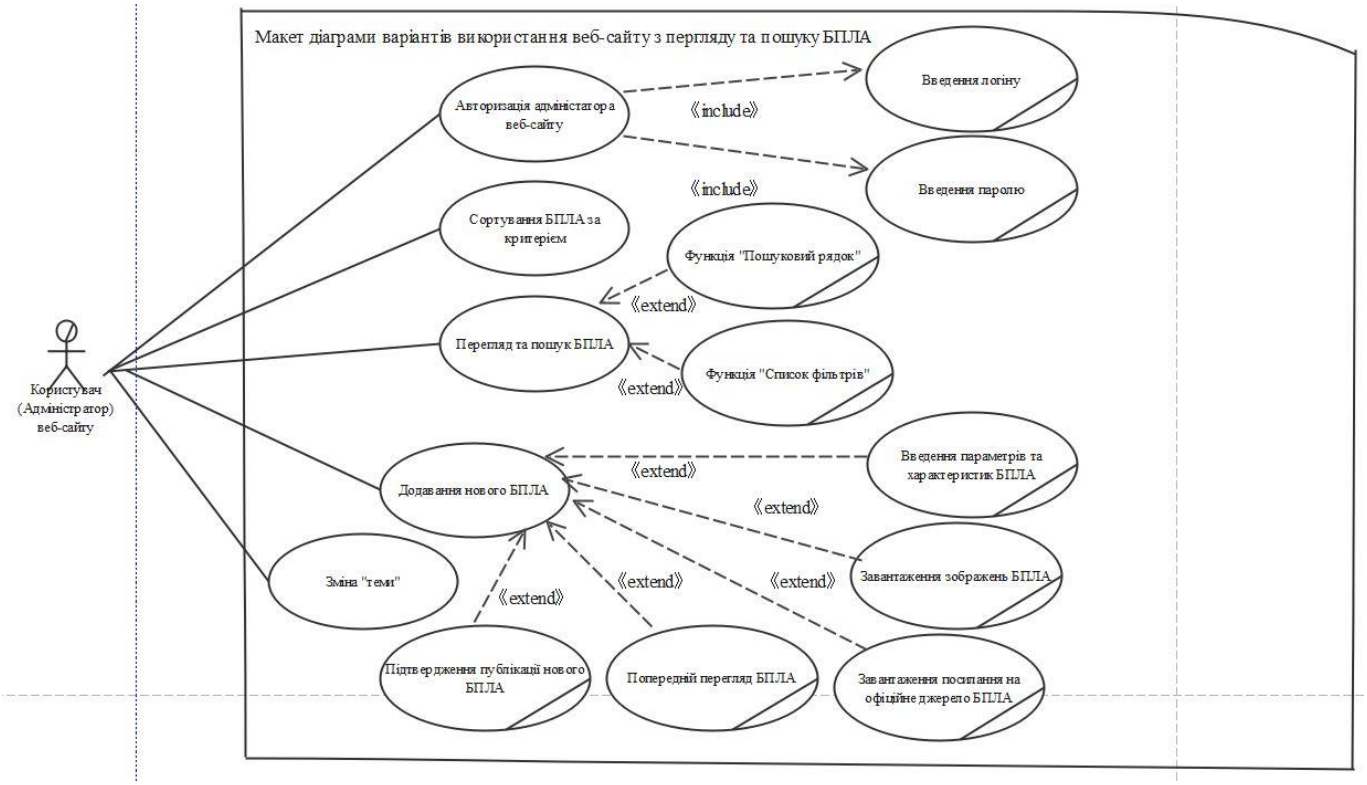
– діаграму розміщення;

– діаграми компонентів.

4.2.1 Діаграма варіантів використання

Діаграма варіантів використання — діаграма, на якій зображуються варіанти використання проектованої системи, укладені в кордон (границю) суб'єкта, і зовнішні актори, а також взаємовідношення між акторами і варіантами використання [11, 18]. Основне завдання та ціль цих типів діаграм— створити зрозумілу та надійну комунікацію між розробником програмного забезпечення та замовником послуг, а також надає можливість обговорювати функіонал програми на одному рівні.

Для даного дипломного проекту діаграма варіантів використання була створена за допомогою десктопного програмного продукту Wondershare EdrawMax (див. рис. 4.8).



«Рис. 4.8. Діаграма варіантів використання для проектування веб-сайту з перегляду та пошуку БПЛА»

Основні елементи, які є в даних типах діаграм – це ектор, юридична чи фізична особа, що використовує функції системи чи програмного продукту, а також варіанти використання – це фукції, які надають дана система чи програмний продук. Ектором в даній системі виступають двоє осіб – це звичайни користувач веб-сайту, а також його адміністратор. Кожен з екторів має свій функціонал, з яким може працювати.

Головними варіантами використання виступають:

— «Перегляд та пошук БПЛА». Функція, що дозволяє переглядати БПЛА, що наявні на сайті (записані в його БД), а також здійснювати пошук потрібних БПЛА за певними параметрами чи характеристиками. З даним функціоналом працює користувач сайту.

— «Сортування БПЛА за критерієм». Функція, що дозволяє переглядати сортувати БПЛА за певним критерієм, в порядку зростання чи спадання цього критерія. Наприклад, за алфавітом, від першої до останньої букви. З даним функціоналом працює користувач веб-сайту.

— «Авторизація адміністратора веб-сайту». Функція, що дозволяє авторизуватися в адмін панелі веб-сайту, для подальшого створення БПЛА та додавання його до БД сайту. З даним функціоналом працює адміністратор веб-сайту.

— «Додавання нового БПЛА». Функція, що дозволяє створити новий БПЛА та додати запис про його створення до БД сайту. З даним функціоналом працює адміністратор веб-сайту.

— «Зміна «теми»». Функція, що дозволяє змінити «тему» (фон) сайту. На вибір доступні два фони – у світлих та темних тонах. З даним функціоналом працює користувач або ж адміністратор веб-сайту.

Для варіанта використання «Авторизація адміністратора веб-сайту» є два варіанта використання «Введення логіну» та «Введення паролю», які є складовими основного компоненту. Відношення між варіантами використання «include» означає, що варіант використання, що включається, наприклад, «Введення паролю» повинен бути обов’язковим для варіанту використання, що доповнюється, «Авторизація адміністратора веб-сайту». Це означає наступне: адміністратор, який хоче увійти в адмін панель веб-сайту, обов’язково повинен увести свій логін та пароль.

Відношення «extend» відображає можливе приєднання одного варіанту використання до іншого. Також для приєднуваного варіанту використання можна вказати умови його виконання.

Варіант використання «Додавання нового БПЛА» включає в себе наступні варіанти використання:

— «Введення параметрів та характеристик БПЛА». Введення таких параметрів, як назва БПЛА, його модель, опис, тип двигуна і тд. Повний список параметрів описано в розділах 4.3.2 та 4.3.3.

— «Завантаження зображень БПЛА». Завантаження, з ПЕОМ користувача веб-сайту, зображень (картинок) для нового БПЛА.

— «Завантаження посилання на офіційне джерело БПЛА». Завантаження посилання на офіційне джерело БПЛА в глобальній мережі Інтернет. Наприклад, на компанію виробника чи сторінки БПЛА в Вікіпедії.

— «Попередній перегляд БПЛА». Попередній перегляд всіх введених параметрів та посилань, а також завантажених зображень. Здійснюється адміністратором веб-сайту перед збереженням БПЛА в БД сайту та його публікацією на сайті.

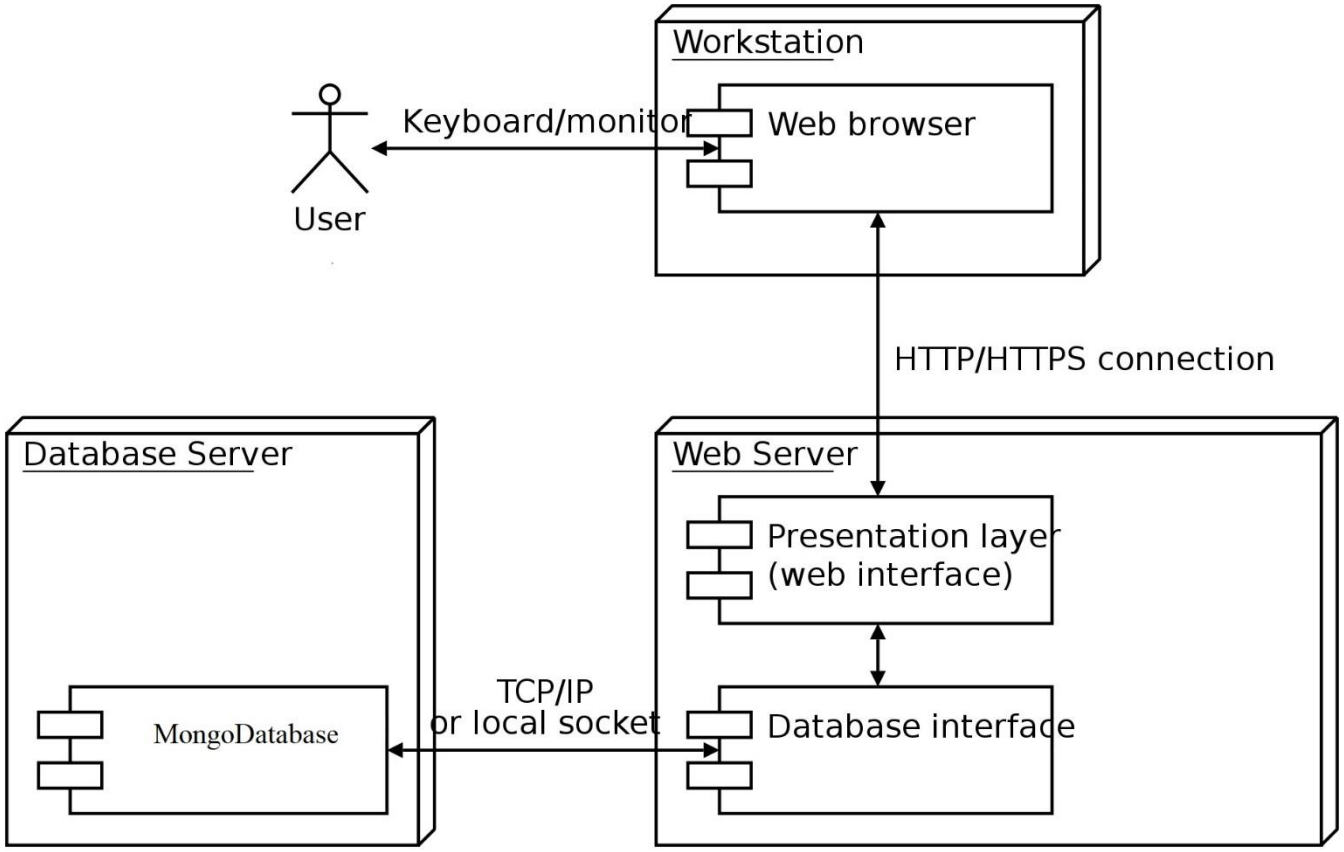
— «Підтвердження публікації нового БПЛА». Підтвердження публікації нового БПЛА на веб-сайті та запис його в БД даного сайту.

4.2.2. Діаграма розміщення

Діаграма розміщення – це тип діаграм в нотації UML, що відображає фізичні взаємозв'язки між програмними та апаратними компонентами (вузлами) системи та артефактами, що розгорнуті на них [11, 18]. Діаграми даного типу якнайкраще відображають схематичне зображення шляхів руху об’єктів та компонентів у розподіленій системі.

В даних типах діаграм, вузли – це прямокутні паралелепіпеди із артефактами, які знаходяться в них та зображені у вигляді прямокутників. Вузли – також можуть мати підвузли. Останні представлені вкладеними прямокутними паралелепіпедами. Вузол однієї діаграми може містити в собі від двох і більше вузлів. Це може бути кластер серверів, як приклад.

Діаграму розміщення для веб-сайту з пошуку та перегляду БПЛА із заданими властивостями було описано нижче (див. рис. 4.9). Дана діаграма була створена за допомогою десктопного програмного продукту Wondershare EdrawMax.



«Рис. 4.9. Діаграма розміщення для проектування веб-сайту з пошуку та перегляду БПЛА»

Діаграма розміщення для веб-сайту з пошуку та перегляду БПЛА із заданими властивостями описується наступним чином:

1. Користувач (клієнт) веб-сайту взаємодіє із системою шляхом використання сервісу з пошуку та перегляду БПЛА із заданими властивостями.
2. Веб-сайт відправляє запити та отримує відповіді від веб-сервера.
3. Сервер надає користувацький інтерфейс.
4. Сервер відправляє запити на сервер, на якому знаходить БД веб-сайту.
5. Сервер з БД відправляє відповідь на сервер, а сервер виводить дані про БПЛА на користувацький інтерфейс.

4.2.3. Діаграма компонентів

Діаграма компонентів – діаграма, на якій відображаються компоненти, залежності та зв'язки між ними них [11, 18]. Тобто, даний дип діаграм використовується для належного зображення взаємозалежності іж компонентами програмного забезпечення, включаючи компоненти вихідних кодів, бінарні

компоненти, та компоненти, що можуть виконуватись. Частина чи модуль програмного забезпечення може бути відображений на діаграмі як компонент. Також потрібно розуміти, що частина компонентів існує лише під час компіляції програмного коду, а інші – під час компонування коду. А деякі, взагалі, існують лише під час роботи (функціонування) програмного продукту.

Даний тип діаграм пояснює лише структурні характеристики. Для пояснення принципу роботи окремих екземплярів використовують діаграму розгортування (діаграму розміщення).

Для даного проекту було створено наступні діаграми компонентів:

— Загальну діаграму компонентів.

— Діаграму компонентів головної сторінки.

— Діаграму компонентів сторінки входу до адмін-сторінки.

— Діаграму компонентів сторінки для створення нового запису про БПЛА.

— Діаграму компонентів сторінки із детальної інформацією про конкретний БПЛА.

Діаграми компонентів для веб-сайту з пошуку та перегляду БПЛА із заданими властивостями було описано нижче. Дані діаграми було створена за допомогою десктопного програмного продукту Wondershare EdrawMax.

Спочатку потрібно описати загальну діаграму компонентів, що пояснює принцип роботи всіх сторінок веб-сайту та всього його функціоналу. Графічне відображення даної схеми знаходиться нижче (див. рис. 4.10). Як бачимо, дана діаграма складається з чотирьох частин:

— «User». Це особа, що використовує функціонал даного сайту. Може бути як звичайним користувачем, так і адміністратором. Тобто, два різні користувачі, які використовують лише свій тип функціоналу.

— «Frontend (client)». Це частина з компонентами, що відповідає за відображення клієнтської частини сайту. Тобто – це та інформація, що відображається користувачу чи адміністратору у веб-браузері. Більш детально функціонал кожного

із компонентів цієї частини – описано в розділах 4.3.2 та 4.3.3. Дана частина складається з наступних компонентів:

1. «MainPage». Головна сторінка веб-сайту (компонент) з пошуку та перегляду БПЛА із заданими властивостями. На ній відображено всі БПЛА, що наявні в БД сайту, а також функціонал з пошуку, перегляду, сортування та відображення БПЛ..

2. «DatailPage». Сторінка (компонент) з відображенням детальної інформації про конкретний БПЛА, що був вибраний для перегляду користувачем веб-сайту. На даній сторінці відбувається відображення параметрів БПЛА, його характеристик, зображень та посилань на офіційні джерела в глобальній мережі Інтернет.

3. «CreatePage». Сторінка (компонент) для створення нового БПЛА (нового запису в БД веб-сайту) адміністратором даного сайту.

4. «AuthPage». Сторінка (компонент) для авторизації адміністратора в адмін сторінці сайту. Після авторизації відбувається перехід на сторінку для створення нового БПЛА.

— «Back-end (server)». Це частина з компонентами, що відповідає за відображення серверної частини сайту, яка розташована на відповідному сервері в глобальній мережі Інтернет. Тобто – це функції, що знаходяться на веб-сервері. Вони обробляють запити клієнтської частини сайту (запити користувачів та адміністраторів сайту) та видають результати клієнту. Ця частина складається з наступних компонентів:

1. «Bplas». Компонент, що відповідає за обробку запиту клієнта (користувача сайту) на отримання сторінки конкретного БПЛА, а також, як результат виводить цю сторінку.

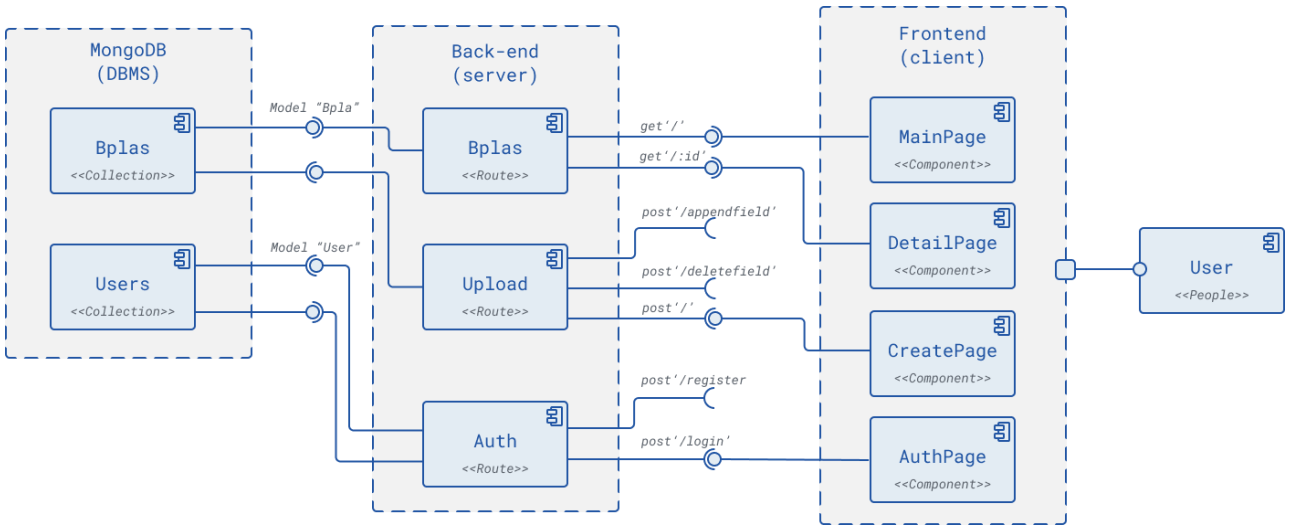
2. «Upload». Компонент, що відповідає за обробку запиту клієнта (адміністратора сайту) на створення сторінки для нового БПЛА та запису інформації про нього в БД сайту.

3. «Auth». Компонент, що відповідає за обробку запиту клієнта (адміністратора сайту) на авторизацію на адмін сторінці сайту (сторінці створення нового БПЛА).

— «MongoDB (DBMS)» Це частина з компонентами, що відповідає за БД веб-сайту, що розташована на відповідному сервері для БД, що знаходиться в глобальній мережі Інтернет. Тобто, це записи в БД даного сайту, які взаємодіють із «Back-end (server)» частиною та видають результати «Frontend (client)» частині сайту. Дана частина складається з наступних компонентів:

1. «Bplas». Таблиця (компонент) БД сайту, в якій зберігаються дані (параметри) кожного БПЛА.

2. «Users». Таблиця (компонент) БД сайту, в якій зберігаються облікові дані адміністратора сайту, тобто його логін та пароль.



«Рис. 4.10. Загальна діаграма компонентів для проектування веб-сайту з пошуку та перегляду БПЛА із заданими властивостями»

Тобто, принцип роботи компонентів, що «зображені на рис. 4.10», наступний:

1. Для роботи з головною сторінкою сайту для відображення всіх БПЛА потрібно виконати наступні дії:

1.1. Компонент «User» (звичайний користувач веб сайту) завантажує компонент «MainPage» (для відображення головної сторінки сайту зі всіма БПЛА).

1.2. Компонент «MainPage» робить запит на сервер сайту, до компоненту «Bplas» (файл чи окрема функція файлу на сервері, що відповідає за обробку запитів

клієнта, що пов’язані з відображенням БПЛА на сайті та отриманням інформації про них).

1.3. Компонент «Bplas» здійснює запит на сервер БД, до компоненту «Bplas», що відповідає за таблицю БД з записаними в них БПЛА. Це може бути як звернення одного файлу до іншого, так і однієї функції до іншої в одному і тому ж файлі.

1.4. Компонент «Bplas», що відповідає за таблицю БД із записаними в них БПЛА, надає відповідь у вигляді інформації (записів) про параметри всіх БПЛА. Відповідь надається компоненту «Bplas», що знаходиться в частині «Back-end (server)».

1.5. Компонент «Bplas», що знаходиться в частині «Back-end (server)», видає відповідь компоненту «MainPage» (частини «Frontend (client)») для відображення результатів.

1.6. Компонент «MainPage» видає користувачу веб-сайту головну сторінку даного сайту із завантаженими до нього карточками з всіма БПЛА. Тобто, дані про БПЛА – показуються не за допомогою записів в таблицях, а в зручному для користувача вигляді (з красивими та елегантними клавішами, формами, текстовими полями та посиланнями).

2. Для роботи з головною сторінкою із детальної інформацією про конкретний БПЛА – використовуються такі ж компоненти, що і в пункті 1. Різниця лише в тому, що компонент «Bplas» , який знаходиться в частині «Back-end (server)», буде видавати відповідь компоненту «DetailPage», що відповідає за відображення інформації про конкретний БПЛА. А вже компонент DetailPage» – буде виводити дані про конкретний БПЛА в зручному для користувача вигляді (з красивими та елегантними клавішами, формами, текстовими полями та посиланнями).

3. Для роботи зі сторінкою входу до адмін-сторінки потрібно виконати наступні дії:

3.1. Компонент «User» (адміністратор веб сайту) завантажує компонент «AuthPage» (для відображення сторінки авторизації).

3.2. Компонент «AuthPage» робить запит на сервер сайту, до компоненту «Auth» (файл чи окрема функція файлу на сервері, що відповідає за обробку запитів

адміністратора, що пов’язані з авторизацією). В цьому запиті передаються облікові дані адміністратора, тобто його логін та пароль в системі.

3.3. Компонент «Auth» здійснює запит на сервер БД, до компоненту «Users», що відповідає за таблицю БД з записаними в них обліковими даними адміністратора.

3.4. Компонент «Users», що відповідає за таблицю БД із записаними в них обліковими даними адміністратора, надає відповідь у вигляді інформації (записів) про дані облікові записи. Відповідь надається компоненту «Auth», що знаходиться в частині «Back-end (server)».

3.5. Компонент «Auth», що знаходиться в частині «Back-end (server)», видає відповідь компоненту «AuthPage» (частини «Frontend (client)») для авторизації адміністратора.

3.6. Компонент «Auth» видає адміністратору доступ до сторінки (компоненту) «CreatePage» (компоненту-сторінки, що відповідає за створення нового БПЛА). Цей етап відбувається при умові уведення адміністратором своїх вірних облікових даних, які збігаються з тими, що записані в БД сайту, в таблиці «Users». В іншому випадку – пункт 3.6 не спрацьовує і виводиться повідомлення про помилку авторизації.

4. Для роботи зі сторінкою для створення нового запису про БПЛА потрібно виконати наступні дії:

4.1. Компонент «User» (адміністратор веб сайту) завантажує компонент «CreatePage» (для відображення сторінки створення нового БПЛА).

4.2. Компонент «CreatePage» робить запит на сервер сайту, до компоненту «Upload» (файл чи окрема функція файлу на сервері, що відповідає за обробку запитів адміністратора, що пов’язані зі створення запису про новий БПЛА). В цьому запиті передаються параметри (характеристики), зображення БПЛА та посилання на нього в глобальній мережі Інтернет. Більш детально ці параметри описані в розділах 4.3.2 та 4.3.3.

4.3. Компонент «Upload» здійснює запит на сервер БД, до компоненту «Bplas», що відповідає за таблицю БД з записаними в них БПЛА. Це запит на запис нового БПЛА до БД сайту, в таблицю «Bpla».

4.4. Компонент «Bplas», що відповідає за таблицю БД з записаними в них БПЛА, видає відповідь про успішний запис нового БПЛА компоненту «Bplas», що знаходиться в частині «Back-end (server)».

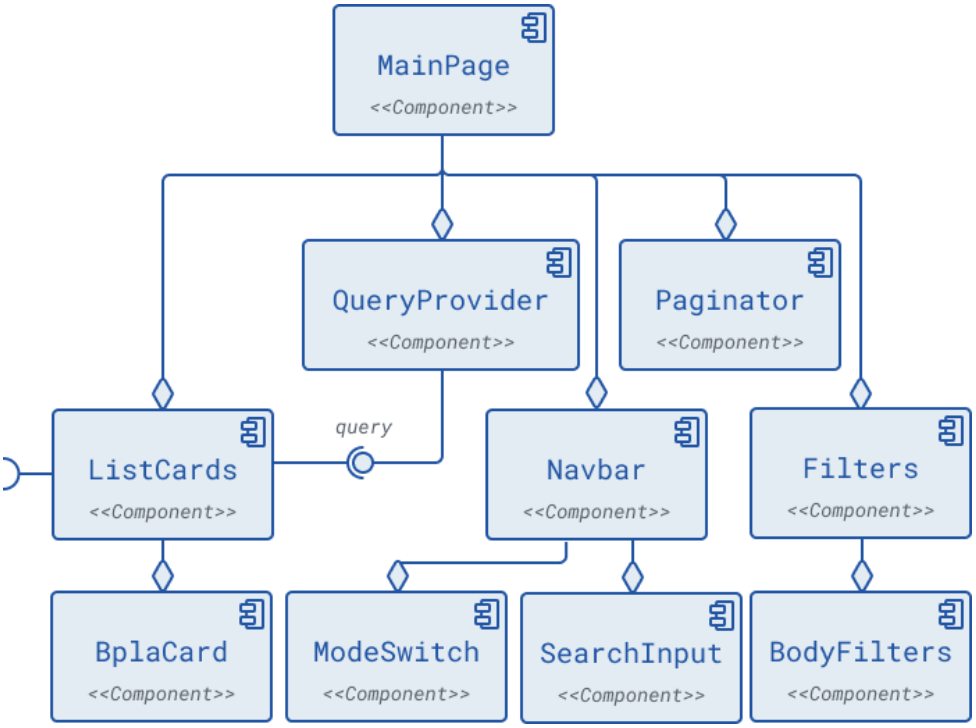
4.5. Компонент «Bplas», що знаходиться в частині «Back-end (server)», видає відповідь компоненту «CreatePage» – про успішний запис нового БПЛА в БД сайту. Також «Bplas» надає відповіді компонентам «MainPage» та «DetailPage» – для відображення нового БПЛА на головній сторінці сайту, а також для створення окремої сторінки сайту для створеного БПЛА.

4.6. Компонент «CreatePage» видає відповідь компоненту «User» (адміністратору сайту). Ця відповідь містить повідомлення про успішний запис нового БПЛА до БД сайту.

В наступних підрозділах буде описано діаграми компонентів кожної сторінки веб-сайту, що проектується.

4.2.3.1. Діаграма компонентів головної сторінки

Діаграма компонентів головної сторінки веб-сайту з пошуку та перегляду БПЛА із заданими властивостями зображено на наступній схемі (див. рис. 4.11).



«Рис. 4.11. Діаграма компонентів головної сторінки веб-сайту, що проектується»

В даній діаграмі головним компонентом є «MainPage» – відповідає за відображення в веб-браузері користувача головної сторінки зі всіма наявними БПЛА. Дані БПЛА записані в БД сайту.

Компонент «MainPage» використовує компонент «QueryProvider», що відповідає за запит до БД сайту для відображення БПЛА. Далі, компонент «QueryProvider» здійснює запит «query» для отримання даних про БПЛА, та виводить їх у компонент «ListCards», який відповідає за відображення на головній сторінці карточок зі всіма БПЛА. А компонент «ListCards» – містить в собі компоненти «BplaCard», кожен з яких відповідає за окремий БПЛА, що розташований в карточці.

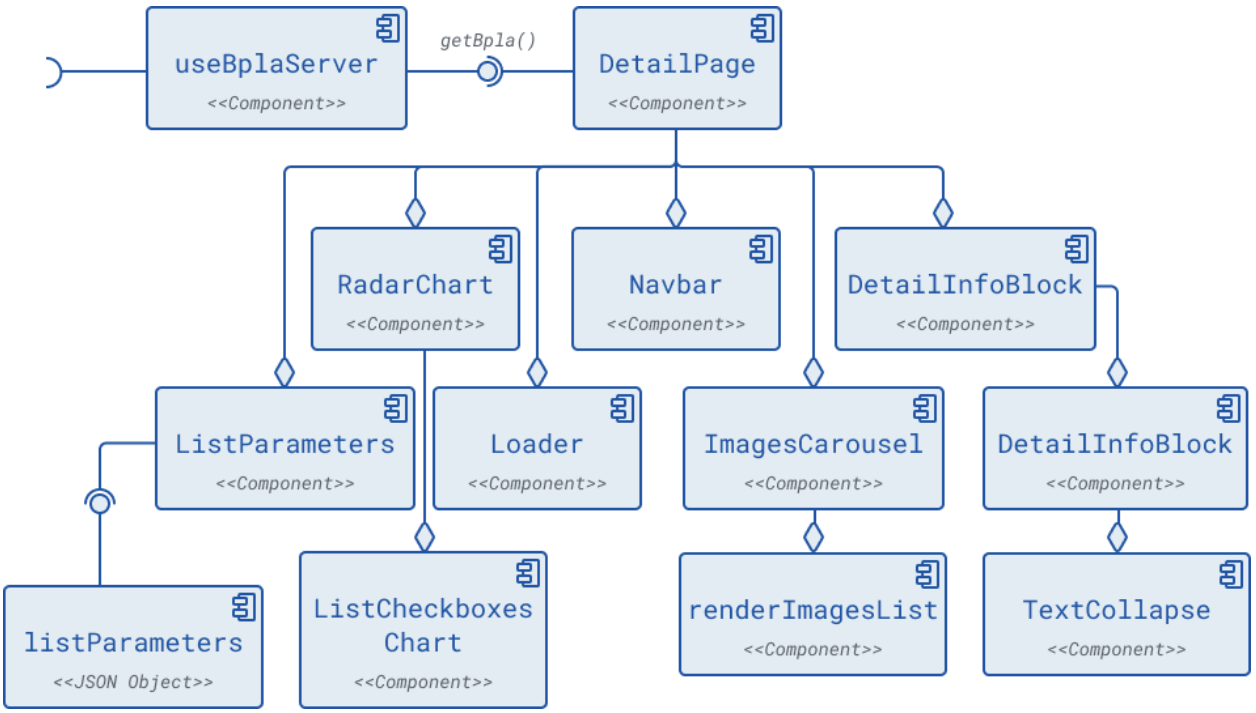
Також на головній сторінці присутній компонент «Paginator», що відповідає за кількість БПЛА, що виводяться на одну сторінку, а також за номер сторінки з БПЛА.

Компонент «Filters», що завантажується через компонент «MainPage», відповідає за фільтрацію БПЛА за певними критеріями (наприклад, за типом двигуна чи країною виробником). Для цього на сайті буде створено список фільтрів, в якому є певна кількість критеріїв. Ці критерії знаходяться в компоненті «BodyFilter», який спрацьовує після компоненту «Filters».

Компонент «Navbar» відповідає за функцію сортування виведених БПЛА та спрацьовує після завантаження компоненту «MainPage». Сортування відбувається за певним критерієм – в порядку його збільшення чи зменшення. За вибір критерію відповідає компонент «SearchInput», а за порядок сортування – «ModeSwitch». Наприклад, сортування за алфавітом в порядку зростання, від першої до останньої букви.

4.2.3.2. Діаграма компонентів сторінки конкретного БПЛА

Діаграма компонентів головної сторінки перегляду конкретного БПЛА зображено на наступній схемі (див. рис. 4.12).



«Рис. 4.12. Діаграма компонентів сторінки перегляду конкретного БПЛА веб-сайту, що проектується»

В даній діаграмі головним компонентом є «DetailPage» – відповідає за відображення в веб-браузері користувача сторінки з конкретним вибраним БПЛА. Даний БПЛА записаний в БД сайту. Цей компонент відбувається внаслідок переходу з головної сторінки сайту на сторінку з конкретним БПЛА. Даний механізм забезпечує компонент «useBplaServer», що спрацьовує на веб-сервері та передає результат спрацювання клієнту. Результат відобразиться у веб-браузері у вигляді сторінки.

Також присутній компонент «Navbar», функції якого були описані в розділі 4.2.3.1.

Компонент «ImagesCarousel» відповідає за виведення списку картинок (зображень) конкретного БПЛА. В цьому списку є одна чи декілька таких картинок, за кожну із яких відповідає компонент «renderImagesList».

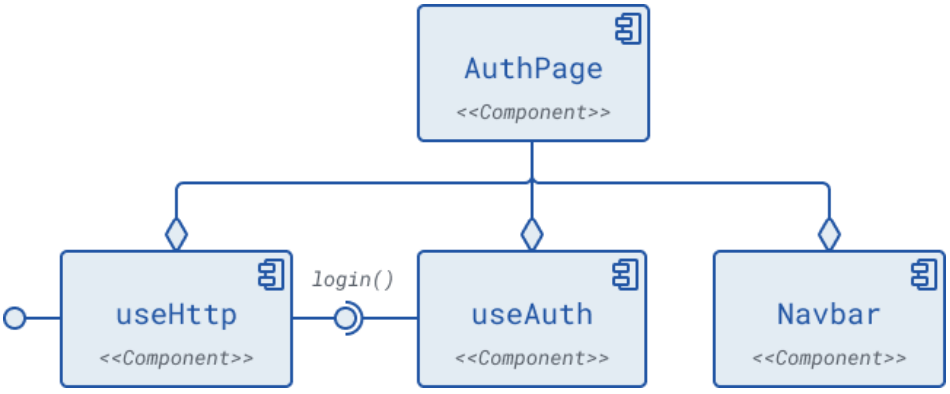
Компонент «DetailInfoBlock» відповідає за форму (блок), в якому будуть відображатися вся текстова інформація про БПЛА, а сам текст – відповідає компонент «TextCollapse».

Компонент «RadarChart» відповідає за додаткову функцію сайту – графічного виведення числових значень (параметрів) БПЛА. Для цього дані параметри потрібно передати до вищеописаного компоненту. За передачу кожного параметру відповідає компонент «ListCheckBoxesChart».

Компонент «ListParameters» відповідає за вивід на сторінку конкретного БПЛА саме числових параметрів. Наприклад, дальність польоту (км) чи розмах крил (м). Для цього їх потрібно завантажити з БД сайту. За процес завантаження відповідає компонент «listParameters».

4.2.3.3. Діаграма компонентів сторінки авторизації адміністратора

Діаграма компонентів сторінки авторизації адміністратора зображено на наступній схемі (див. рис. 4.13).



«Рис. 4.13. Діаграма компонентів сторінки авторизації адміністратора веб-сайту, що проектується»

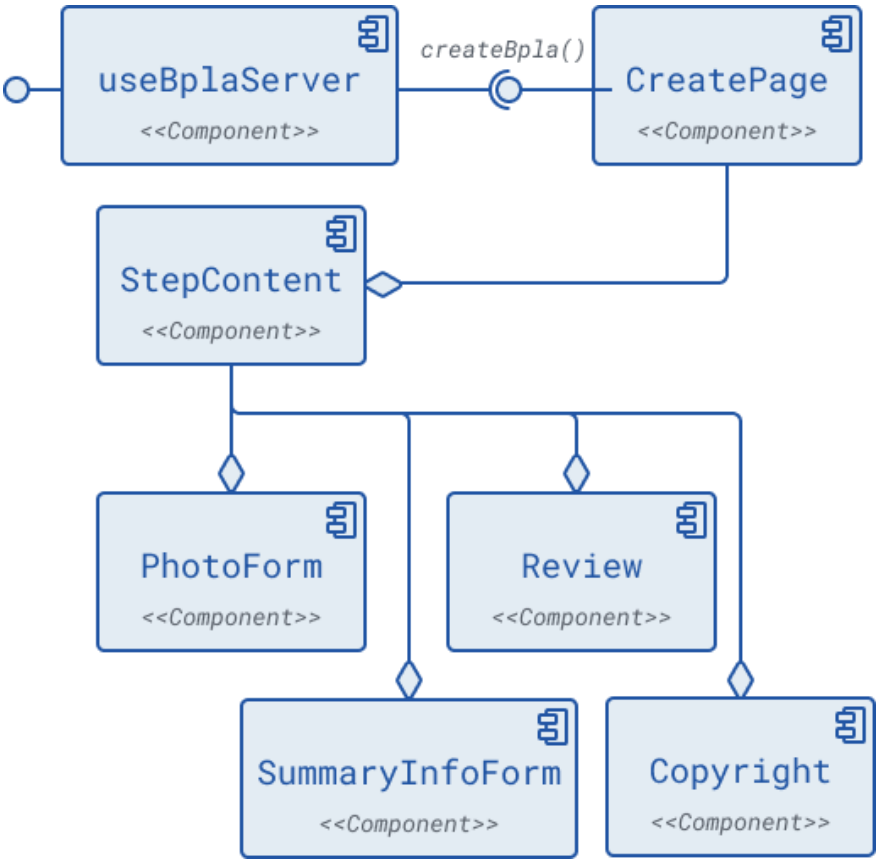
В даній діаграмі головним компонентом є «AuthPage» – відповідає за авторизацію в адмін панелі. Дана адмін-панель – це сторінка в якій відбувається створення та додавання в БД сайту нового БПЛА. Процес авторизації в адмін панелі може здійснювати тільки адміністратор сайту.

Для належної роботи функціоналу даної сторінки потрібно, щоб компонент «useHttp», який відповідає за http-запит на веб-сервер, передав на цей сервер облікові дані адміністратора. Цими обліковими даними є логін та пароль. Після отримання сервером цих даних – спрацьовує компонент «useAuth», який використовує логін та пароль адміністратора для його успішної авторизації в адмін-панелі.

Також присутній компонент «Navbar», функції якого були описані в розділі 4.2.3.1.

4.2.3.4. Діаграма компонентів сторінки створення нового БПЛА

Діаграма компонентів сторінки створення нового БПЛА зображено на наступній схемі (див. рис. 4.14).



«Рис. 4.14. Діаграма компонентів сторінки створення нового БПЛА веб-сайту, що проектується»

В даній діаграмі головним компонентом є «CreatePage» – створення нового БПЛА та додавання інформації про нього в БД сайту. Процес авторизації в адмін панелі може здійснювати тільки адміністратор сайту. Цей компонент відбувається внаслідок переходу зі сторінки авторизації на сторінку створення нового БПЛА. Даний механізм забезпечує компонент «useBplaServer», що спрацьовує на веб-сервері та передає результат спрацювання клієнту. Результат відобразиться у веб-браузері у вигляді сторінки.

Компонент «StepContent» відповідає за заповнення та зчитування текстової та числової інформації про параметри БПЛА. Також даний компонент передає дану

інформацію на опрацювання наступному компоненту – «PhotoForm», який відповідає за завантаження картинок (зображень) нового БПЛА. Далі – спрацьовує компонент «Review», що відповідає за попередній огляд введених параметрів та зображень БПЛА. Тобто, адміністратор зможе глянути, як саме буде виглядати ці дані на сторінці БПЛА, а також переконається в наявності всіх параметрів.

Компонент «SummaryInfoForm» відповідає за вивід на екран (у вигляді красивої графічної форми) повідомлення про успішне створення БПЛА та запис інформації про нього до БД сайту.

Компонент «Copyright» відповідає за копіювання всіх введених даних про БПЛА з форми для введення до БД сайту.

4.3. Розробка програмного забезпечення

Для створення програмного коду веб-сайту (веб-сервісу) з пошуку та перегляду БПЛА із заданими властивостями було використано середовище розробки Microsoft Visual Studio Code версії 1.77.3. Також було використано браузер Chrome для перегляду та тестування даного веб-сайту.

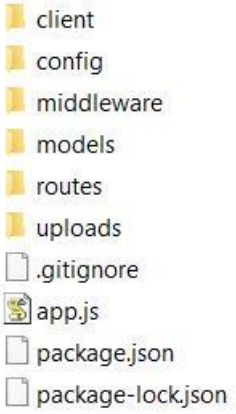
4.3.1. Файлова структура проекту

Для початку потрібно описати файлову структуру веб-сайту з пошуку та перегляду БПЛА із заданими властивостями.

Кореневий каталог веб-сайту має назву «\bpla-main» (див. рис. 4.15). В кореневому каталозі веб-сайту знаходяться наступні каталоги та файли:

1) каталог «\client».
2) каталог «\config».
3) каталог «\middleware».
4) каталог «\models».
5) каталог «\routes».
6) каталог «\uploads».
7) файл «.gitignore».
8) файд «app.js».
9) файл «package.json».

10) файл «package-lock.json».



«Рис. 4.15. Файлова структура кореневого каталогу проекта»

Як було описано раніше – даний проект працює за архітектурою типу «клієнт-сервер». Тобто, Користувач веб-сайту завантажує даний екземпляр сайту в своєму браузері (як клієнтський додаток), а вже цей клієнт (браузер) відправляє запити на веб-сервер, де здійснюється обробка серверної частини проекту. В результаті – сервер видає відповідь клієнту у вигляді завантаженої сторінки в браузері .

Тобто, даний проекті присутні окремо каталоги з файлами, що відповідають за:

– front-end (клієнтську) частину сайту;

– back-end (серверну) частину сайту.

Спочатку опишемо каталоги та файли, що відповідають за back-end (серверну) частину сайту. Структура даних каталогів та файлів наступна:

1. «\config». Каталог, в якому знаходиться конфігураційні файли, що відповідають за роботу серверу, на основі якого працюватиме сайт (див. рис. 4.16). В даному каталозі знаходяться наступні файли:

1.1. «Default.json» – це файл, що відповідає за конфігураційні дані для завантаження та роботи серверу в режимі відлагодження сайту.

1.2. «Production.json» – це файл, що відповідає за конфігураційні дані для завантаження та роботи серверу в режимі «готової продукції», тобто в коли сайт знаходиться в продакшені.

2. «\middleware». Каталог, в якому знаходяться файли, що відповідають за роботу middleware – проміжного програмного забезпечення, що складається з агентів,

які є посередниками між різними компонентами додатків, зазвичай використовується в розподілених застосунках (див. рис. 4.17). В даному каталозі знаходяться наступні файли:

2.1. «Auth.middleware.js» – це файл, що відповідає за middleware для перехоплення токену та виявлення, чи є клієнт, що прислав запит до сервера, зареєстрованим на веб-сайті.

2.2. «Cors.middleware.js» – це файл, що відповідає за middleware, який вказує хедери (заголовки) для запитів, що приходять.

3. «\models». Каталог, в якому знаходяться файли, що відповідають за роботу та обробку моделей даних (див. рис. 4.18). В даному каталозі знаходяться наступні файли:

3.1. «Bpla.js» – це файл, що описує принцип роботи та обробки моделі даних БПЛА та взаємодію з БД.

3.2. «User.js» – це файл, що описує принцип роботи та обробки моделі даних користувача веб-сайту та взаємодію з БД.

4. «\routes». Каталог, в якому знаходяться файли, що відповідають за визначення обробника для конкретної запитуваної сторінки (див. рис. 4.19). В даному каталозі знаходяться наступні файли:

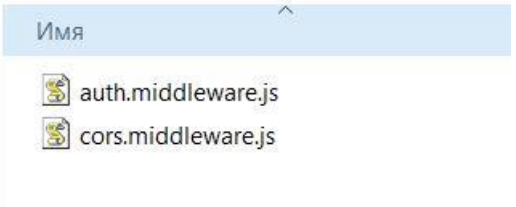
4.1. «Auth.routes.js» – це файл, що описує всі функції, що пов’язані з входом та реєстрацією користувачів веб-сайту. Тут відбувається перевірка надходження даних та вся логіка, що пов’язана даними користувача, а саме – записування даних БД та зчитування з неї.

4.2. «Bpla.routes.js» – це файл, що описує всі функції, що пов’язані з отриманням даних про БПЛА. Тут відбувається перевірка надходження даних та вся логіка, що пов’язана даними про БПЛА, а саме – записування даних БД та зчитування з неї

4.3. «Upload.routes.js» – це файл, що описує всі функції, що пов’язані з записом нових даних про БПЛА, видаленням та додаванням нових полів у модель БПЛА. Також, тут відбувається перевірка надхоження даних та вся логіка, що пов’язана з записом даних про БПЛА до БД.

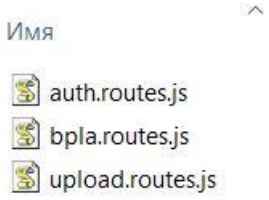
5. «App.js» – це файл, що описує всі функції з налаштування та підключення endpoints (кінцевих пристроїв, що захищаються, наприклад комп'ютер з антивірусом), middlewares. Також даний файл відповідає за роботу самого серверу та включення його в роботу.

«Рис. 4.16. Файлова структура каталогу «\config»»



«Рис. 4.17. Файлова структура каталогу «\middleware»»

«Рис. 4.18. Файлова структура каталогу «\models»»



«Рис. 4.19. Файлова структура каталогу «\routes»»

Далі опишемо каталоги та файли, що відповідають за front-end (клієнтську) частину сайту. Програмний код кожного із файлів зображено на відповідних рисунках. Структура даних каталогів та файлів наступна:

1) «\components». Каталог, в якому знаходиться всі файли (компоненти), що відповідають за роботу клієнтської сторони сайту (див. рис. 4.20). В даному каталозі знаходяться наступні файли:

a. «BodyFilters.js» – компонент відображення тіла-списка фільтрів БПЛА.

b. «BplaCard.js» – компонент відображення карточки із коротким описом БПЛА на головній сторінці.

c. «DetailInfoBlock.js» – компонент-блок відображення головної інформації про БПЛА на сторінці з детальною інформацією про літак.

d. «DoubleSlider.js» – компонент подвійного слайдера для встановлення мінірисьного та максирисьного значення.

e. «Filters.js» – компонент-провайдер для різного відображення фільтрів на головній сторінці.

f. «FilterTextInput.js» – компонент для введення текстових даних. Для панелі із фільтрами на головній сторінці.

g. «ImagesCarousel.js» – компонент для відображення декількох фотографій із можливістю їх «свайпити».

h. «ListCards.js» – компонент для завантаження даних із сервера через відповідний інший компонент і відображення списка цих даних у дочірніх компонентах BplaCard.

i. «ListCheckboxes.js» – компонент для відображення списку чекбоксів одної группи.

j. «ListCheckboxesCart.js» – компонент для відображення списку чекбоксів для налаштування графіка числових значень про БПЛА.

k. «ListParametes.js» – компонент для відображення списку даних про БПЛА.

l. «Loader.js» – компонент для відображення блоку завантаження.

m. «ModeSwitch.js» – компонент-перемикач теми відображення сайту (світлий\темний).

n. «MultipleSelect.js» – компонент для вибору декількох одразу даних з наданого списку.

o. Navbar.js – компонент для відображення верхньої панелі навігації, пошуку та іншого.

p. «NumberSlider.js» – компонент-слайдер для вибору одного числового значення.

q. «Paginator.js» – компонент для відображення та вибору сторінки та кількості відображених елементів на одній сторінці.

r. «PhotoForm.js» – компонент-форма для вибору декількох фото при створенні нового запису у БД про БПЛА. Використовує дочірній компонент UploadImages.

s. «RadarChart.js» – компонент для відображення діаграми числових даних про БПЛА у вигляді «радару».

t. «Review.js» – компонет-форма для відображення всіх уведених значень про новий БПЛА. Потрібен для перевірки уведених даних.

u. «SearchInput.js» – компонент поля для введення строки для пошуку.

v. «SelectInput.js» – компонент-поле для вибору одного значення із списку запропонованих.

w. «Sort.js» – компонент для вибору типу сортування по сайту.

x. «SummaryInfoForm.js» – компонент-форма для введення тектових і числових значень для нового запису про БПЛА.

y. «TextCollapse.js» – компонент для виведення тексту, який, при відображенні досить великого тексту, скриває частину тексту.

z. «TextInput.js» – компонент для введення тектових даних.

aa. «UploadImages.js» – компонент для відображення інтерфейсу завантаження фото для нового БПЛА.

2) «\context». Каталог, в якому знаходиться всі файли (компоненти), що відповідають за контексти сайту (див. рис. 4.21). В даному каталозі знаходяться наступні файли:

a. «configContext.js» – створює конфігураційний контекст для клієнта: зміна теми сайту.

b. «Context.js» – створює authContext для надання і змінення даних про вхід до акаунта на сайті.

c. «formContext.js» – створює контект даних для форм при заповнені даних про новий БПЛА.

d. «listParameters.js» – надає масив об’єктів з інформацією про те, які фільтри має сайт і які їх налаштування. Потрібно для відображення списків фільтрів і подіного.

e. «queryContext.js» – створює контекст для змінення і читання поточного запиту філтрів на сайті. Ці дані зберігаються у адресному полі браузера.

3) «\hooks». Каталог, в якому знаходиться всі файли (компоненти), що відповідають за хук-функції сайту (див. рис. 4.22). В даному каталозі знаходяться наступні файли:

a. «Auth.hook.js» – хук-функція, яка надає можливість увійти до акаунта у системі та вийти з нього. Зберігає і читає дані для входу у браузері клієнта.

b. «Bpla.hook.js» – хук-функція, яка надає можливість надіслати запист до сервера для отрпимання даних про БЛПА.

c. «formData.hook.js» – хук-функція, яка надає провайдер із відповідним «станом» для даних контекта-форми.

d. «http.hook.js» – хук-функція, яка надає можливість надсилати різні запити до сервера з вказаними параметрами і даними.

e. «queryBuilder.hook.js» – хук-функція, яка надає можливість праці із запитом фільтрації до сервера та створює відповідний контекст даних.

f. «Theme.hook.js» – хук-функція, для завантаження стану теми з браузера та встановлення її як поточна.

4) «\pages». Каталог, в якому знаходиться всі файли (компоненти), що відповідають за роботу сторінок сайту (див. рис. 4.23). В даному каталозі знаходяться наступні файли:

a. «AuthPage.js» – компонент-сторінка входу до акаунта адміна.

b. «CreatePage.js» – компонент-сторінка створення нового запису про БПЛА.

c. «DetailPage.js» – компонент-сторінка відображення детальної інформації про конкретний БЛПА.

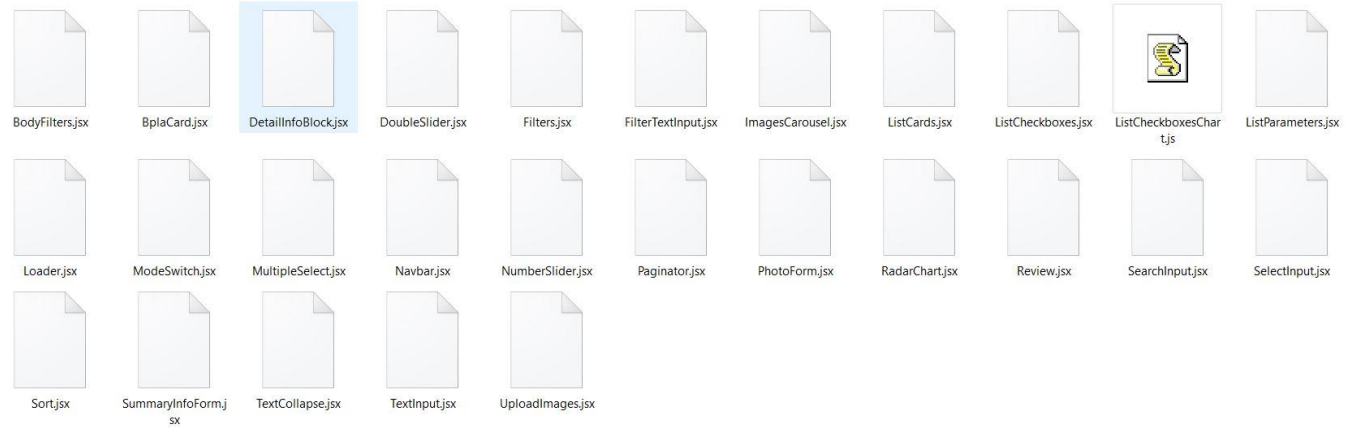
d. «MainPage.js» – компонент-сторінка відображення почтакової і головної сторінки з даними, фільтрами та іншим.

5) «App.js» – компонент, де налаштовуються всі провайдери та роути. Огортується контектами і містить рендер сторінок.

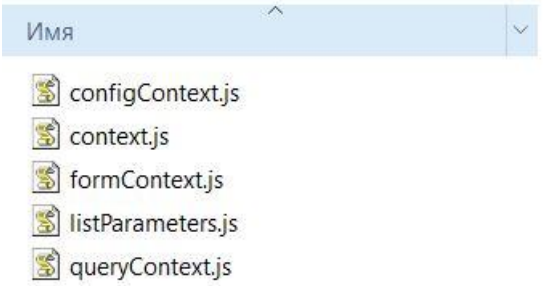
6) «Axios.common.js» – створюються базові налаштування для бібліотеки axios.

7) «Index.js» – точка входу для javascript комопнентів і всього коду.

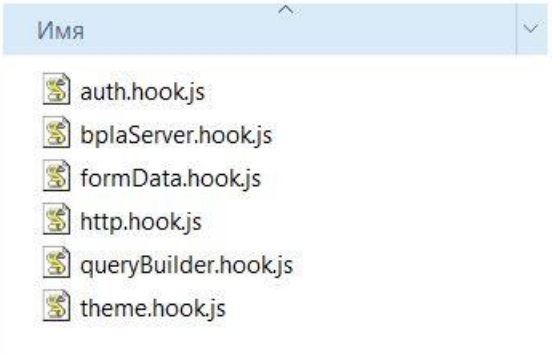
8) «Routes.js» – відображаються за певними умовами та чи інша компонент-сторінка.



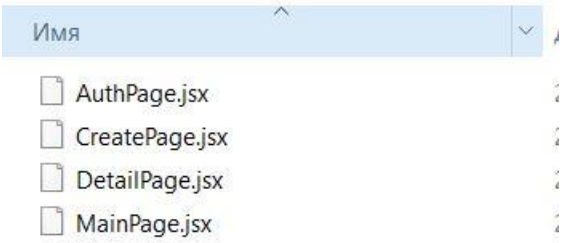
«Рис. 4.20. Файлова структура каталогу «\components»»



«Рис. 4.21. Файлова структура каталогу «\context»»



«Рис. 4.22. Файлова структура каталогу «\hooks»»



«Рис. 4.23. Файлова структура каталогу «\pages»»

4.3.2. Загальний опис функціоналу проекту

В процесі розробки веб-сайту було реалізовано функціонал, який відповідає вимогам, що були описані в розділі 3. Інструкція користування даним веб-сайтом буде описана в розділі 5.

Відповідно до описаного в розділах 4.1.3 та 4.1.4 стає зрозуміло що – веб-сайт з пошуку та перегляду БПЛА із заданими властивостями складатиметься з наступних чотирьох сторінок:

1. Головна сторінка веб-сайту, де здійснюється відображення всіх наявних для пошуку БПЛА.
2. Сторінка опису обраного БПЛА – на даній сторінці відбувається детальний опис конкретно вибраного БПЛА користувачем сайту.
3. Сторінка «Вхід до акаунта адміна» (вхід до адмін сторінки веб-сайту).
4. Сторінка «Додати новий БПЛА до бази» (адмін сторінка для створення нових записів).

Розглянемо функціонал головної сторінки веб-сайту з пошуку та перегляду БПЛА із заданими властивостями.

Як було вказано вище, за допомогою головної сторінки користувач веб-сайту може здійснити пошук потрібного йому БПЛА за певними параметрами. Пошук може здійснюватися двома способами – за допомогою функції «Пошуковий рядок» або ж за допомогою функції «Пошук за фільтрами».

Пошук за допомогою функції «Пошуковий рядок» здійснюється за рахунок порівняння рядку, з введеним текстом, із наявним рядком чи частиною рядку, що знаходиться в таблиці БД, яка відповідає за інформацію про БПЛА. А конкретно – пошук здійснюється по назві, моделі, короткому або ж повному опису БПЛА.

Для пошуку певного БПЛА за допомогою функції «Пошуковий рядок» потрібно виконати наступні дії:

1. В текстовому полі «Пошуковий рядок» ввести рядок з текстовою інформацією, за якою здійснюється пошук.
2. Натиснути клавішу «Enter» чи клавішу «Лупи».
3. З головної сторінки пропаде список всіх наявних БПЛА, і буде виведено лише ті, що задовольняють параметрам пошуку.
4. Якщо потрібно відмінити пошук за певним рядком – то видаляємо його або ж натискаємо на клавішу «Х», яка відповідає за видалення рядку пошуку та знаходиться в пошуковому рядку.

Друга корисна функція, що знаходиться в лівій частині (лівому сайдбарі) на головній сторінці сайту, це «Пошук за фільтрами». За допомогою цієї функції користувач має можливість відфільтрувати пошук за певними параметрами та вивести результат пошуку на екран. Даними параметрами пошуку (фільтрації) є:

— виробник БПЛА;
— країна виробник БПЛА;
— тип двигуна;
— наявні функції;
— рівень застосування;
— рівень воєнних дій (в яких може застосовуватися БПЛА);
— клас БПЛА;
— дальність польоту (в кілометрах);
— розмах крил;
— максимальна злітна маса;
— корисне навантаження;
— максимальна швидкість;
— крейсерна швидкість;
— максимальна висота польоту;
— операційна висота використання;

— тривалість польоту.

Для пошуку певного БПЛА за допомогою функції «Пошук за фільтрами» потрібно виконати наступні дії:

1. Вибрати потрібні параметри фільтрів.
2. Натиснути клавішу «Застосувати фільтри». Тільки після натиску на цю кнопку будуть збережені обрані до цього фільтри у url-рядку та здійснений пошук за таблицею БД, в якій зберігаєтьс інформація про БПЛА.
3. Почекати декілька секунд поки оновиться інформація.
4. Всі наявні на головній сторінці БПЛА пропадуть і замість них з’являться лише ті, що відповідають заданим параметрам фільтрів.
5. Якщо користувач бажає здійснити пошук за новими (іншими) фільтрами – потрібно заново виконати дії пунктів 1-4, попередньо видаливши старі фільтри за допомогою клавіші «Скинути фільтри» Дана клавіша скидує всі обрані фільтри до значень за змовчуванням і відбувається пошук та відображення даних із БД без фільтрів.

Крім клавіш «Застосувати фільтри» та «Скинути фільтри» у функції «Пошук за фільтрами» присутні текстові поля (введення і точного пошуку за відповідним параметром), списки «checkbox» (надають можливість обрати декілька конкретних варіантів, які мають, як характеристику, БПЛА) та діапазони чисел (це числові межі, що визначають мінірисьне і максирисьне значення для фільтрації БПЛА за відповідним числовим параметром).

Також на головній сторінці сайту присутня функція «Перемикач «теми» сайту». Дана функція, зазвичай» присутня на багатьох сайтах електронної комерції. Дана функція дозволяє вибрати тему (фон) сайту – в світлих чи темних тонах. Важливою особливістю даного сайту є те, що вибір теми запам’ятовується у браузері конкретного клієнта. Тобто, при перезавантажені сторінки це налаштування зберігається.

Ще однією особливість, що присутня на сайті – наявність функції «Сортування БПЛА». Дана функція надає можливість сортувати БПЛА за певним типом (параметром) сортування. Для цього потрібно вибрати потрібний користувачу тип

сортування та натиснути на спеціальний перемикач, що відповідає за напрям сортування (в порядку збільшення або зменшення). Наприклад, якщо вибрати сортування за злітною масою в порядку збільшення – то на головну сторінку веб-сайту будуть виведені всі наявні в БД БПЛА (з використанням чи без використання фільтрів) та відсортовані в порядку збільшення їх злітної маси (від найменшої до найбільшої).

Також на головній сторінці веб-сайту присутній «Пагінатор». Цей елемент дозволяє вибрати кількість елементів БПЛА, які максирисьно можливо виводити на одну сторінку, та номер сторінки, на якій розміщені БПЛА.

Ще один із функціоналів головної сторінки веб-сайту – «Список елементів БПЛА». Це є адаптивна для різних екранів сітка із карточок. Кожна карточка відображає інформацію про БПЛА: назва, модель, короткий опис та перше фото.

Якщо не знайдено жодного БПЛА за обраними фільтрами чи за допомогою пошукового рядку, то відображається відповідний напис про відсутність БПЛА за заданими параметрами.

Розглянемо функціонал сторінки з детальним відображенням даних про БПЛА.

Для того щоб переглянути інформацію про певний конкретний (бажаний) БПЛА – потрібно перейти на сторінку з цим БПЛА. Для цього потрібно:

1. Вибрати потрібний БПЛА за певними характеристиками (принцип вибору описано раніше).
2. Натиснути на зображення потрібного БПЛА.
3. Почекати декілька секунд.
4. Здійснеться перехід на сторінку з вибраним БПЛА.

На сторінці із детальним відображенням даних про БпЛА є:

— Стрічка всіх наявних фото конкретного БпЛА, які можна гортати.

— Блок із загальною і найбільш важливою інформацією про БПЛА. На ньому відображається:

- назва БПЛА;
- модель БПЛА (зобаржено у дужках);

– короткий опис БПЛА (дрібним текстом);

– детальний опис БПЛА. Якщо опис тексту занадто великий, то він відображається тільки частково. Для перегляду всього опису тексту треба натиснути на клавішу «Детальніше», що знаходиться під ним;

– клавіша з посиланням на ресурс цього БПЛА, де про нього можливо або детально почитати, або купити. Це залежить від того, яке посилання було записано в БД.

— Список із іншими текстовими та числовими даними.

— Графік у вигляді радару із списком перемикачів, котрими моможливо обрати дані, що потрібно відображати на графіку. Якщо про БПЛА немає числових даних, то цей елемент не відображається.

Розглянемо функціонал сторінки «Вхід до акаунта адміна».

На даному сайті є можливість додавати інформацію про нові БПЛА. Цю функцію може використовувати лише адміністратор веб-сайту, який попередньо повиден авторизуватися через сторінку адміна. Для того, щоб зайти на дану сторінку потрібно в адресному рядку браузера користувача ввести домен сайту та ключове слово «login». Така url-адреса буде виглядати наступним чином – <domain_site>/login. В цілях захисту від недбалого знищення чи модифікації сайту – на дану сторінку не існує звичайного посилання.

На даній сторінці, сторінці «Вхід до акаунта адміна» присутні наступні елементи:

— Поле логіну у вигляді електронної пошти адміна веб-сайту.

— Поле для введення паролю адміна.

— Клавіша «Увійти», для підтвердження входу та переходу на сторінку.

Для переходу на сторінку з додаванням інформації про нові БПЛА вводимо наступні дані:

1. В полі логін вводимо електрону пошту адміністратора веб-сайту.
2. В полі пароль вводимо секретний пароль адміністратора веб-сайту.
3. Натискаємо на клавішу «Увійти».

4. При правильному виконанні дії, що описані в пунктах 1-3, буде здійснено успішний вхід на сторінку з додаванням інформації про нові БПЛА.

Розглянемо функціонал сторінки «Додати новий БПЛА до бази».

Сторінка «Додати новий БПЛА до бази» (сторінка адміна) відповідає за додавання інформації про новий БПЛА в БД веб-сайту. Нова створена сторінка з БПЛА буде відображатися на головній сторінці даного сайту.

Сторінка «Додати новий БПЛА до бази» поділена на три частини, в кожній з яких є форма для введення певної інформації чи параметрів про БПЛА. На першому етапі введення інформації присутня форма для введення текстових та числових даних, в ній присутні поля для введення:

— Назви БПЛА (у вигляді введення одного рядку).

— Моделі БПЛА (у вигляді введення одного рядку).

— Короткий опис БПЛА (у вигляді введення одного або декількох рядків).

— Детальний опис загальної інформації про БПЛА (у вигляді введення одного або декількох рядків).

— Посилання на джерело БПЛА (у вигляді введення одного рядку).

— Назви виробника БПЛА (у вигляді введення одного рядку).

— Країни виробника БПЛА (у вигляді вибору одного параметру із випадаючого списку).

— Тип двигуна БПЛА (у вигляді вибору одного параметру із випадаючого списку).

— Функції БПЛА (у вигляді вибору одного чи декількох параметрів із випадаючого списку).

— Рівень застосування БПЛА (у вигляді вибору одного чи декількох параметрів із випадаючого списку).

— Рівень воєнних дій БПЛА (у вигляді вибору одного параметру із випадаючого списку).

— Клас БПЛА (у вигляді вибору одного параметру із випадаючого списку).

— Дальність польоту БПЛА, у кілометрах (у вигляді слайдеру із числовим полем для введення).

— Розмах крил БПЛА, у метрах (у вигляді слайдеру із числовим полем для введення).

— Максимальна злітна маса БПЛА, у кілограмах (у вигляді слайдеру із числовим полем для введення).

— Корисне навантаження, що може нести БПЛА, у кілограмах кілограмах (у вигляді слайдеру із числовим полем для введення).

— Максимальна швидкість польоту БПЛА, в кіломентарх на годину кілограмах (у вигляді слайдеру із числовим полем для введення).

— Крейсерна швидкість польоту БПЛА, в кіломентарх на годину кілограмах (у вигляді слайдеру із числовим полем для введення).

— Максимальна висота польоту БПЛА, в кілометрах кілограмах (у вигляді слайдеру із числовим полем для введення).

— Операційна висота використання БПЛА, в кілометрах кілограмах (у вигляді слайдеру із числовим полем для введення).

— Тривалість польоту БПЛА, в годинах (у вигляді слайдеру із числовим полем для введення).

Після заповнення всіх вищеописаних полів – потрібно натиснути клавішу «Далі», для переходу на другий етап введення інформації. На цьому, другому, етапі здійснюється завантаження (одного чи декількох) фото БПЛА, із комп’ютера користувача сайту. Для цього є клавіша «ОБЕРІТЬ ФОТО». Також в цьому етапі відображається список вже завантажених фото, котрі можливо, за потреби, видалити, натиснувши на відповідну клавішу «Х». В кінці – потрібно натиснути на клавішу «Далі – для переходу на наступний етап.

На третьому, останньому, етапі здійснюється перевірка усіх введених даних. У вигляді стрічки всіх фото можливо переглянути обрані фотографії БПЛА, та у вигляді вертикального іменованого списку – інші текстові та числові дані. В кінці етапу

потрібно натиснути на клавішу «ДОДАТИ ДАНІ» для публікації нового БПЛА в БД веб-сайту. Як результат – БПЛА буде додано.

На кожному із цих трьох етапів є можливість повернутися на попередній етап, за допомогою клавіші «НАЗАД». Також, є клавіші навігації.

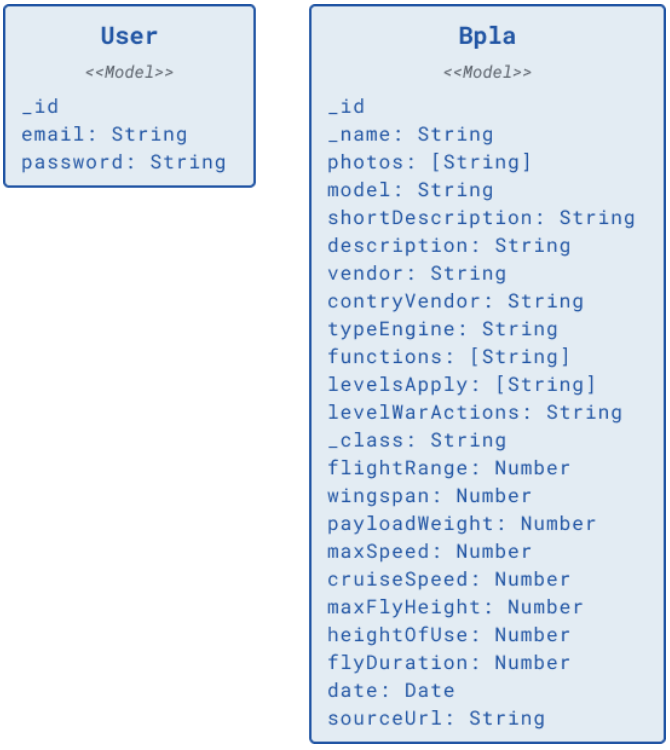
Після закінчення додавання інформації про нові БПЛА – потрібно вийти з даної сторінки, за допомогою клавіші «LOG OUT».

4.3.3. База даних проекту

Робота з базою даних відбувається за допомогою СУБД MongoDB. Як було описано раніше – це документо-орієнтована система керування базами даних (СКБД) з відкритим вихідним кодом, яка не потребує опису схеми таблиць [14-16].

В процесі проектування та створення веб-сайту з пошуку та перегляду БПЛА із заданими властивостями також було створено БД для цього сайту. Схема даної БД преставлена у вигляду діаграми моделей (колекцій) не реляційної БД (СУБД – MongoDB). Вона складається з наступних двох таблиць (див. рис. 4.24):

1. «User». Таблиця, в якій зберігаються дані про адміністраторів веб-сайту.
2. «Bpla». Таблиця, в якій зберігаються дані про БПЛА.



«Рис. 4.24. Схема бази даних проекту»

Це дві окремі (невзаємопов’язані таблиці), кожна з яких відповідає за свій функціонал взаємодії з веб-сайтом. Розглянемо кожну з цих таблиць окремо.

В таблиці «User» зберігаються облікові записи для авторизації адміністраторів веб-сайту. Ці дані може заносити лише розробник сайту. Авторизація здійснюється в адмін-сторінці сайту, за допомогою якої адміністратор може додати новий БПЛА до сайту (детально було описано в попередніх підрозділах). Дана таблиця складається з наступних полів:

— «_id». Унікальний ID адміністратора веб-сайту, дані про якого зберігаються в БД.

— «email». Логін адміністратора веб-сайту, у вигляді електронної пошти. Тип даних поля – String (рядок).

— «password». Пароль адміністратора веб-сайту. Тип даних поля – String (рядок).

В таблиці «Bpla» зберігаються інформація про БПЛА, що записані в БД веб-сайту. Дані БПЛА відображаються на головній сторінці сайту у вигляді карточки з назвою БПЛА, моделлю БПЛА, його коротким описом та зображенням. Дана таблиця складається з наступних полів:

— «_id». Унікальний ID БПЛА.

— «_name». Назва (ім’я) БПЛА. Тип даних поля – String (рядок).

— «model». Назва (тип) моделі БПЛА. Тип даних поля – String (рядок).

— «shortDescription». Короткий опис (інформація) про БПЛА. Тип даних поля – String (рядок).

— «description». Загальний (повний) опис (інформація) про БПЛА. Тип даних поля – String (рядок).

— «vendor». Виробник (назва компанії), що створює БПЛА. Тип даних поля – String (рядок).

— «contryVendor». Країна, що виробляє БПЛА. Тип даних поля – String (рядок).

— «typeEngine». Тип двигуна БПЛА. Тип даних поля – String (рядок).

— «functions». Функції БПЛА. Тип даних поля – String (рядок).

— «levelsApply». Рівень застосування БПЛА. Тип даних поля – String (рядок).

— «levelWarActions». Рівень воєнних дій БПЛА. Тип даних поля – String (рядок).

— «_class». Клас БПЛА. Тип даних поля – String (рядок).

— «flightRange». Дальність польоту БПЛА. Тип даних поля – Number (число подвійної точності з плаваючою комою).

— «wingspan». Розмах крил БПЛА. Тип даних поля – Number (число подвійної точності з плаваючою комою).

— «payloadWeight». Корисне навантаження БПЛА. Тип даних поля – Number (число подвійної точності з плаваючою комою).

— «maxSpeed». Максимальна швидкість польоту БПЛА. Тип даних поля – Number (число подвійної точності з плаваючою комою).

— «cruiseSpeed». Крейсерна швидкість польоту БПЛА. Тип даних поля – Number (число подвійної точності з плаваючою комою).

— «maxFlyHeight». Максимальна висота польоту БПЛА. Тип даних поля – Number (число подвійної точності з плаваючою комою).

— «heightOfUse». Операційна висота використання БПЛА. Тип даних поля – Number (число подвійної точності з плаваючою комою).

— «flyDuration». Тривалість польоту БПЛА. Тип даних поля – Number (число подвійної точності з плаваючою комою).

— «date». Дата створення та занесення в БД запису (інформації) про БПЛА. Тип даних поля – Data (дата з днем, місяцем та роком).

— «sourceUrl». Посилання на джерело БПЛА. Тип даних поля – String (рядок).

Важливою особливістю в даній БД є те, що паролі адміністраторів веб-сайту не зберігаються в БД та не передаються до неї у звичайному (відкритому) вигляді. Дана змінна шифрується та зберігається в таблиці «Use» (в полі «password») – в

шифрованому виді, у вигляді хешу. Хеш – це результат обробки даних хеш-функцією [18]. А хеш-функція – функція, що перетворює вхідні дані будь-якого (як правило великого) розміру у вихідний бітовий рядок фіксованого розміру [18]. Процес хешування в даному дипломному проекті досягається за рахунок використання бібліотеки Bcrypt.js.

Bcrypt.js – бібліотека, що написана мовою JavaScript, для кодування і декодування даних. Потрібна для кодування паролю адміністратора, щоб хакери не змогли на пряму бачити пароль в БД. Дана бібліотека чудовий вибір для хешування паролів, оскільки його "робочий коефіцієнт" можна регулювати, а це означає, що час, необхідний для генерації хешу, може бути збільшений із збільшенням потужності обладнання. При хешуванні паролів повільний процес — це добре. Чим довше алгоритм займається хешуванням пароля, тим більше часу знадобиться для злому облікового запису.

4.3.4. Домен та хостинг

Веб-сайт з пошуку та перегляду БПЛА із заданими властивостями повинен задовольняти потреби користувачів у пошуку потрібних їм БПЛА за заданими характеристиками. Для цього сайт повинен мати відповідний функціонал, що доступний кожному користувачу в будь-який період часу. Це можливо досягти за рахунок публікації сайту в глобальній мережі Інтернет. Тому, було прийнято рішення про використання домену та хостингу для даної дипломної роботи.

Використання послуг веб-хостингу та доменного ім’я – це основні (головні) два параметри, які потрібно налаштувати для налаштування роботи веб-сайту в глобальній мережі Інтернет. Спочатку потрібно розібратися з поняттями «доменне ім’я» та «веб-хостинг» (або по іншому – хостинг).

Доменне ім’я – це головна адреса сайту, за допомогою якої кожен користувач Інтернету може знайти ваш сайт та почати з ним взаємодіяти [18]. А хостинг – це послуга надання дискового простору, підключення до мережі та інших ресурсів для розміщення інформації на сервері [18]. Тобто, це процес резервування для певного

веб-сайту ресурсів на конкретному сервері та фізичного простору на диску цього серверу.

Проте, хостинг та доменне ім’я – це схожі елементи, що доповнюють один одного, проте це зовсім два різних поняття. Тому, потрібно зрозуміти точну різницю між ними.

Якщо пояснювати простими словами, то доменне ім’я – це URL-адреса веб-сайту, що вводиться в веб-браузері для доступу до певного веб-сайту. Тобто, доменне ім’я – це зручний спосіб доступу людей до веб-сайтів. Без його наявності – потрібно використовувати інший спосіб доступу до сайту – за допомогою цифрового еквіваленту (мітки), що присвоюється кожному веб-серверу, який підключений до глобальної мережі Інтернет та на якому фізично розташовуються веб-сайти. Ця мітка називається – IP-адресою [18].

Щоб отримати певне доменне ім’я для сайту – його попередньо потрібно зареєструвати. Процес реєстрації домену – це резервування імені (адресного рядку) в глобальній мережі Інтернет на певний час (період).

В даному дипломному проекті, для веб-сайту з пошуку та перегляду БПЛА із заданими властивостями, було вибрано наступне доменне ім’я (адресний рядок): bpla-client.vercel.app.

Далі розглянемо вибір веб-хостингу. Як було зазначено раніше, веб-хостинг дозволяє опублікувати веб-сайт в глобальній мережі Інтернет. Тобто, провайдери хостингу орендують частину свого веб-сервера для зберігання файлів та даних вашого веб-сайту.

Кожного разу, коли користувач вводить доменне ім’я веб-сайту – постачальник веб-хостингу повинен нести відповідальність за за надсилання його вмісту користувачеві сайту [18].

Переваги використання веб-хостингу для публікації веб-сайту наступні:

— Надання можливості зручного керування та управління веб-сайтом. Це здійснюється за допомогою спеціалізованих інструментів, що легкі у використанні.

— Підтримка надійної та безвідмовної роботи веб-сайту в режимі реального часу.

— Захист серверу та файлів веб-сайту від кібератак.

З всього вищеописаного можливо зробити наступний висновок: доменне ім’я використовується як адреса веб-сайту (вводиться в пошуковий рядок веб-браузера), а хостинг – для фізичного розташування сайту на дисковому просторі певного веб-серверу. Тобто, вони працюють взаємопов’язано та злагоджено та надають якісні послуги користувачам веб-сайту.

Для належної роботи веб-сайту з пошуку та перегляду БПЛА із заданими властивостями в мережі Інтернет потрібно передати створене створене доменне ім’я постачальнику послуг веб-хостингу. Ці дії виконуються після отримання чи придбання доменного ім’я.

Для даної дипломної роботи було обрано веб-хостинг Vercel. Vercel – платформа, яка дозволяє розгортати веб-сервіси у хмарі [19-20]. З її допомогою розробники розміщують статичні веб-сайти, які миттєво розгортаються, а QA команда може протестувати їх у цьому ж оточенні[19-20].

Vercel – це дуже зручний веб-хостинг, що не вимагає налаштування та працює з будь-яким типом веб-інфраструктури. Зручною його особливістю є те, що даний веб-хостинг надає можливість попереднього отримання URL-адреси, яку можливо передати своїм тіммейтам для спільної роботи над веб-сайтом. Тобто, надає можливість розподіленої розробки та модернізації сайту цілою командою, а не лише окремим розробником. Також перевагою в сторону вибору Vercel було те, що:

— У продакшен потрапляє більш стабільна і чиста версія продукту. Чистіший продукт — щасливіший за клієнта.

— План використання «Hobby». Можливість використання безкоштовного хостингу для персональних та некомерційних проектів.

4.4. Тестування програмного забезпечення

Процес тестування веб-сервісу здійснювався методикою QA тестування.

Тестування забезпечення якості (QA) – це процедура забезпечення того, що продукт пройшов всі стадії перевірки та був виданий клієнту з найкращою якістю, без

збоїв та багів [21]. А процес забезпечення якості – це спеціальні методи, які використовуються при тестуванні з метою запобігання проблемам із програмним продуктом або послугою та забезпечення чудового досвіду роботи для клієнтів [21].

Зазвичай, при створені веб-додатків, тестування впроваджувалось ще в циклі розробки програмного коду. Проте, цей підхід може виявитись не надійним. Наприклад, використання саме моделі. Як приклад – модель Waterfall. Дана модель розробки програмного забезпечення працює за наступним чином:

— Написаний код перетворюють в реальний програмний продукт.

— Програмний продукт та програмний код передаються команді QA, яка здійснює тестування продукту та коду.

— Результати, у вигляді відгуків на наявність проблем, команда QA відправляє на доопрацювання.

— І так далі – поки не буде досягнуто кінцевої мети.

Проте, на відміну від моделі Waterfall, є більш покращена модель – модель Agile. Це принцип за яким команди розробників та команди тестувальників та спеціалістів з кібербезпеки працюють разом, як один злагоджений механізм. Оскільки розробники, оператори та тестувальники несуть спільну відповідальність за те, щоб забезпечити якісний кінцевий продукт, QA часто намагається знайти своє місце [14].

Основна ціль контролю якості програмного забезпечення – отримання правильних результатів, що задовольняють поставлену мету, за допомогою набору спеціальних процедур та інструментів. Це означає, що організації та установи повинні бути впевненими, що їх дії спрямовані на досягнення бажаних результатів та показників високої якості.

Тестування програмного забезпечення – це єдиний та дієвий спосіб зберегти високу якість продукту на всьому етапі його розробки та впровадженні.

Висновки за розділом 4

Створено опис програмної реалізації веб-сервісу (веб-сайту) з пошуку та перегляду БПЛА із заданими властивостями. Даний опис охоплював аналіз вимог до програмного забезпечення, що включав у себе:

— Глосарій – короткий словник термінів та понять.

— Предмет розробки.

— Вимоги до графічного інтерфейсу.

— Вимоги до функціональності сайту.

— Вимоги до вмісту сайту.

На основі вимог було здійснено проектування програмного забезпечення. Це стало можливим за рахунок використання діаграми варіантів використання, діаграми розміщення та діаграм компонентів.

На основі описаних вимог та спроектованих діаграм – було здійснено розробку програмного забезпечення. Також, було створено та описано БД проекту.

В процесі перевірки якості програмного забезпечення проводилось його постійне тестування на етапі розробки та впровадження.

В кінцевому результаті – було здійснено вибір домену та хостингу, а також здійснена публікація веб-сайту в глобальній мережі Інтернет. Публікація була здійснена на веб-сервері Versel.

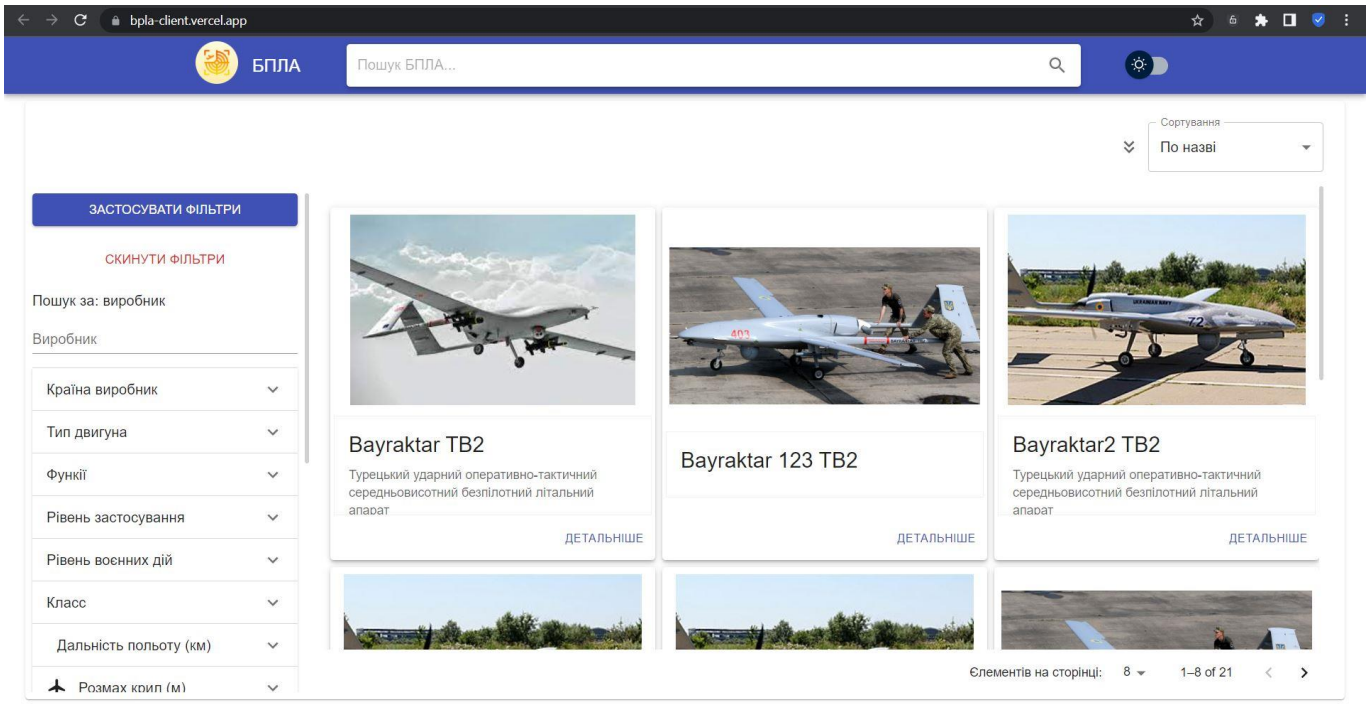
РОЗДІЛ 5
МЕТОДИКА РОБОТИ КОРИСТУВАЧА

Для належної роботи зі створеним веб-сервісом потрібно мати завантажений браузер для відображення сайту. В браузері буде відображатися весь функціонал робочого сайту.

Також потрібно мати постійний доступ до глобальної мережі Інтернет, щоб підключитися до серверу на якому розташований даний сервіс. В якості тестового серверу було використано сервіс Vercel.

5.1. Взаємодія користувача з головною сторінкою

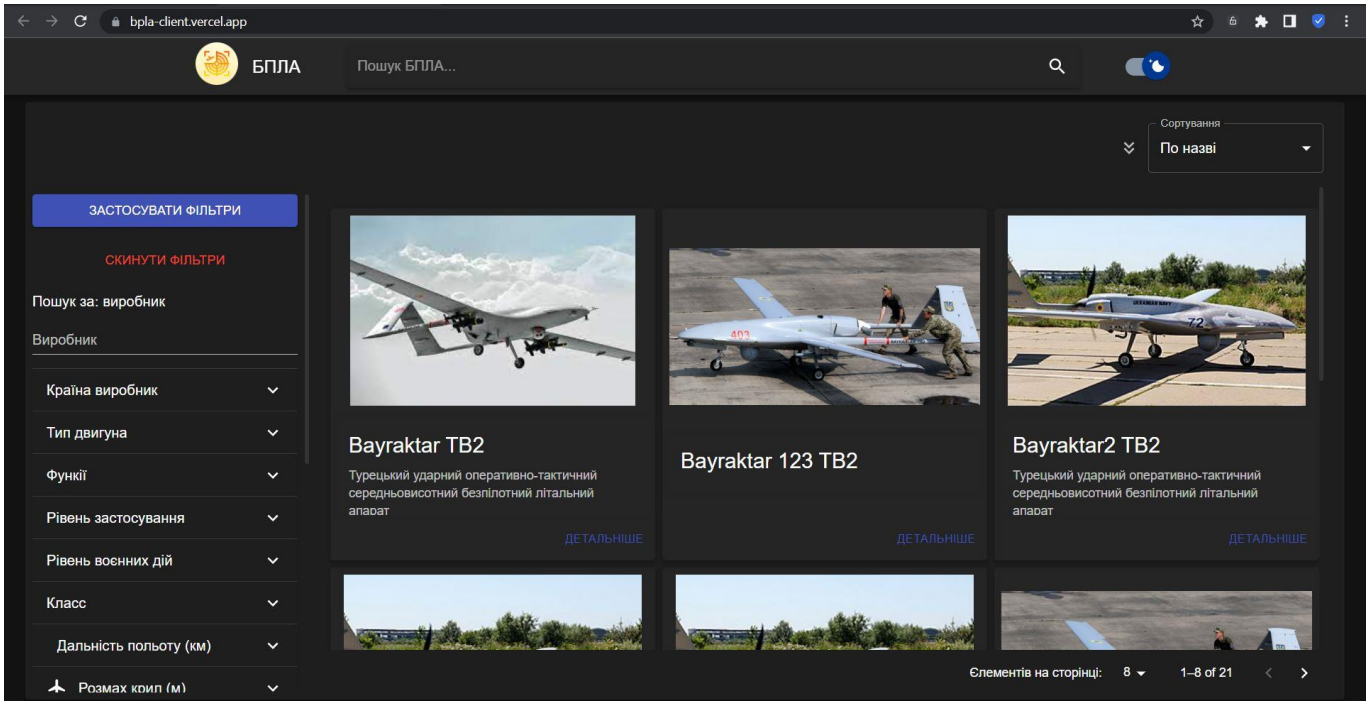
Завантаження сайту (перехід на сайт) відбувається при введені в адресному рядку браузера посилання https://bpla-client.vercel.app/. Дане посилання переносить користувача головну сторінку веб-сайту, де відбувається виведення виведення всіх наявних БПЛА, що записані в БД даного веб-сайту (див. рис. 5.1).



«Рис. 5.1. Головна сторінка сервісу пошуку та перегляду БПЛА із заданими властивостями»

Також на головній сторінці сайту присутня функція «Перемикач «теми» сайту», що дозволяє вибрати один із фонів сайту – в світлих чи темних тонах. Щоб продемонструвати її діяльність – потрібно натиснути на відповідний перемикач, що

знаходиться в верхній частині веб-сайту, праворуч від пошукового рядка (див рис. 5.2).

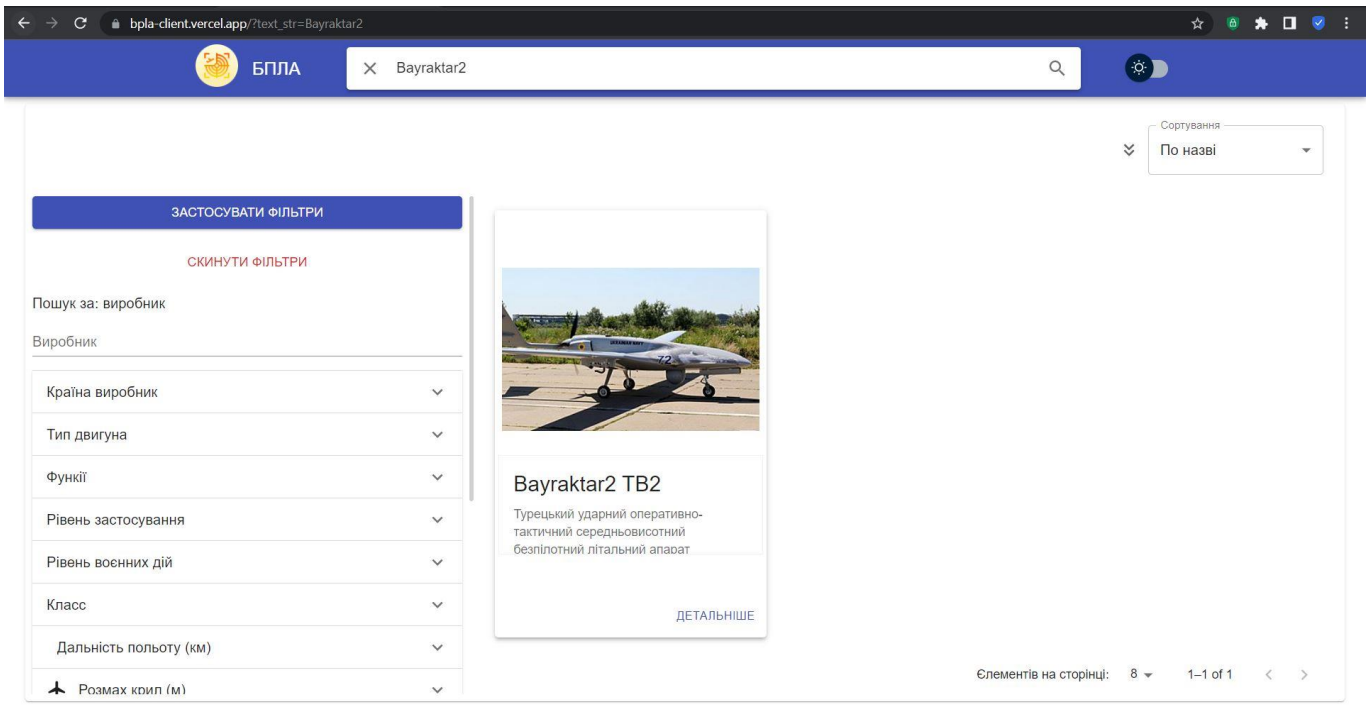


«Рис. 5.2. Головна сторінка сервісу пошуку та перегляду БПЛА із заданими властивостями в темному фоні»

5.2. Взаємодія користувача з функціями рекомендаційної системи

Як було вказано в розділі 4, за допомогою головної сторінки користувач веб-сайту може здійснити пошук потрібного йому БПЛА за певними параметрами. Пошук може здійснюватися двома способами – за допомогою функції «Пошуковий рядок» або ж за допомогою функції «Пошук за фільтрами».

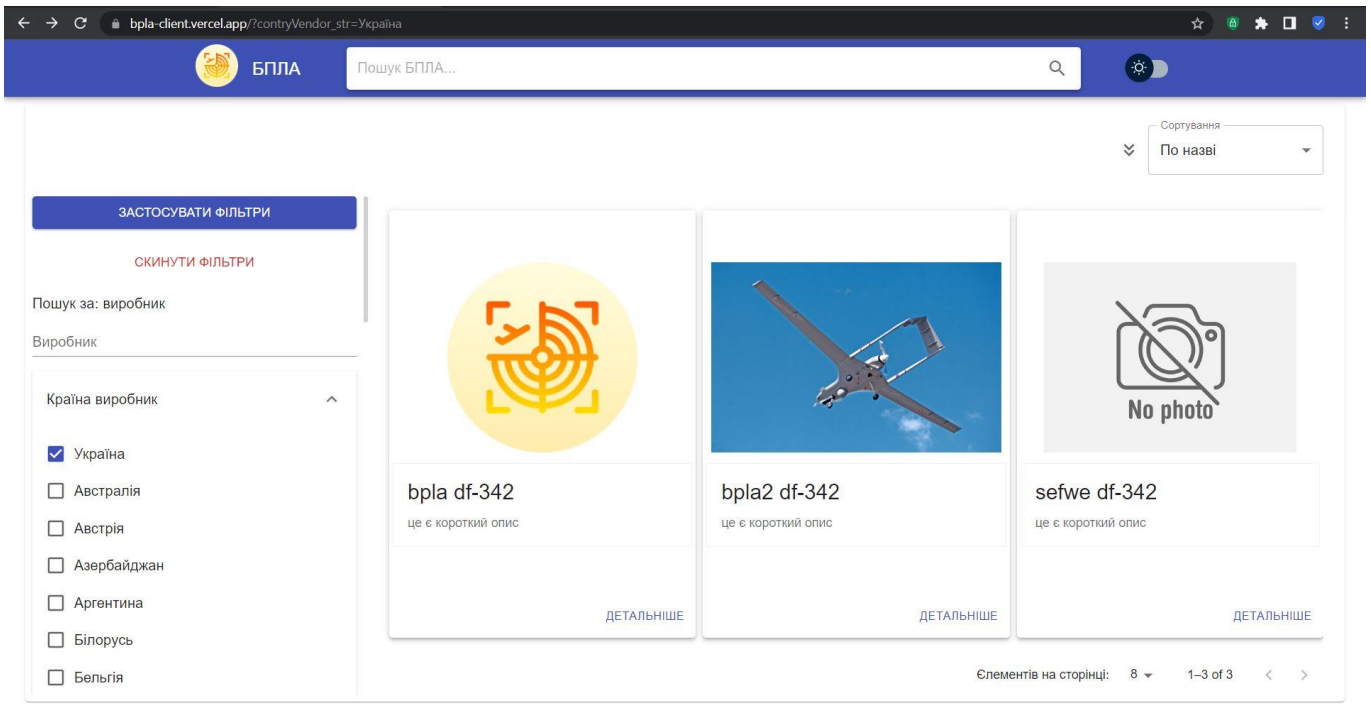
Для перевірки функціонування функції «Пошуковий рядок» потрібно ввести назву, модель, короткий чи довгий опис БПЛА який цікавить та натиснути на клавішу «Enter» чи клавішу «Лупа», що знаходиться на пошуковому рядку праворуч від поля вводу даних. В результаті – буде виведено лише один чи декілька БПЛА зі всіх наявних та які задовольняють умови пошуку. Як приклад, можливо вивести лише ті БПЛА, в назвах яких є «Bayraktar TB2». Отримаємо результат, що зображено на наступному рисунку (див. рис. 5.3).



«Рис. 5.3. Відображення роботи функції «Пошуковий рядок» сервісу пошуку та перегляду БПЛА із заданими властивостями»

Для завершення роботи з функцією «Пошуковий рядок» – потрібно натиснути на клавішу «х», що знаходиться ліворуч від пошукового рядку.

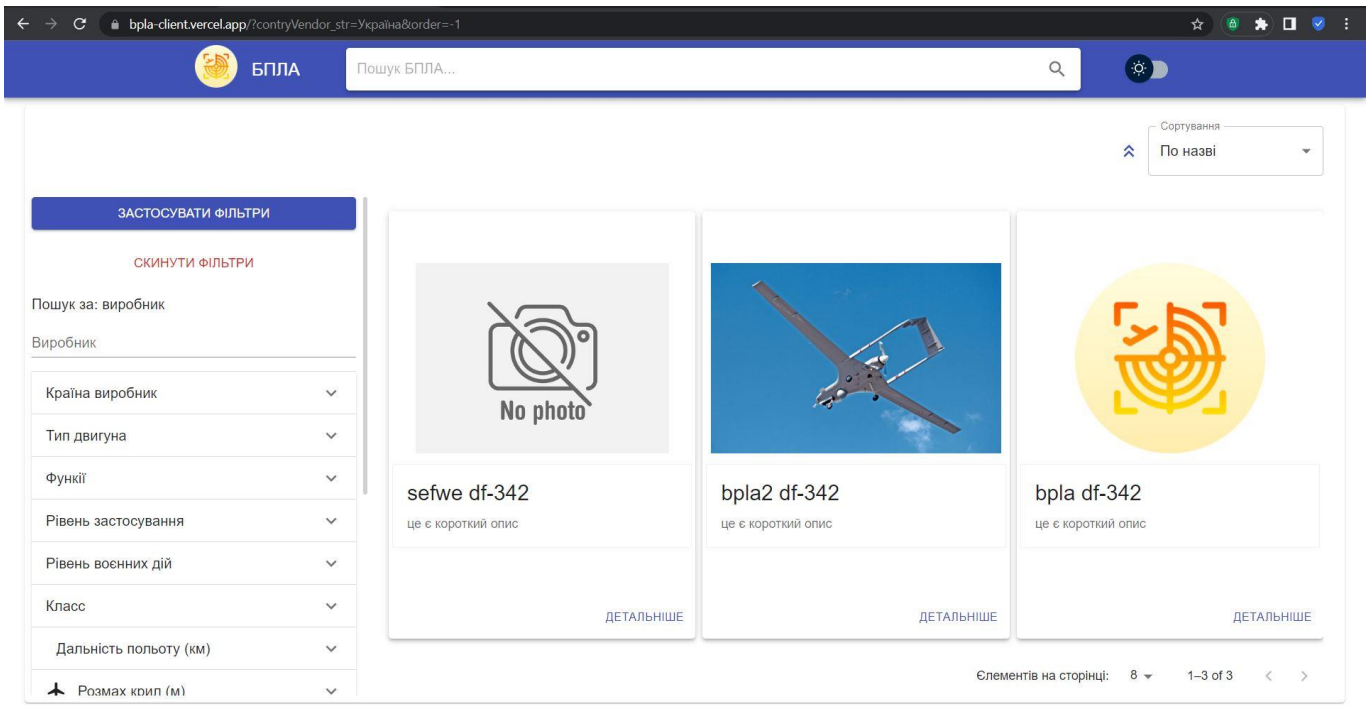
Далі, розглянемо принцип роботи функції «Пошук за фільтрами». Для пошуку БПЛА за фільтрами – потрібно вибрати параметри, за якими проводити пошук та натиснути клавішу «ЗАСТОСУВАТИ ФІЛЬТРИ». В результаті будуть виведені, на головну сторінку веб-сайту, лише БПЛА, які відповідають параметрам фільтрів. Всі описані клавіші та параметри знаходяться в лівому сайдбарі головної сторінки веб-сайту. Як приклад, можливо вивести лише ті БПЛА, країна виробник яких «Україна». Отримаємо результат, що зображено на наступному рисунку (див. рис. 5.4).



«Рис. 5.4. Відображення роботи функції «Країна виробник» сервісу пошуку та перегляду БПЛА із заданими властивостями»

Для завершення роботи з функцією «Пошук за фільтрами» – потрібно натиснути на клавішу «СКИНУТИ ФІЛЬТРИ», що знаходиться ліворуч від пошукового рядку.

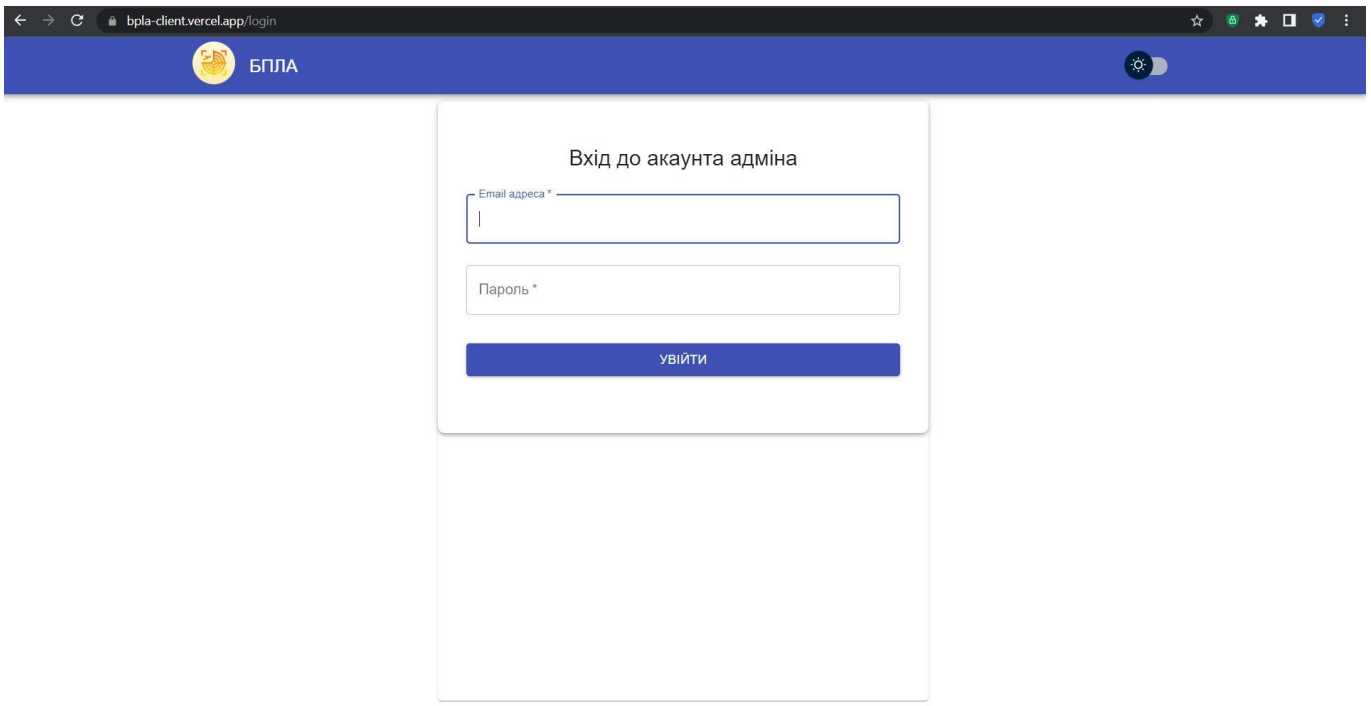
Ще однією особливість, що присутня на сайті – наявність функції «Сортування БПЛА». За допомогою даної функції можливо сортувати БПЛА за певним типом (параметром) сортування. Для цього потрібно вибрати потрібний користувачу тип сортування та натиснути на спеціальний перемикач, що відповідає за напрям сортування (в порядку збільшення або зменшення). Як приклад, буде здійснено сортування по назві БПЛА в порядку зменшення, тобто від останньої до першої букви алфавіту (див. рис. 5.5).



«Рис. 5.5. Відображення роботи функції «Сортування» сервісу пошуку та перегляду БПЛА із заданими властивостями»

5.3. Взаємодія адміністратора з оновленням рекомендаційної системи

Також за допомогою даного сервісу можливо додавати нові БПЛА та виводити їх на головну сторінку веб-сайту. Для цього потрібно записати карточку з інформацією про новий БПЛА до БД веб-сайту. Цю функцію може виконувати лише адміністратор веб-сайту. Для цього потрібно попередньо перейти на сторінку авторизації. Це здійснюється за рахунок уведення в адресний рядок браузера наступного рядку: https://bpla-client.vercel.app/login. Після цього в браузері відкриється вікно авторизації (вікно входу в адмін сторінку веб-сайту). Даний процес зображено на наступному рисунку (див. рис. 5.6).

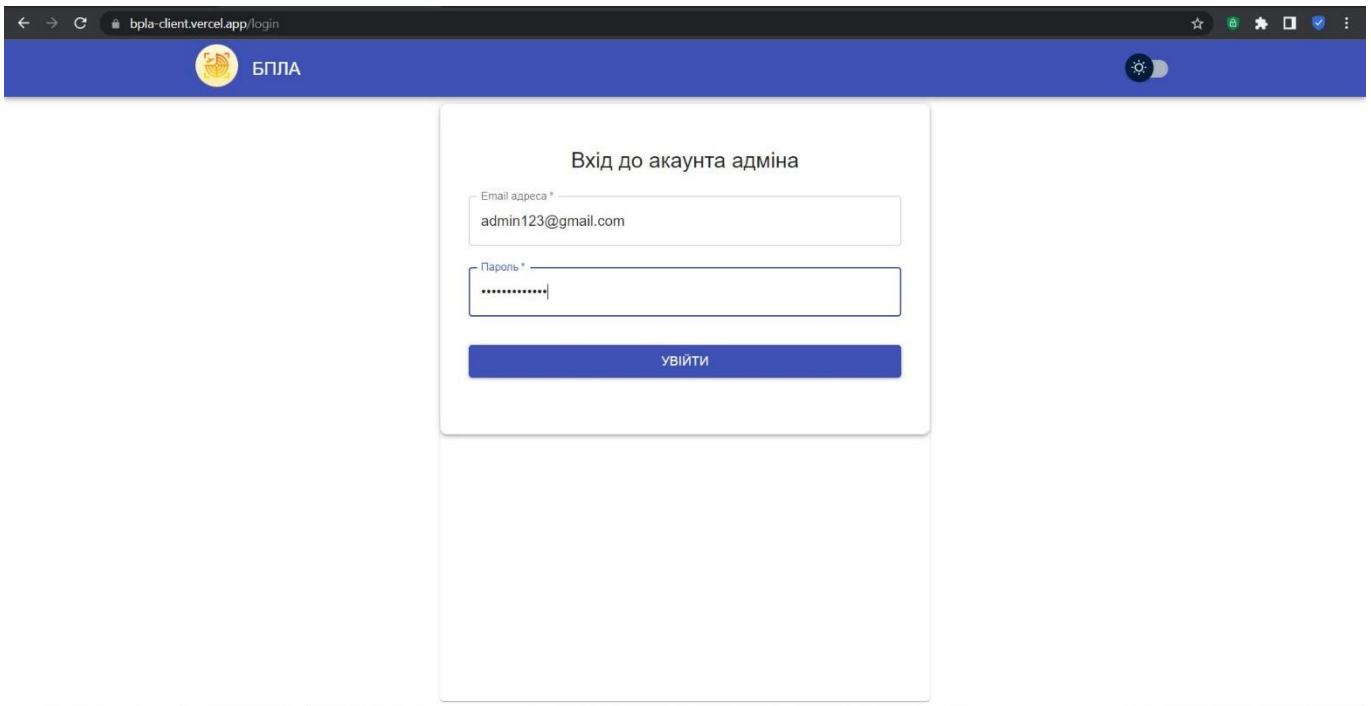


«Рис. 5.6. Відображення сторінки «Вхід до акаунта адміна»»

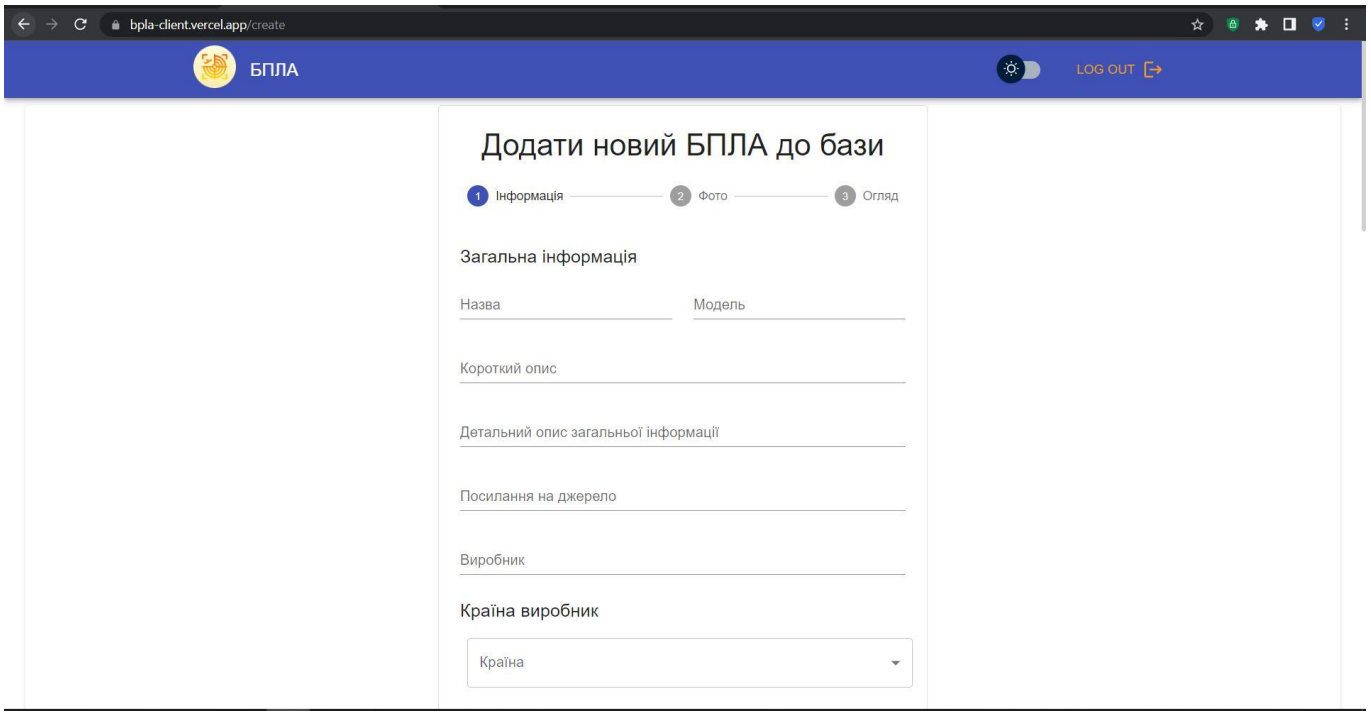
Для переходу на сторінку з додаванням інформації про нові БПЛА вводимо наступні дані (див. рис. 5.7):

1. В полі логін вводимо електрону пошту адміністратора веб-сайту.
2. В полі пароль вводимо секретний пароль адміністратора веб-сайту.
3. Натискаємо на клавішу «Увійти».
4. При правильному виконанні дії, що описані в пунктах 1-3, буде здійснено успішний вхід на сторінку з додаванням інформації про нові БПЛА, а саме на першу частину сторінки – «Інформація» (див. рис. 5.8).

Після успішної авторизації адміністратора на даній сторінці, крім успішного входу на нову сторінку, буде змінено url-сторінки в адресному рядку веб-браузера. Тобто нове посилання буде виглядати наступним чином: https://bpla-client.vercel.app/create



«Рис. 5.7. Відображення введення логіну та паролю адміністратора на сторінці «Вхід до акаунта адміна»»



«Рис. 5.8. Відображення успішного переходу на частину «Інформація» сторінки «Додати новий БПЛА до бази»»

Далі, після переходу на сторінку «Додати новий БПЛА до бази» в адміністратора веб-сайту з’являється можливість «створити» новий БПЛА (новий

запис про БПЛА) та відобразити його на головній сторінці даного сайту. Для цього потрібно заповнити відповідні поля потрібними параметрами.

На першому етапі, етапі «Інформація» – адміністратор додає основну інформацію про БПЛА. Як приклад, буде створено та добавлено новий запис про БПЛА вірменського походження «Крунк 25-1». Для цього вводимо наступну інформацію (див. рис. 5.9 - 5.11):

— Назва – «Крунк 25-1».

— Модель – «звичайна».

— Короткий опис – «тактичний безпілотний літальний апарат (міні БПЛА)».

— Детальний опис загальної інформації – «Виконано на двухбалковій системі з високо розташований крилом та гвинтом, що тягнеться».

— Посилання на джерело – «https://uk.wikipedia.org/wiki/%D0%9A%D1%80%D1%83%D0%BD%D0%BA_(%D0%91%D0%9F%D0%9B%D0%90)».

— Виробник – «ВПК Вірменії».

— Країна виробник – «Вірменія».

— Тип двигуна – «Дизельний».

— Функції – «Розвідка», «Цілевказання», «Підсвічування в темний час» та «Пошук і рятування».

— Рівень застосування – «ТВД», «Оперативна група», «Бригада».

— Рівень воєнних дій – «Тактичний підрозділ».

— Клас – «БПЛА середньої дальності».

— Дальність польоту (км) – «2000».

— Розмах крил (м) – «4».

— Максимальна злітна маса (кг) – «100».

— Корисне навантаження (кг) – «20».

— Максимальна швидкість (км/год) – «140».

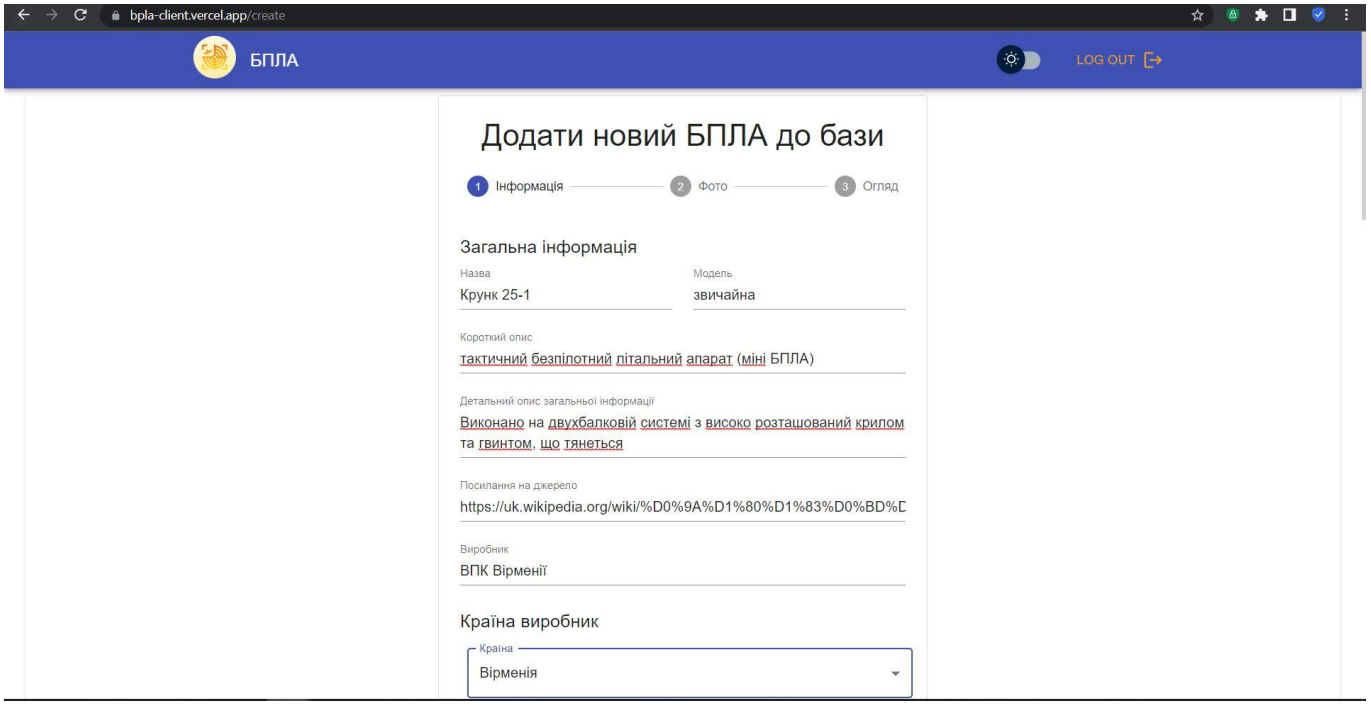
— Крейсерна швидкість (км/год) – «110».

— Максимальна висота польоту (км) – «4».

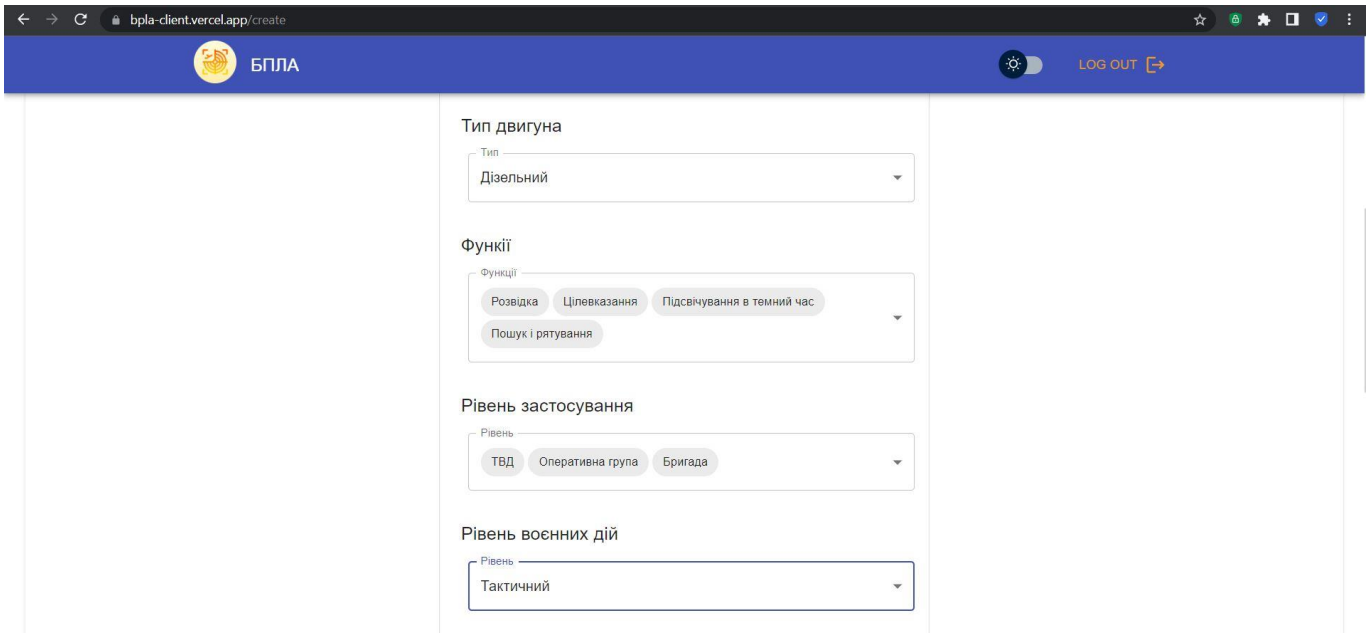
— Операційна висота використання (км) – «3».

— Тривалість польоту (годин) – «5».

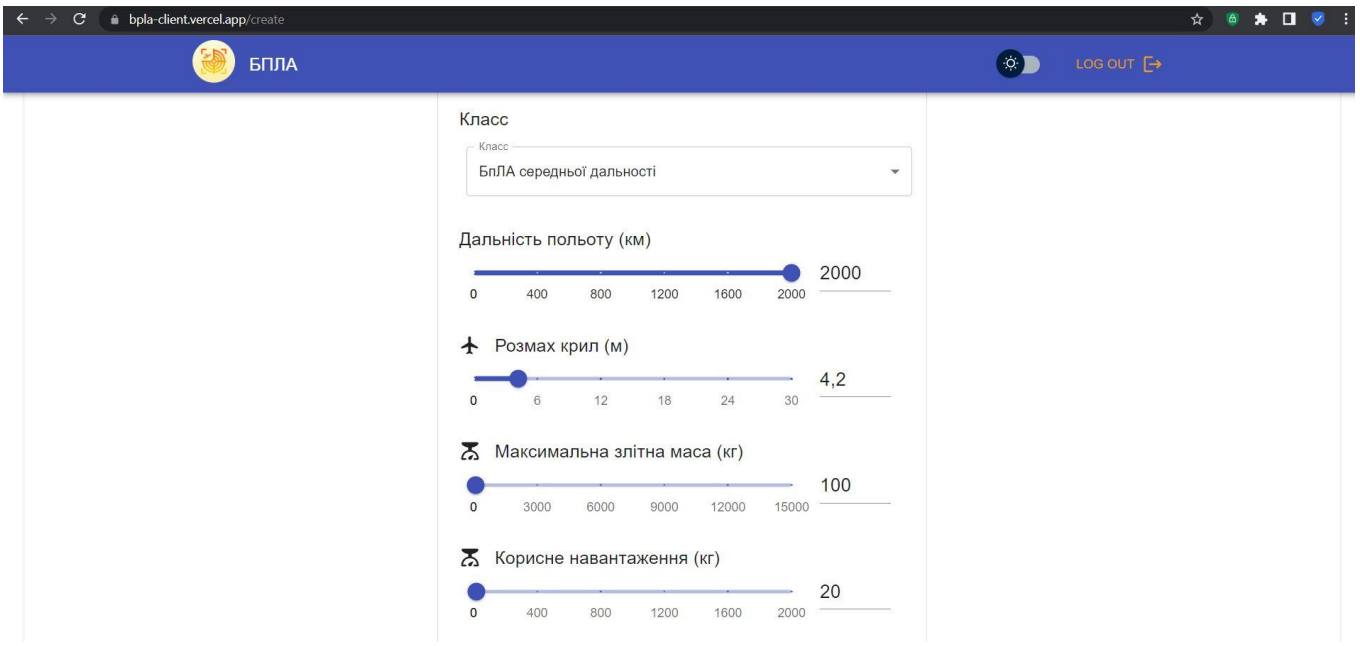
Після введення всієї інформації – адміністратор натискає клавішу «ДАЛІ», для переходу на наступну сторінку (сторінка «Фото»).



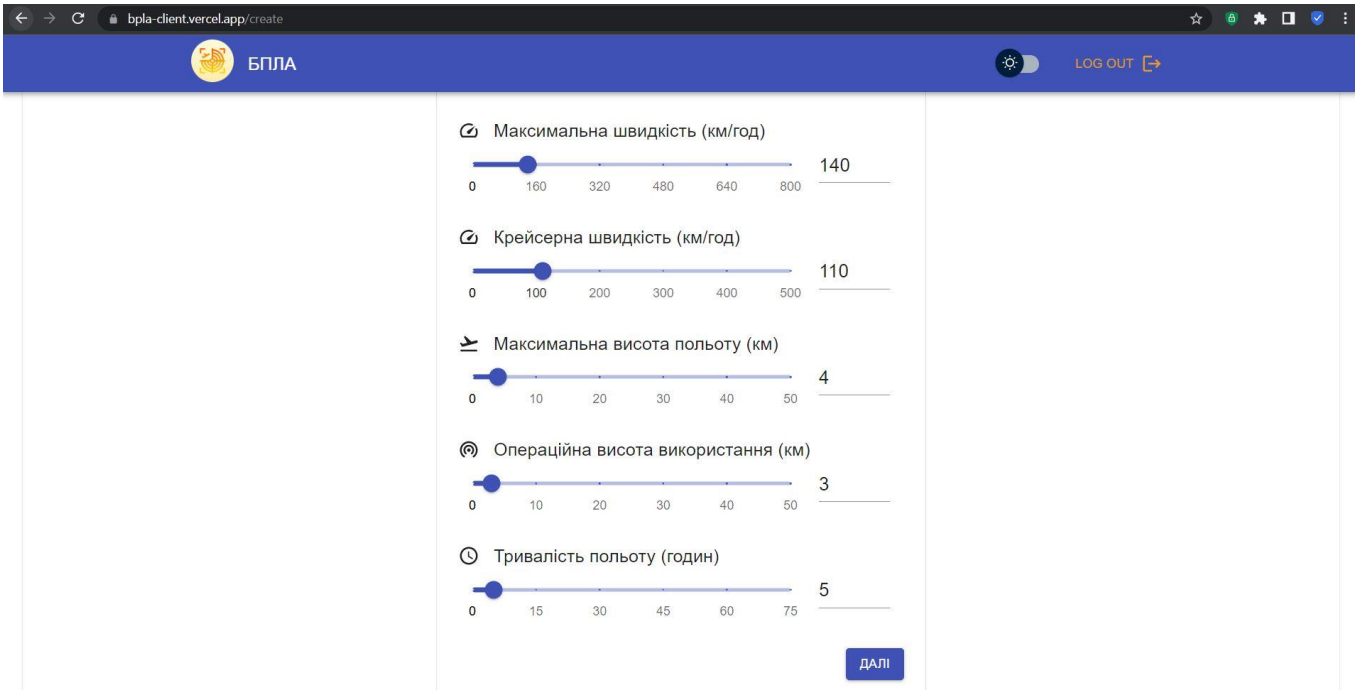
«Рис. 5.9. Відображення введення інформації (параметрів) про новий БПЛА»



«Рис. 5.10. Відображення введення інформації (параметрів) про новий БПЛА»

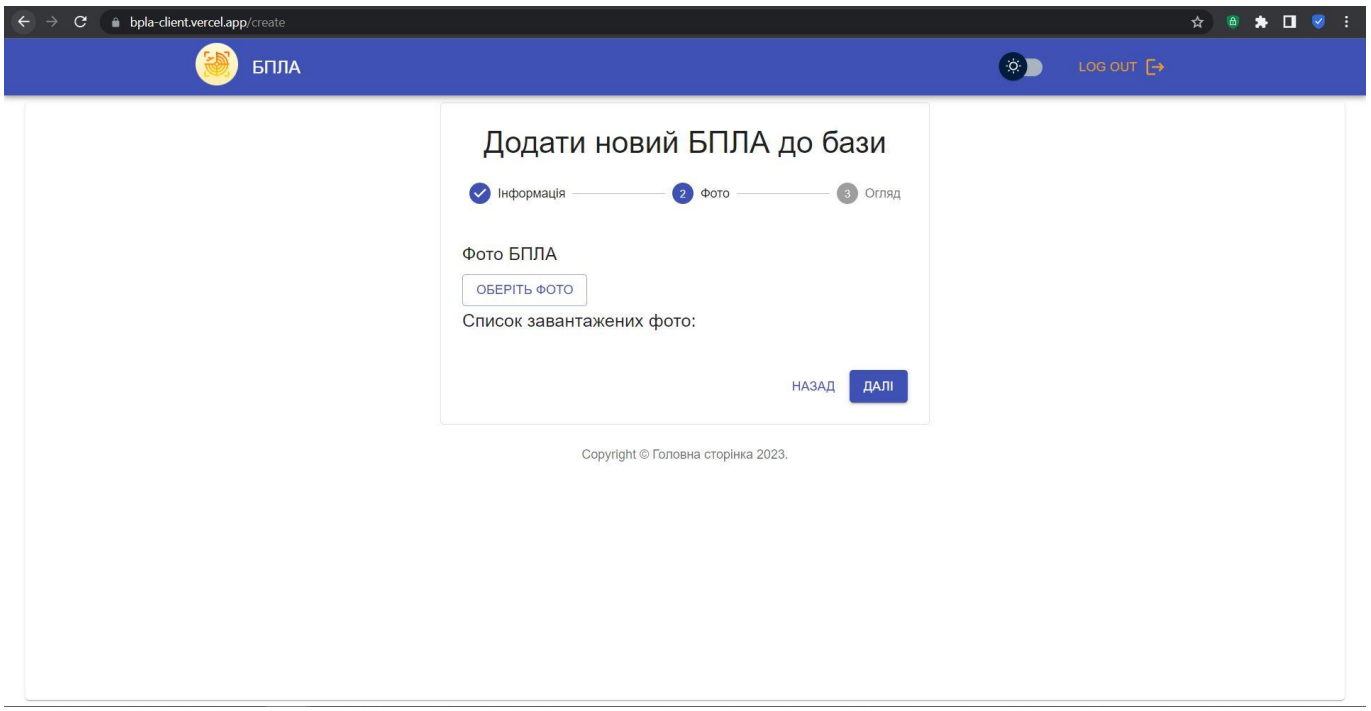


«Рис. 5.11. Відображення введення інформації (параметрів) про новий БПЛА»



«Рис. 5.12. Відображення введення інформації (параметрів) про новий БПЛА»

На наступній частині сторінки «Додати новий БПЛА до бази», частині «Фото» відбувається завантаження одного чи декількох фото БПЛА (див. рис. 5.13). Завантаження відбувається з твердого диску ПЕОМ адміністратора. Фото БПЛА, що відображатиметься на головній сторінці веб-сайту, буде перше із завантажених.



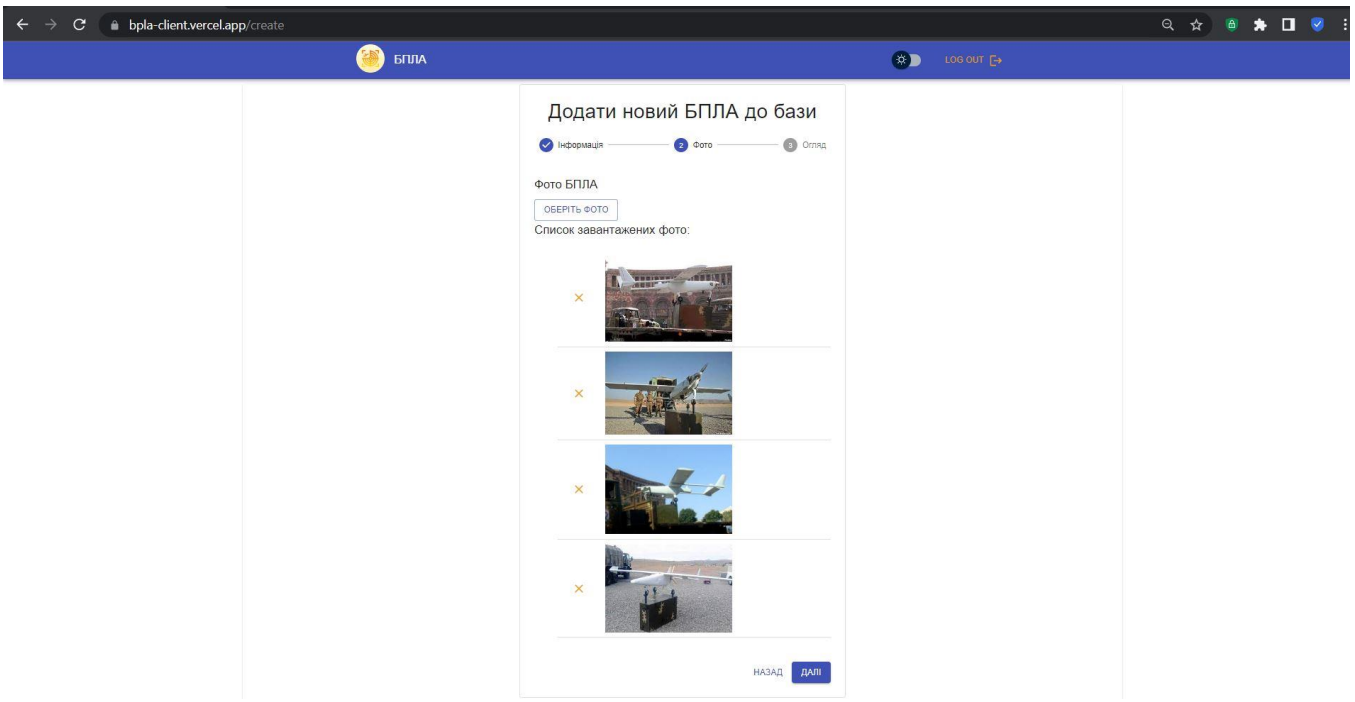
«Рис. 5.13. Відображення успішного переходу на частину «Фото» сторінки «Додати новий БПЛА до бази» веб-сайту»

На цій частині сторінки адміністратор веб-сайту вибирає які саме зображення потрібно завантажити для певного БПЛА. Для цього потрібно:

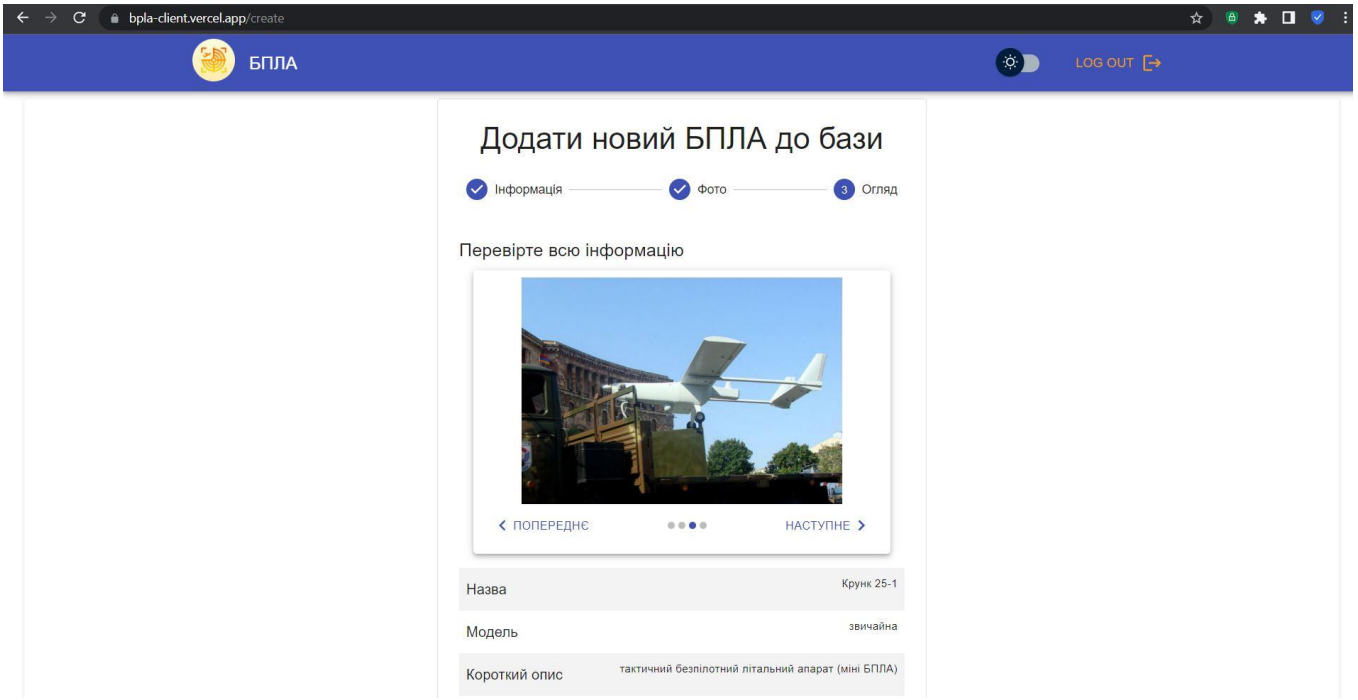
1. Натиснути на клавішу «ОБЕРІТЬ ФОТО».
2. Вибрати потрібну картинку (зображення) з твердого диску ПЕОМ адміністратора та натиснути клавішу «ОК» чи «Підтвердити».
3. Повторити кроки пунктів 1-2 до тих пір, поки не буде обрано потрібна кількість зображень.
4. Натиснути клавішу «ДАЛІ» для підтвердження вибору та переходу наступну частину сторінки.

В даному прикладі – було здійснено вибір чотирьох картинок для БПЛА «Крунк 25-1» та здійснено перехід на наступну частину сторінки «Додати новий БПЛА до бази» (див. рис. 5.14 - 5.15).

Якщо потрібно повернутися на попередню частину сторінки – тиснемо клавішу «НАЗАД».



«Рис. 5.14. Відображення завантаження картинок (зображень) БПЛА»



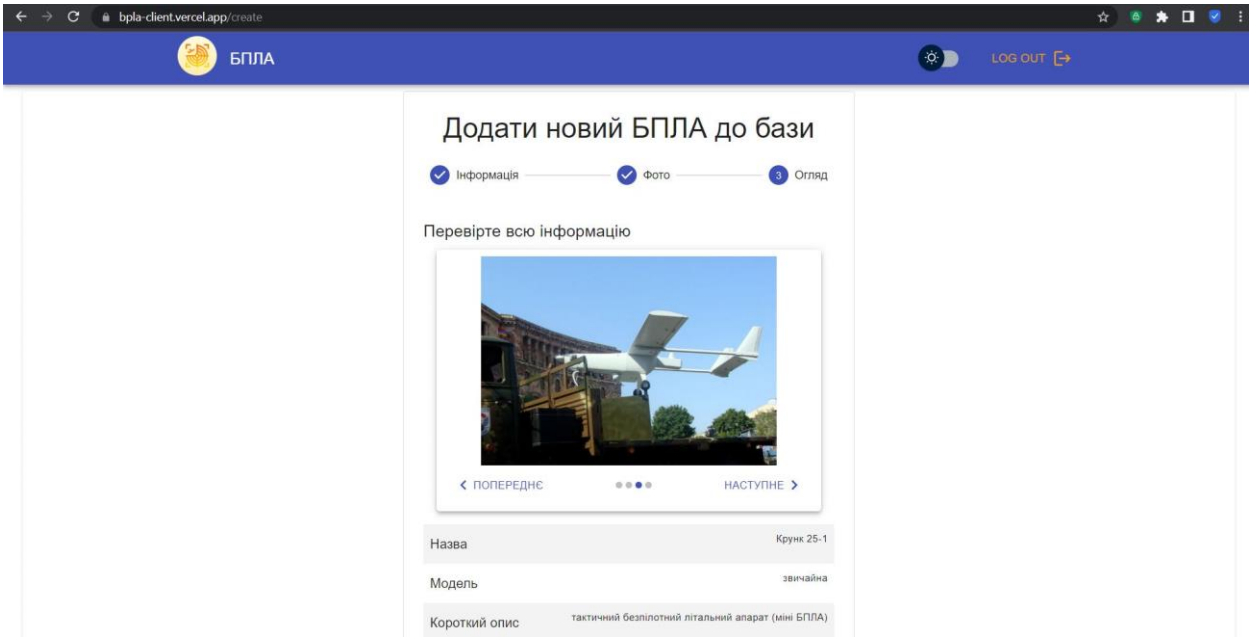
«Рис. 5.15. Відображення успішного переходу на частину «Огляд» сторінки «Додати новий БПЛА до бази» веб-сайту»

На третій частині сторінки «Додати новий БПЛА до бази», частині «Огляд», відбувається перегляд введених параметрів та завантажених зображень для вибраного БПЛА. Якщо адміністратора влаштовує всі дані які були уведені ним – то потрібно

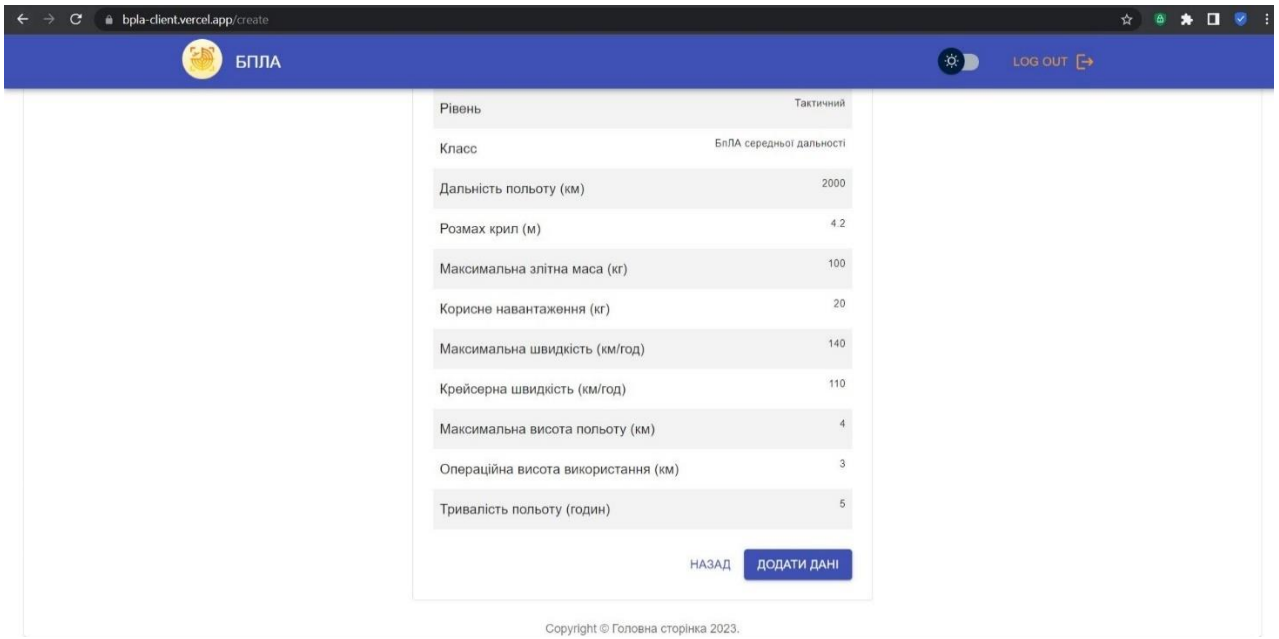
натиснути клавішу «ДОДАТИ ДАНІ». Після цього новий БПЛА буде успішно записано в БД веб-сайту.

Якщо потрібно повернутися на попередню частину сторінки – тиснемо клавішу «НАЗАД».

В даному випадку можливо переглянути введену інформацію та завантажені зображення про БПЛА «Крунк 25-1» (див. рис. 5.15 - 5.17).

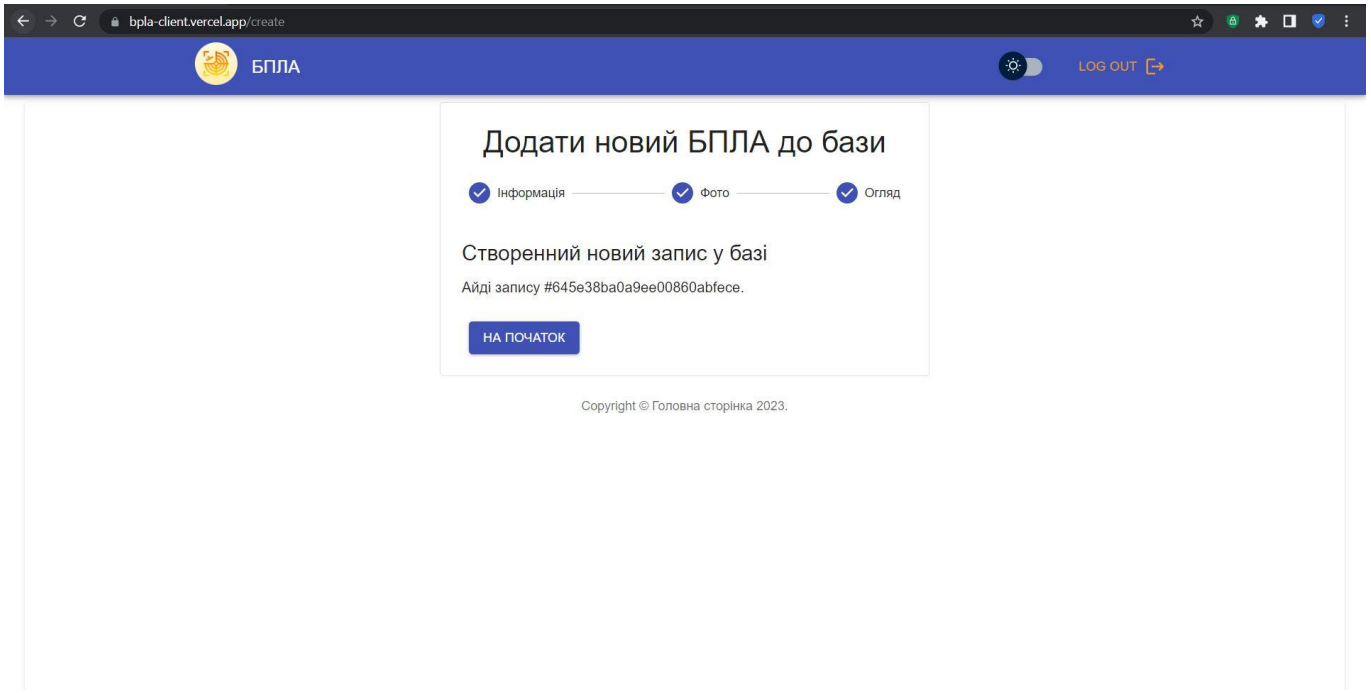


«Рис. 5.16. Відображення введеної інформації та завантажених зображень про новий БПЛА на частині «Огляд» сторінки «Додати новий БПЛА до бази» веб-сайту»



«Рис. 5.17. Відображення введеної інформації про новий БПЛА на частині «Огляд» сторінки «Додати новий БПЛА до бази» веб-сайту»

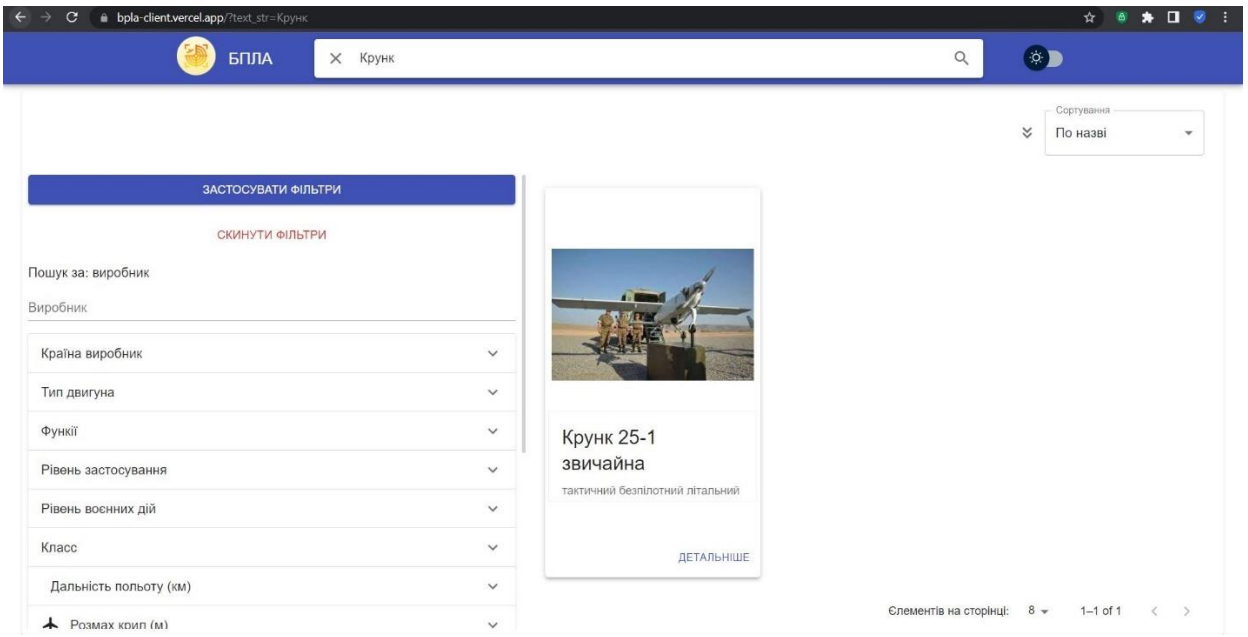
Після натискання клавіші «ДОДАДТИ ДАНІ» – адміністратору буде вивелнео повідомлення про успішне занесення нового БПЛА до БД веб-сайту (див. рис. 5.18). В даному випадку для БПЛА «Крунк 25-1» буде створено наступний ID в БД веб-сайту: #645e38ba0a9ee00860abfece.



«Рис. 5.18. Відображення запису про успішний запис інформації про новий БПЛА до БД веб-сайту»

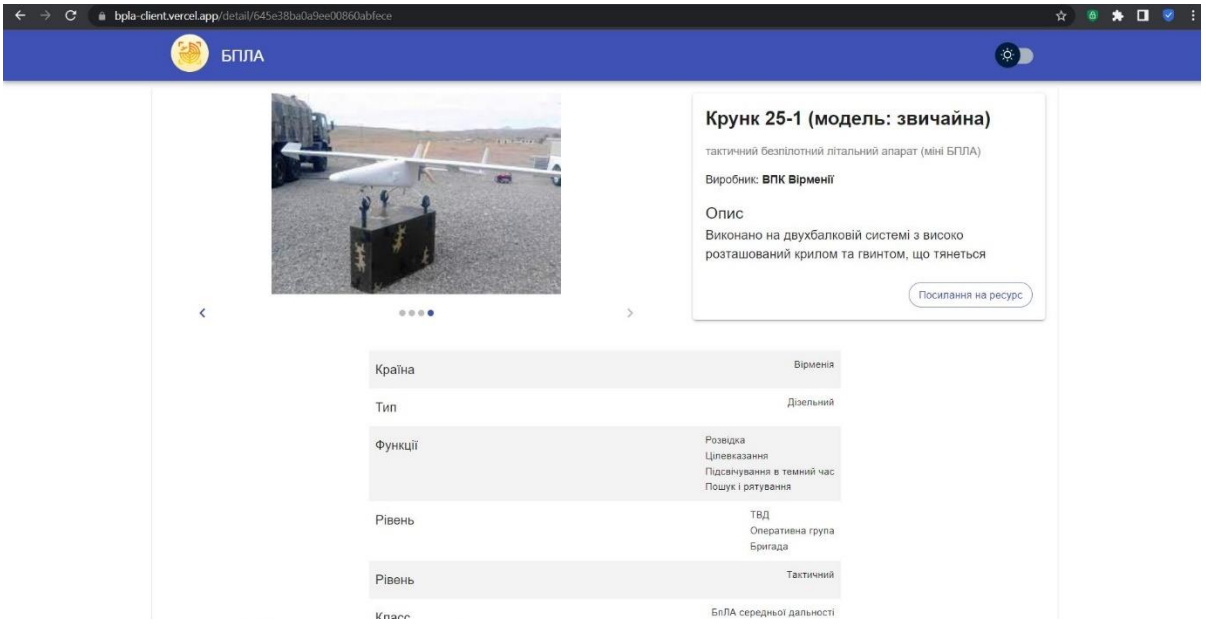
Для продовження додавання новий БПЛА до БД веб-сайту – потрібно натиснути клавішу «НА ПОЧАТОК» та повторити раніше описані кроки. Для завершення роботи зі сторінкою «Додати новий БПЛА до бази» потрібно натиснути на клавішу «LOG OUT», що знаходиться у верхній частині сторінки, праворуч від клавіші «Перемикач».

Для перевірки наявності створеного БПЛА в БД веб-сайту – вводимо на головній сторінці сайту, в пошуковому рядку, «Крунк» та натискаємо клавішу «Enter» чи клавішу «Лупи». В результаті буде відображено одна карточка зі створеним БПЛА (див. рис. 5.19).

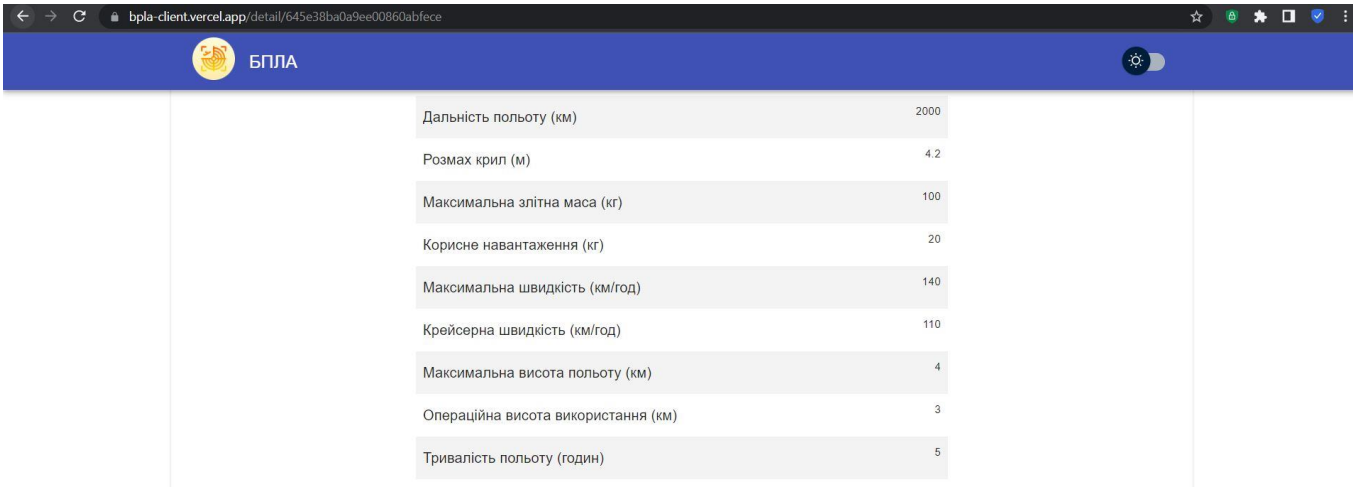


«Рис. 5.19. Відображення успішно створеного та записаного до БД веб-сайту БПЛА «Крунк 25-1»»

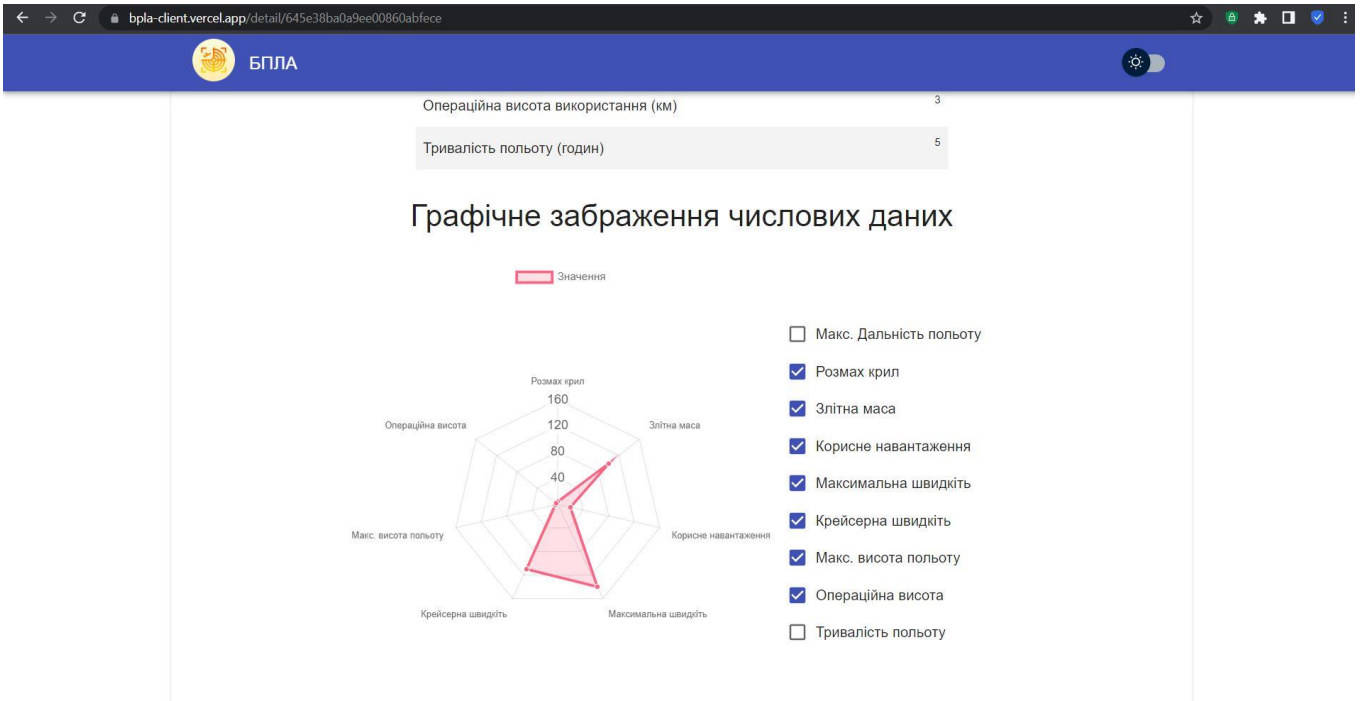
Для перегляду інформації про «Крунк 25-1» – потрібно натиснути на картинку даного БПЛА. В результаті, користувача веб-сайту буде переадресовано на сторінку з даним БПЛА. На ній буде зображена вся інформація про «Крунк 25-1», а також його зображення та посилання на офіційне джерело про даний БПЛА в глобальній мережі Інтернет(див. рис. 5.20 - 5.22). А також буде відображено графік «Графічне зображення числових даних» – більш красиве відображення параметрів БПЛА.



«Рис. 5.20. Відображення інформації про БПЛА «Крунк 25-1» на його сторінці даного веб-сайту»

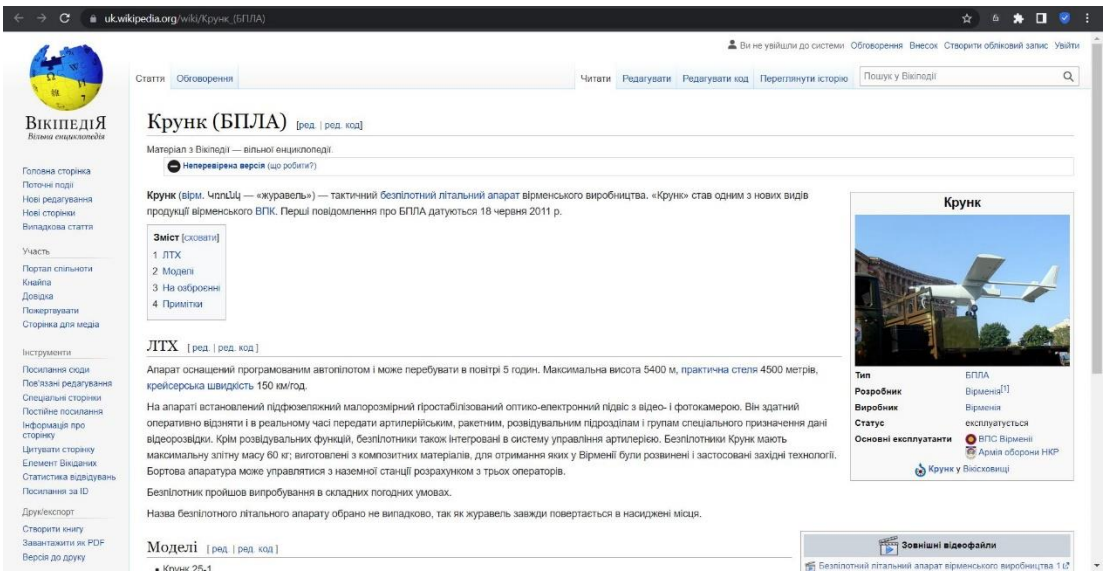


«Рис. 5.21. Відображення інформації про БПЛА «Крунк 25-1» на його сторінці даного веб-сайту»



«Рис. 5.22. Відображення інформації про БПЛА «Крунк 25-1» на його сторінці даного веб-сайту»

Для перегляду інформації про «Крунк 25-1» на офіційному джерелі – потрібно натиснути клавішу «Посилання на ресурс». В результаті, в браузер користувача веб-сайтом буде завантажено окрему сторінку з посиланням. В даному випадку: https://uk.wikipedia.org/wiki/Крунк_(БПЛА) (див. рис. 5.23).



«Рис. 5.23. Відображення посилання на офіційне джерело про БПЛА «Крунк 25-1» в глобальній мережі Інтернет»

ВИСНОВКИ

Вирішено актуальне завдання моделювання рекомендаційної системи вибору безпілотного літального апарату із заданими властивостями. Це стало можливим завдяки отриманню таких окремих результатів:

1. Здійснена постановка задачі на створення рекомендаційної системи вибору безпілотного літального апарату із заданими властивостями, саме – веб-сервісу (веб-сайту) з пошуку та перегляду БПЛА із заданими властивостями.

2. Проаналізовано веб-сайти (веб-сервіси) систем, що графічний інтерфейс та функціонал яких використовувались в якості прототипів (шаблонів) при проектуванні, створені та розробці рекомендаційної системи вибору безпілотного літального апарату із заданими властивостями. Серед них виокремлено найбільш розповсюджені програмні продукти – це інтернет-магазин Rozetka та сервіс E-Katalog. Було виділено їх переваги, що будуть використовуватися в веб-сервісі з пошуку та перегляду БПЛА із заданими властивостями.

3. Проаналізовано основні засоби розробки системи (веб-сайту) з пошуку та перегляду БПЛА із заданими властивостями. Зокрема розглянуто мови розробки, такі як – HTML, CSS, JavaScript. Також, для розробки системи використовується бібліотека React, що надійно служить при розробці графічних інтерфейсів. Також було використано спеціальні бібліотеки для покращення та прискорення розробки клієнтської та серверної частин програмного продукту. Дані авторизації адміністраторів веб-сайту, а також інформація про БПЛА містяться в БД Mongo, доступ до якої здійснюється за допомогою СУБД MongoDB. Дана БД разом із серверною частиною сайту опублікована в глобальній мережі Інтернет – для зручного доступу в будь-який момент часу та в будь-якому куточку Земної кулі.

4. Створено опис вимог до програмної реалізації веб-сайту, а саме – аналіз основних вимог, оцінено предмет розробки, створено вимоги до графічного дизайну, функціональності та вмісту веб-сайту. Було здійснено проектування даного програмного забезпечення, що представлено у вигляді діаграми варіантів використання, діаграми розміщення та діаграм компонентів. Було розглянуто саму

розробку програмного забезпечення. Було здійснено вибір БД та СУБД для даної системи. А також здійснено вибір домену та хостингу для веб-сайту.

5. Розроблено веб-сервіс для пошуку та перегляду БПЛА із заданими властивостями, а також проведення його тестування на всьому етапі розробки та створення. Як результат – створено інструкцію (методику) роботи користувача з даним веб-сервісом.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. «Як я став… Чечоткіним» — Epravda, 1 липня 2016. Електронний ресурс: http://www.epravda.com.ua/cdn/cd1/2016/07/chechetkin/
2. Alexa: rozetka.com.ua станом на 19.04.2020. Електронний ресурс: https://web.archive.org/web/20200419104428/https://www.alexa.com/siteinfo/rozetka.com.ua
3. Інтернет-термінал НСМЕП // Банківська енциклопедія / С. Г. Арбузов, Ю. В. Колобов, В. І. Міщенко, С. В. Науменкова. — Київ : Центр наукових досліджень Національного банку України : Знання, 2011. — С. 199. — (Інституційні засади розвитку банківської системи України).
4. Gerardus Blokdyk Database Replication A Complete Guide - 2020 Edition, 2020. — 31с.
5. Дакетт Джон. HTML и CSS. Разработка и создание веб-сайтов, 2017. — 23-25с.
6. Мельник Р.А. Програмування веб-застосувань (фронт-енд та бек-енд). — Видавництво Львівської політехніки. 2018. — 34-37с.
7. Джоэл Х. Спольски, Джоэл о программировании, 2006. — 79с.
8. Сергей Рогачев. Обобщённый Model-View-Controller // rsdn.org. — 2007.
9. Джесс Чедвик и др. ASP.NET MVC 4: разработка реальных веб-приложений с помощью ASP.NET MVC = Programming ASP.NET MVC 4: Developing Real-World Web Applications with ASP.NET MVC. — М.: «Вильямс», 2013. — 432 с. — ISBN 978-5-8459-1841-3.
10. Адам Фримен. ASP.NET MVC 4 с примерами на C# 5.0 для профессионалов, 4-е издание = Pro ASP.NET MVC 4, 4th edition. — М.: «Вильямс», 2013. — 688 с. — ISBN 978-5-8459-1867-3.

11. Ирина Бородкина, Георгий Бородкин, Інженерія програмного забезпечення. Посібник для студентів вищих навчальних закладів. — Центр навчальної літератури. 2018. — 38-41с.
12. David Flanagan, and Gregor M. Novak Java-Script: The Definitive Guide, Second Edition / American Institute of Physics — 1998 — с.2-3 — Електронне видання: https://doi.org/10.1063/1.168647.
13. Denis Shestakov Databases on the web: national web domain survey. — 2011 — 156с. Електронне видання.
14. Эрик Редмонд, Джим. Р. Уилсон. Семь баз данных за семь недель. Введение в современные базы данных и идеологию NoSQL = MongoDB in Action. — ДМК Пресс, 2013. — 384 с.
15. Кайл Бэнкер. MongoDB в действии = MongoDB in Action. — ДМК Пресс, 2014. — 394 с.
16. Kristina Chodorow. MongoDB: The Definitive Guide, 2nd Edition. — O'Reilly, 2013. — 432 с.
17. Alessandro Del Sole. Visual Studio Code Succinctly. — SyncFusion Inc., 2016. — 128 с.
18. Swaminathan Sivasubramanian, Michal Szymaniak, Guillaume Pierre, Maarten van Steen. Replication for web hosting systems — 2004 — 291с. Електронне видання: https://doi.org/10.1145/1035570.1035573.
19. Camden, Raymond. Rinaldi, Brian (June 21, 2022). The Jamstack Book. Manning Publications. p. 138. ISBN 9781638356936.
20. So, Preston (September 9, 2021). Gatsby: The Definitive Guide. O'Reilly Media. p. 367. ISBN 9781492087489.
21. David Woodhouse. Quality and Quality Assurance — 1999 — 29с. Електронне видання: https://www.oecd-ilibrary.org/docserver/9789264173361-en.pdf?expires=1622926984&id=id&accname=guest&checksum=5BA644CBA8B5323F40998E7C4552C343#page=30

ДОДАТОК А

Рекомендаційна система оптимального вибору безпілотного літального апарата із заданими властивостями

Лістинг програми

Аркушів 19

Київ – 2023

auth.routes.js

```
const { Router } = require('express').
const bcrypt = require('bcryptjs').
const config = require('config').
const User = require('../models/User').
const jwt = require('jsonwebtoken').
const { check, validationResult } = require('express-validator').
const router = Router().

// /api/auth/rerister
router.post(
  '/register',
  [
    check('email', 'Некоректний email').isEmail(),
    check('password', 'Мінірисьна довжина паролю 6 символів').isLength({ min: 6 }),
  ],
  async (req, res) => {
    try {
      const errors = validationResult(req).

      if (!errors.isEmpty()) {
        return res.status(400).json({
          errors: errors.array(),
          message: 'Не коректні дані при реєстрації',
        }).
      }

      const { email, password } = req.body.

      const candidate = await User.findOne({ email }).

      if (candidate) {
        return res.status(400).json({ message: 'Такий користувач вже існує' }).
      }

      const hashedPassword = await bcrypt.hash(password, 12).
      const user = new User({ email, password: hashedPassword }).

      await user.save().

      res.status(201).json({ message: 'Користувач створен' }).
    } catch (e) {
      res.status(500).json({ message: e.message }).
    }
  },
).
```

```
// /api/auth/login
router.post(
  '/login',
  [
    check('email', 'Уведіть коректний email').normalizeEmail().isEmail(),
    check('password', 'Уведіть пароль').exists(),
  ],
  async (req, res) => {
    try {
      const errors = validationResult(req).

      if (!errors.isEmpty()) {
        return res.status(400).json({
          errors: errors.array(),
          message: 'Не коректні дані при вході у систему',
        }).
      }

      const { email, password } = req.body.

      const user = await User.findOne({ email }).

      if (!user) {
        return res.status(400).json({ message: 'Користувач не знайден' }).
      }

      const isMatch = await bcrypt.compare(password, user.password).

      if (!isMatch) {
        return res.status(400).json({ message: 'Неправильний пароль, спробуйте знову' }).
      }

      const token = jwt.sign({ userId: user.id }, config.get('jwtSecret'), { expiresIn: '1h' }).

      res.json({ token, userId: user.id }).
    } catch (e) {
      res.status(500).json({ message: e.message }).
    }
  },
).

module.exports = router.
```

bpla.routes.js

```
const { Router } = require('express').
const { Bpla } = require('../models/Bpla').
const router = Router().

router.get('/', async (req, res) => {
  try {
    const rangeParams = [].
    const valueParams = {}.

    for (let key in req.query) {
      if (key.includes('_min')) {
        const name = key.replace('_min', '').
        const min = parseInt(req.query[`${name}_min`]).

        rangeParams.push({ name, min }).
      } else if (key.includes('_max')) {
        const name = key.replace('_max', '').
        const max = parseInt(req.query[`${name}_max`]).

        rangeParams.push({ name, max }).
      } else if (key.includes('_str') && key !== 'text_str' && key !== 'sort_str') {
        const name = key.replace('_str', '').
        let value = req.query[key].
        if (!Array.isArray(value)) {
          value = [value].
        }
        valueParams[name] = value.
      } else if (key.includes('_num')) {
        const name = key.replace('_num', '').
        const value = parseInt(req.query[name]).
        valueParams.push({ name, value }).
      }
    }

    let queryToBD = null.

    if (req.query.text_str) {
      const regex = new RegExp(`.*${req.query.text_str}.*`, 'i').
      queryToBD = Bpla.find({
        $or: [
          { _name: { $regex: regex } },
          { model: { $regex: regex } },
          { shortDescription: { $regex: regex } },
          { description: { $regex: regex } },
        ],
```

```
    }).
   } else {
    queryToBD = Bpla.find().
   }

   if (rangeParams.length !== 0) {
    for (let param of rangeParams) {
     if (param.min) {
      queryToBD.where(param.name).gte(param.min).
     } else if (param.max) {
      queryToBD.where(param.name).lte(param.max).
     }
    }
   }

   if (valueParams) {
    for (let [name, params] of Object.entries(valueParams)) {
     queryToBD.find({ [`${name}`]: { $in: [...params] } }).
    }
   }

   if (req.query.sort_str) {
    queryToBD.sort({ [req.query.sort_str]: parseInt(req.query.order || 1) }).
   } else {
    queryToBD.sort({ _name: parseInt(req.query.order || 1) }).
   }

   const page = parseInt(req.query.page ?? 0). // start from 0 page
   const limit = parseInt(req.query.limit ?? 8).

   const countTotal = await Bpla.count(queryToBD).
   const listBpla = await queryToBD.skip(page * limit).limit(limit).

   if (!listBpla) {
    return res.status(404).json({ message: 'БПЛА не знайдені за заданими парарметрами' }).
   }

   res.json({ countTotal, listBpla }).
  } catch (e) {
   res.status(500).json({ message: e.message }).
  }
}).

router.get('/:id', async (req, res) => {
 try {
  const bpla = await Bpla.findById(req.params.id).

  if (!bpla) {
```

```
      return res.status(404).json({ message: 'Такий БПЛА не існує' }).
    }

    res.json(bpla).
  } catch (e) {
    res.status(500).json({ message: e.message }).
  }
}).

module.exports = router.
```

upload.routes.js

```
const { Router } = require('express').
const mongoose = require('mongoose').
let { Bpla } = require('../models/Bpla').
let { schemaBpla } = require('../models/Bpla').
const config = require('config').
const auth = require('../middleware/auth.middleware').
const multer = require('multer').
const path = require('path').

const storage = multer.diskStorage({
  destination: (req, file, cb) => {
    cb(null, 'uploads').
  },
  filename: (req, file, cb) => {
    cb(null, file.fieldname + '-' + Date.now()).
  },
}).

const upload = multer({ storage: storage }).

const router = Router().

router.put('/appendfield', auth, async (req, res) => {
  try {
    let newField = {
      [req.body.fieldName]: { type: String, default: req.body.defaultValue },
    }.
    if (typeof req.body.defaultValue == 'number') {
      newField = {
        [req.body.fieldName]: { type: Number, default: req.body.defaultValue },
      }.
    }
    schemaBpla.add(newField).

    const result = await Bpla.updateMany(
      {},
      { $set: { [req.body.fieldName]: req.body.defaultValue } },
    ).

    res.json(result).
  } catch (error) {
    res.status(500).json({ message: error.message }).
  }
}).
```

```
router.put('/deletefield', auth, async (req, res) => {
  try {
    const fields = await Bpla.find({ [req.body.fieldName]: { $exists: true } }).

    if (fields.length === 0) {
      return res.json('not found field in collection').
    }

    const result = await Bpla.updateMany(
      {},
      { $unset: { [req.body.fieldName]: req.body.defaultValue } },
    ).

    schemaBpla.remove([req.body.fieldName]).
    Bpla = mongoose.model('Bpla', schemaBpla).

    return res.json(result).
  } catch (error) {
    res.status(500).json({ message: error.message }).
  }
}).

router.post('/', auth, upload.array('photos'), async (req, res) => {
  try {
    const existing = await Bpla.findOne({ _name: req.body._name }).
    if (existing) {
      return res.status(400).json({ message: 'Такий БПЛА вже існує' }).
    }

    const photos = [].
    for (let file of req.files) {
      photos.push(config.get('baseUrl') + path.join('/uploads/' + file.filename)).
    }

    const bpla = new Bpla({
      photos: photos,
      _name: req.body._name,
      model: req.body.model,
      shortDescription: req.body.shortDescription,
      description: req.body.description,
      sourceUrl: req.body.sourceUrl,
      vendor: req.body.vendor,
      contryVendor: req.body.contryVendor ?? '',
      typeEngine: req.body.typeEngine ?? '',
      functions: req.body.functions,
      levelsApply: req.body.levelsApply ?? '',
      levelWarActions: req.body.levelWarActions ?? '',
      _class: req.body._class,
```

```
      flightRange: req.body.flightRange,
      wingspan: req.body.wingspan,
      maxFlyWeight: req.body.maxFlyWeight,
      payloadWeight: req.body.payloadWeight,
      maxSpeed: req.body.maxSpeed,
      cruiseSpeed: req.body.cruiseSpeed,
      maxFlyHeight: req.body.maxFlyHeight,
      heightOfUse: req.body.heightOfUse,
      flyDuration: req.body.flyDuration,
    }).

    bpla.save().

    res.status(201).json({ id: bpla._id }).
  } catch (error) {
    res.status(500).json({ message: error.message }).
  }
}).

module.exports = router.
```

Bpla.js

```
const { Schema, model } = require('mongoose').

let schemaBpla = new Schema({
  date: { type: Date, default: Date.now },
  photos: [{ type: String }],
  _name: { type: String, required: true, unique: true },
  model: { type: String, default: '' },
  shortDescription: { type: String, default: '' },
  description: { type: String, default: '' },
  sourceUrl: { type: String, default: '' },
  vendor: { type: String, default: '' },
  contryVendor: { type: String, default: '' },
  typeEngine: { type: String, default: '' },
  functions: [{ type: String }],
  levelsApply: [{ type: String }],
  levelWarActions: { type: String, default: '' },
  _class: { type: String, default: '' },
  flightRange: { type: Number, default: 0 },
  wingspan: { type: Number, default: 0 },
  maxFlyWeight: { type: Number, default: 0 },
  payloadWeight: { type: Number, default: 0 },
  maxSpeed: { type: Number, default: 0 },
  cruiseSpeed: { type: Number, default: 0 },
  maxFlyHeight: { type: Number, default: 0 },
  heightOfUse: { type: Number, default: 0 },
  flyDuration: { type: Number, default: 0 },
}).

schemaBpla.index({ _name: 'text', model: 'text', shortDescription: 'text', description: 'text' }).

const Bpla = model('Bpla', schemaBpla).

Bpla.createIndexes()
  .then(() => console.log('Index created successfully'))
  .catch((error) => console.log(error)).

module.exports = { Bpla, schemaBpla }.
```

User.js

```
const { Schema, model } = require('mongoose').

const schema = new Schema({
  email: { type: String, required: true, unique: true },
  password: { type: String, required: true },
}).

module.exports = model('User', schema).
```

auth.middleware.js

```
const jwt = require('jsonwebtoken').
const config = require('config').

module.exports = (req, res, next) => {
  if (req.method === 'OPTIONS') {
    return next().
  }

  try {
    const token = req.headers.authorization.split(' ')[1]. // 'Bearer TOKEN'
    if (!token) {
      return res.status(401).json({ message: 'Немає авторизації' }).
    }

    const decoded = jwt.verify(token, config.get('jwtSecret')).
    req.user = decoded.
    next().
  } catch (e) {
    return res.status(401).json({ message: 'Немає авторизації' }).
  }
}.
```

cors.middleware.js

```
function cors(req, res, next) {
  res.header('Access-Control-Allow-Origin', '*').
  res.header('Access-Control-Allow-Methods', 'GET, PUT, PATCH, POST, DELETE').
  res.header('Access-Control-Allow-Headers', 'Content-Type, Authorization').
  next().
}

module.exports = cors.
```

default.json

```
{
  "port": 5000,
  "jwtSecret": "bpla secret_jwt",
  "mongoUri":
"mongodb+srv://bpla_user:bpla_user_password@cluster0.qmdno9j.mongodb.net/?retryWrites=tr
ue&w=majority",
  "baseUrl": http://localhost:5000
}
```

production.json

```
{
  "port": 80,
  "jwtSecret": "bpla production-secret_jwt",
  "mongoUri":
"mongodb+srv://bpla_user:bpla_user_password@cluster0.qmdno9j.mongodb.net/?retryWrites=tr
ue&w=majority",
  "baseUrl": "http://localhost:5000"
}
```

app.js

```
const express = require('express').
const config = require('config').
const mongoose = require('mongoose').
const path = require('path').
const corsMiddleware = require('./middleware/cors.middleware').

const app = express().

app.use(corsMiddleware).
app.use(express.json({ extended: true })).

app.use('/api/auth', require('./routes/auth.routes')).
app.use('/api/upload', require('./routes/upload.routes')).
app.use('/api/bpla', require('./routes/bpla.routes')).

app.use(express.static('public')).
app.use('/uploads', express.static('uploads')).

if (process.env.NODE_ENV === 'production') {
  app.use('/', express.static(path.join(__dirname, 'client', 'build'))).

  app.get('*', (req, res) => {
    res.sendFile(path.resolve(__dirname, 'client', 'build', 'index.html')).
  }).
}

const PORT = config.get('port') ?? 5000.

async function start() {
  try {
    await mongoose.connect(config.get('mongoUri'), {
      useNewUrlParser: true,
      useUnifiedTopology: true,
    }).
    app.listen(PORT, () => console.log(`app has been started on port ${PORT}...`)).
  } catch (e) {
    console.log('Server Error', e.message).
    process.exit(1).
  }
}
```

start().

package.json

```
{
  "name": "mern",
  "version": "1.0.0",
  "description": "mern lern",
  "main": "app.js",
  "scripts": {
    "start": "cross-env NODE_ENV=production node app.js",
    "server": "nodemon app.js",
    "client": "npm run start --prefix client",
    "client:install": "npm install --prefix client",
    "client:build": "npm run build --prefix client",
    "dev": "cross-env NODE_ENV=development concurrently \"npm run server\" \"npm run client\""
  },
  "keywords": [
    "mern",
    "react"
  ],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "bcryptjs": "^2.4.3",
    "config": "^3.3.9",
    "express": "^4.18.2",
    "express-validator": "^6.15.0",
    "jsonwebtoken": "^9.0.0",
    "mongoose": "^7.0.3",
    "multer": "^1.4.5-lts.1"
  },
  "devDependencies": {
    "concurrently": "^8.0.1",
    "cross-env": "^7.0.3",
    "nodemon": "^2.0.22"
  }
}
```

index.html

```
<!DOCTYPE html>
<html lang="ua">
  <head>
    <meta charset="utf-8" />
    <link rel="shortcut icon" href="%PUBLIC_URL%/favicon.ico" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <meta name="description" content="Сайт для пошуку БПЛА" />
    <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo.png" />
    <title>Пошук БЛПА</title>
  </head>
  <body>
    <noscript>You need to enable JavaScript to run this app.</noscript>
    <div id="root"></div>
  </body>
</html>
```

manifest.json

```
{
  "short_name": "Пошук БПЛА",
  "name": "Сервіс для пошуку необхідного БПЛА",
  "icons": [
    {
      "src": "favicon.ico",
      "sizes": "64x64 32x32 24x24 16x16",
      "type": "image/x-icon"
    },
    {
      "src": "logo.png",
      "type": "image/png",
      "sizes": "512x512"
    }
  ],
  "start_url": ".",
  "display": "standalone",
  "theme_color": "#000000",
  "background_color": "#ffffff"
}
```

App.css

```
*::-webkit-scrollbar {
  width: 5px.
  height: 5px.
}

/* Track */
* ::-webkit-scrollbar-track {
  background: transparent.
}

/* Handle */
*::-webkit-scrollbar-thumb {
  background: #6766664e.
  border-radius: 3px.
}

/* Handle on hover */
*::-webkit-scrollbar-thumb:hover {
  background: #555555a2.
}
```

App.jsx

```
import './App.css'.

import { BrowserRouter as Router } from 'react-router-dom'.
import { useRoutes } from './routes'.
import { useAuth } from './hooks/auth.hook'.
import { AuthContext } from './context/context'.
import { ConfigContext } from './context/configContext'.
import Loader from './components/Loader'.

import { ThemeProvider } from '@mui/material/styles'.
import useTheme from './hooks/theme.hook'.

function App() {
  const { token, login, logout, userId, ready, isAuthenticated } = useAuth(). // bplaPassAdmin -
password. admin123@gmail.com email
  const routes = useRoutes(isAuthenticated).
  const { themeDark, themeLight, modeView, setModeView } = useTheme().

  if (ready === false) {
    return <Loader />.
  }

  return (
    <ConfigContext.Provider value={{ modeView, setModeView }}>
      <AuthContext.Provider
        value={{
          token,
          login,
          logout,
          userId,
          isAuthenticated,
        }}>
        <ThemeProvider theme={modeView === 'light' ? themeLight : themeDark}>
          <Router>{routes}</Router>
        </ThemeProvider>
      </AuthContext.Provider>
    </ConfigContext.Provider>
  ).
}

export default App.
```

axios.common.js

```
import axios from 'axios'.

export default axios.create({
  baseURL: 'http://localhost:5000/api',
  headers: {
    'Content-type': 'application/json',
  },
}).
```

index.css

```
body {
  font-family: 'Roboto', sans-serif.
}
```

index.js

```
import React from 'react'.
import ReactDOM from 'react-dom/client'.
import './index.css'.
import App from './App'.

const root = ReactDOM.createRoot(document.getElementById('root')).
root.render(<App />).
```

routes.jsx

```
import CreatePage from './pages/CreatePage'.
import DetailPage from './pages/DetailPage'.
import AuthPage from './pages/AuthPage'.
import MainPage from './pages/MainPage'.

import React from 'react'.
import { Routes, Route, Navigate } from 'react-router-dom'.
import { useFormData } from './hooks/formData.hook'.

import Navbar from './components/Navbar'.
import { Container, Paper } from '@mui/material'.
import CssBaseline from '@mui/material/CssBaseline'.

export function useRoutes(isAuthenticated) {
  const { FormProvider } = useFormData().

  if (isAuthenticated) {
    return (
      <>
        <Navbar displaySearch={false} displayLogout={true} />
        <Container maxWidth="xl">
          <CssBaseline />
          <Paper sx={{ minHeight: 'calc(100vh - 5rem)', mt: '4.5rem' }}>
            <FormProvider>
              <Routes>
                <Route path="/create" exact element={<CreatePage />} />

                <Route path="*" element={<Navigate to="/create" />} />
              </Routes>
            </FormProvider>
          </Paper>
        </Container>
      </>
    ).
  }

  return (
    <>
      <Routes>
        <Route path="/login" exact element={<AuthPage />} />
        <Route path="/detail/:id" element={<DetailPage />} />
        <Route path="/" exact element={<MainPage />} />
```

```
        <Route path="*" element={<Navigate to="/" />} />
      </Routes>
    </>
  ).
}
```

ДОДАТОК Б

Рекомендаційна система оптимального вибору безпілотного літального апарата із заданими властивостями

Опис програмного коду

Аркушів 9

Київ – 2023

АНОТАЦІЯ

Даний додаток надає інформацію про веб-сервіс, що є програмним засобом пошуку та перегляду БПЛА із заданими властивостями.

Веб-сервіс надає можливість користувачам сервісу:

— здійснити авторизацію для створення нового БПЛА та додавання його до БД. Дану функціонал використовує лише адміністратор сайту.

— здійснювати перегляд всіх наявних на сайті БПЛА.

— здійснювати пошук потрібного БПЛА за певними критеріями та параметрами.

— здійснити сортування, за певним параметром чи критерієм, вибраних БПЛА.

Веб-сервіс створений з використанням мови програмування JavaScript та бібліотеки React, що призначена для розробки інтерфейсів, під платформу Web-браузера.

ЗАГАЛЬНІ ВІДОМОСТІ

Відповідно до назви дипломної роботи, проект має такий опис: “Рекомендаційна система вибору безпілотного літального апарату із заданими властивостями”.

Для користування розробленим веб-сервісом необхідно мати встановлений Інтернет-браузер та доступ до мережі. Операційна система ролі не грає.

Веб-сервіс створений з використанням мови програмування JavaScript та бібліотеки React, що призначена для розробки інтерфейсів, під платформу Web-браузера.

ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ

Веб-сервіс призначений для вирішення проблеми якісного та швидкого пошуку БПЛА із заданими властивостями. Програмний продукт має наступний функціонал:

1. Можливість переглядати всі наявні на сайті БПЛА. Дані БПЛА заносяться на сайт його адміністратором.
2. Можливість здійснювати пошук потрібного БПЛА за допомогою певних критеріїв та параметрів.
3. Можливість сортувати БПЛА, на головній сторінці сайту, за певним параметром чи критерієм. Наприкад, за алфавітом.
4. Можливість авторизації, для адміністратору сайту, для переходу в адмін панель (сторінку). За допомогою даної сторінки будуть додаватися нові БПЛА в БД сайту, для їх подальшого виведення на головну сторінку сайту.
5. Можливість для додавання нового БПЛА в БД сайту, з метою їх подальшого виведення на головну сторінку сайту.
6. Можливість для додавання нового БПЛА в БД сайту, з метою їх подальшого виведення на головну сторінку сайту.

ОПИС ЛОГІЧНОЇ СТРУКТУРИ

Загальна логічна структура веб-сервісу така:

1. Користувач потрапляє на сервіс, де виводяться всі БПЛА, що наявні в БД сайту.
2. Користувач використовує функцію «Пошуковий рядок» або «Списки фільтрів» для знаходження БПЛА із заданими властивостями.
3. Система вивантажує, з бази даних, таблицю з даними про БПЛА, які потрібно знайти користувачу
4. Користувач має змогу переглянути конкретний БПЛА із списку знайдених. Для цього він переходить на сторінку з даним БПЛА.
5. Система вивантажує, з бази даних, дані про конкретний БПЛА та виводить всю інформацію на окрему HTML-сторінку.

ВИКОРИСТАНІ ТЕХНІЧНІ ЗАСОБИ

Для використання даного веб-сервісу необхідно мати пристрій зі встановленим браузером та доступом до мережі Інтернет.

Операційна система пристрою не важлива.

ВИКЛИК І ЗАВАНТАЖЕННЯ

Для користування веб-сервісом необхідно перейти за посиланням https://bpla-client.vercel.app/. За даним посиланням користувач має можливість здійснювати перегляд та пошук БПЛА із заданими властивостями.

Для створення нового БПЛА та додавання його до БД сайту – адміністратор сайту повинен перейти за посиланням https://bpla-client.vercel.app/login, пройти процес авторизації та здійснити процедуру створення та додавання.

ВХІДНІ ДАНІ

Вхідними даними створеного веб-сервісу є дані, які вводить користувач в пошуковий рядок сайту або ж такими даними – є список параметрів, що вибирає користувач в процесі пошуку БПЛА. Дані параметри знаходяться в списку фільтрів веб-сервісу.

ВИХІДНІ ДАНІ

Вихідними даними створеного веб-сервісу є повідомлення виведення на екран користувача знайдених БПЛА із заданими властивостями.