Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Медіасистем та технологій
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Створення веб-сайту для організації інтернет трансляцій і прийому пожертв
(тема)

Виконав:
студент 4 курсу, групи ВПВПС-20-3

Науменко Р.О.
(прізвище, ініціали)

Спеціальність 186 Видавництво та поліграфія
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма
Видавничо-поліграфічна справа
(повна назва освітньої програми)

Керівник доц Дашкевич А.О.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри МСТ _____ Дейнеко Ж.В.
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
Кафедра Медіасистем та технологій
Рівень вищої освіти перший (бакалаврський)
Спеціальність 186 Видавництво та поліграфія
Тип програми Освітньо-професійна
Освітня програма Видавничо-поліграфічна справа
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри МСТ_____
(підпис)
« 20 » травня 2022 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові *Науменку Роману Олеговичу*
(прізвище, ім'я, по батькові)

1. Тема роботи Створення веб-сайту для організації інтернет трансляцій і прийому пожертв

Затверджена наказом по університету від 20 травня 2024р. №458 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 6 червня 2024 р.

3. Вихідні дані до роботи
Вид і призначення Веб-видання: веб-додаток; призначення: веб-додаток для організації відео-трансляцій і прийому пожертв; Мови розробки веб-видання: HTML, CSS, JavaScript, Node.JS; використанні фреймворки: Material CSS; середовище розповсюдження: Інтернет
4. Перелік питань, що потрібно опрацювати в роботі
Вступ. Аналіз завдання на кваліфікаційну роботу; Визначення цільової авдиторії; Аналіз аналогів; Визначення функціональності; Визначення логіки взаємодії; Обґрунтування вибору програмного забезпечення; Розробка модульної сітки; Обґрунтування дизайнерського рішення; Розробка дизайн-макету; Прототипуваня; Наповнення контентом сторінок веб-видання; Тестування та публікація; Економічна частина; Висновок;
5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п. 5 включається до завдання за рішенням випускової кафедри)
Титульний слайд презентації; Актуальність та мета; Задачі роботи; Аналіз цільової авдиторії; Аналіз аналогів; Визначення взаємодії; Модульна сітка; Кольорове рішення; Шрифтове рішення; Логотип; Дизайн-макет; Прототипування; Розробка; Економіка; Висновок

6. Консультанти розділів роботи (п. 6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п. 1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Основна частина	доц Дашкевич А.О.		05.06.2024
Економічна частина	ас. Помогалова Н.В.		04.06.2024

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи		Примітка
1	Отримання завдання	20.05.2024	Викон.
2	Аналіз завдання на кваліфікаційну роботу, визначення цілей і задач проєктування	21.05.2024	Викон.
3	Аналітичний аналіз технологій пов'язанних із темою розробки	22.05.2024	Викон.
4	Проєктування інформаційної структури та логіки взаємодії	24.05.2024	Викон.
5	Розробка дизайнерського рішення	26.05.2024	Викон.
6	Розробка веб-сайту на основі дизайн-макету	29.05.2024	Викон.
7	Економічна частина	29.05.2024	Викон.
8	Оформлення пояснювальної записки	03.06.2024	Викон.
9	Оформлення графічної частини	03.06.2024	Викон.

Дата видачі завдання число 20.05.2024 р.

Студент  Науменко Р.О.
(підпис)

Керівник роботи  доц. Дашкевич А.О.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 56 с., 4 табл., 20 рис. 2 дод., 43 джерела.

ВЕБ-САЙТ, ВЕБ-ДОДАТКИ, ІНТЕРНЕТ-ТРАНСЛЯЦІЇ, ПОЖЕРТВИ, РОЗРОБКА, ІНТЕРНЕТ, ВІДЕО, ДИЗАЙН.

Робота присвячена дослідженню та створенню веб-додатку для проведення інтернет-трансляцій та прийому сповіщень про пожертви під час них. Метою роботи є аналіз технологій пов’язаних із створенням подібного вебсайту, а також розробки графічного інтерфейсу для нього.

У роботі надано детальний опис розробки подібного веб-ресурси, описані етапи розробки користувацького інтерфейс, що були визначені на основі аналізу аналогів, та потреб цільової авдиторії. Також наявна інформація про усі сучасні технології для побудови сучасного адаптивного веб-додатку.

Пояснювальна записка містить розгорнутий опис усіх етапів розробки, надано результат створеного графічного інтерфейсу та фрагменти коду.

ABSTRACT

Explanatory note of the qualification work: 56 p., 4 tabl., 20 pic., 2 app., 43 sources.

WEB-APP, WEB-SITE, INTERNET-BROADCASTING, DONATIONS, DEVELOPMENT, INTERNET, VIDEO, DESIGN

This qualification work dedicated to exploration and development of Web Application for conduction internet-broadcasts and collecting donations during it. The aim of this work is analysis of technologies related with creating of this Web-site and also about creating User Interface for this.

This work has a detailed description of development of web-site kind of this, was described all steps of development UI that was based on analogs analysis and target audience analysis. Also there is all modern teechnologys of creating modern responsive web-app.

The exploration note contains detailed desription of all steps, fragment of created graphical interface and code are included.

ЗМІСТ

С.

ВСТУП

Сучасний світ складно уявити без використання веб-додатків [1]. Вони дають можливість незалежно від програмної платформи і на якому комп’ютері виконується задача, виконувати проєкт, даючи можливість людині бути значно більш мобільною і не мати прив’язки до конкретного обладнання, це також спрощує розробку, коли достатньо написати програму всього для однієї платформи, замість декількох на сьогодні актуальних. Також подібні додатки дозволяють не витрачати час на налаштування та встановлення на конкретний комп’ютер.

Провідним питанні у розробці подібних проєктів є проєктування та створення веб-інтерфейсу. Інтерфейс є конче важливим елементом будь чого: від веб-сайту і закінчуючи настільними програмами. Саме від нього залежить, чи буде користувач використовувати ваш продукт чи ні [2]. Чи зможе він виконувати поставленні задачі легко і обере саме ваш продукт поміж конкурентів.

Метою даної роботи є проєктування та створення інтерфейсу веб-сайту, метою якого є проведення відеотрансляцій в інтернеті із можливістю прийому пожертв. Графічний інтерфейс, що створюється має відповідати великій кількості вимог, аби бути зручним, простим у використанні, доступним та зовнішньо привабливим. Головною метою у розробці інтерфейсів є чіткий аналіз потреб користувача, формулювання переліку задач, які хоче досягти користувач, завдяки продукту, що проєктується, і яким чином він очікує, щоб дані запити були реалізовані [3]. Інтерфейс мусить не тільки надавати доступ до цих функцій, а і бути інтуїтивно зрозумілим, аби користувач відразу розумів, куди він мусить натискати задля отримання результату.

Інтерфейс мусить бути естетично привабливим, підтримувати власну цілісність, швидко вантажитися в пам’ять пристрою та адаптовуватися під

обставини в яких він відображається. Він має бути доступним і зрозумілим широкому колу осіб.

В період активного розвитку інтернету та медіа технологій, що пов’язані із створенням інтернет контенту, розробка веб-сайту задля проведення відео-трансляцій має високу актуальність [4]. Існує потреба в створенні подібного сайту, так як щороку збільшується кількість блогерів, які мають бажання проводити інтернет трансляції, задля підтримки контакту із власною аудиторією, або обговорення якоїсь важливої інформації. Часто такі виходи в етер дозволяють організувати збір коштів від власної аудиторії на підтримку проєкту або задля іншої мети.

Однак альтернативи, що існують, є недосконалими. Їх налаштування може бути надто складним, інтерфейс не інтуїтивним для більшості користувачів, які зазвичай не мають високого рівня обізнаності в організації подібного. Інші конкуренти можуть вимагати високу ціну за власні послуги.

Створення оптимального проєкту, що вирішуватиме вищеописані проблеми і визначають актуальність даної роботи.

Виконання даної роботи передбачає:

– аналіз завдання на кваліфікаційну роботу, розгляд основних тенденцій в галузі, що стосуються розробки веб-додатків;

– розгляд особливостей, що необхідно врахувати при побудові проєкту. Розгляд аналогів сайту, що створюється;

– визначення цільової авдиторії. Формулюваня списку функцій, встановлення їх пріоритетності;

– вибір програмного забезпечення для виконання завдання;

– проведення UX-досліджень, формування стилю проєкту, логіки відповідної взаємодії;

– розробка дизайн-макет та прототипу сайту, що створюється;

– верстання макету, реалізація сайту;

– перевірка працездатності продукту, розміщення в інтернеті;

– оцінка економічної складової продукту, що створюється.

Відповідно до сформульованих задач, що поставленні визначається структура даної кваліфікаційної роботи.

Об’єктом дослідження є розробка вищеописаного веб-додатку, предметом – проведення UX-досліджень, створення UI (інтерфейсу конкретного користувача).

Вступ містить інформацію про аспекти, які необхідно врахувати, актуальність проблеми, об’єкт та мету дослідження, мету, сформульовані задачі, а також структура даної роботи.

У першому розділі виконано аналіз задач поставлених кваліфікаційною роботою, визначено мету розроблюваного сервісу.

В другому розділі проведено аналітичний аналіз технологій, які необхідно застосувати для створення запланованого веб-сайту.

Далі, визначено цільову авдиторію, проведено UX-дослідження, визначено перелік функцій, розглянуто та виявлені переваги та недоліки аналогів, зроблено вибір програмного забезпечення. Спроєктовано структуру продукту, навігацію, опис логіки взаємодії.

У четвертому розділі проведено аналіз засобів для створення розроблюваного продукту.

У п’ятому розділі визначенню стильові властивості продукту, розроблена модульна сітка, на основі попередніх пунктів створено дизайн макет та прототип.

У шостому розділі описано процес розробки продукту на основі дизайн-макету.

Наприкінці наведено економічне обґрунтування реалізації проєкту.

1 АНАЛІЗ ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

Метою даної кваліфікаційною роботи є створення веб-додатку, для проведення трансляцій в інтернеті, які можна буде переглянути на таких сервісах, як наприклад Youtube або Twitch, а також з можливістю сповіщувати користувача про надходження грошових пожертв в авдіо та візуальний спосіб під час трансляції.

Опціонально також доступна можливість прийому сповіщень, які глядачі відправляють в чаті під час перегляду.

Основною метою є спрощення досвіду звичайного користувача, зробити налаштування подібного доступним і простим. Особливо ця задача є складною, коли існує потреба влаштувати декілька трансляцій водночас, наприклад на Youtube та Telegram, аби залучити більшу аудиторію, що часто є нетривіальною задачею.

Важливою також задачею може бути уніфікація сповіщень про внески, наприклад, коли є потреба приймати пожертви на картковий рахунок, сервіс прийому пожертв, або платформа для монетизації водночас.

Даний застосунок має працювати через веб-браузер, аби спростити розробку і зробити його доступним на всіх актуальних комп’ютерних платформах тощо.

Влаштування даного сервісу, попри вищеописане є насправді досить легким. Усі сучасні сервіси відеотрансляцій, або такі, що підтримують подібний тип контенту, як то Youtube, Twitch, Telegram, Facebook, приймають відеопотік по стандартизованому інтернет протоколу, так званий RTMP. На боці клієнту сайт мусить проводити запис відео та авдіо контенту, що є досить легким у реалізації, утворювати з джерел єдиний відео-потік, і передавати на сервер, що кодуватиме його та відправлятиме за місцем призначенням. Для отримання цього «місцяпризначення» необхідно за допомогою інструкцій, які надають кожен із цих сервісів, підключитися до них та отримати адресу необхідного каналу.

Для прийому внесків існує також протокол Webhook, сервіси, що приймають пожертви, як правило мають реалізацію цього протоколу. Користувачу достатньо вказати їм унікальний ключ, що генеруватиме програма, після чого всі ці сервіси будуть відправляти дані по цьому каналу.

Для прийому сповіщень, які користувачі пишуть в чати трансляцій, також має реалізовуватися за допомогою API, що надають дані сервіси.

Задачею веб-додатку буде декодування даних і передача користувачу у правильній формі. Важливо надати зручний та інтуїтивно зрозумілий інтерфейс, аби користувач міг легко керувати цими процесами.

Етапи розробки веб-додатку не сильно відрізняється від створення інших типів сайтів:

1) аналіз цільової аудиторії, створення портрету споживача;
2) визначення важливість тих чи інших функцій, методи рішення поставлених користувачем задач;
3) створення структури та навігації;
4) розробка дизайну – вибір колірної схеми, шрифтового оформлення;
5) створення дизайн-макету, прототипування;
6) верстання макету програмування взаємодії;
7) тестування та виправлення;
8) публікація в інтернеті, підтримка.

2 АНАЛІТИЧНИЙ АНАЛІЗ ТЕХНОЛОГІЙ ПОВ'ЯЗАННИХ ІЗ ТЕМОЮ РОБОТИ

2.1 Веб-додаток

Веб-додаток – це вид програмного забезпечення, що являє собою поєднання функціональності звичайних програм, такі як масштаб доступних функцій і переваг веб-сайтів, такі як легкість, і відсутність прив'язки до конкретного виду пристроїв, а відповідно необхідності встановлення [1].

Межа між веб-додатком та сайтом на сьогодні досить умовна. Прийнято вважати, що веб-сайтом є такий веб-ресурс, який пропонує тільки отримання вже заготовленної інформації користувачем, але не її обробку в будь-який спосіб. В свою чергу, веб-додаток пропонує обидва способи взаємодії із інформацією. Тож, усі веб-додатки, по своїй суті, є веб-сайтами, але не усі веб-сайти є веб-додатками. Часто, можна зустріти веб-ресурси, які мають властивості, як веб-сайтів, так і веб-додатків. Так, наприклад, блог-платформа працює як веб-сайт надаючи статті для прочитання, однак можливість створювати нові пости простими користувачами, на відміну від, наприклад ЗМІ, робить їх веб-додатками. Складно назвати чим є багато елементів так званого Web 1.0, такі як форуми або іміджборди. Чи робить сайт веб-додатком наявність коментарів? Натомість, сторінки справки до веб-додатків можна класифіковувати як веб-сайти.

Тим не менш, поява, і в першу чергу, активне поширення веб-додатків, пов'язане із поширенням стабільного та швидкого інтернету, замість повільного та нестабільного з'єднання по мобільній мережі, поява глобального інтернет-пошуку, коли користувачі припинили, в основному, користуватися сервісами, що надавав інтернет-провайдер. А головне, поява таких веб-технологій, як HTML5, CSS, JavaScript, Flash, ActiveX, PHP, Node.JS, WebAssembly, Apache та багато інших [6].

Найбільш важливим кроком в утворенні сучасних веб-додатків відіграла поява мови сценаріїв JavaScript в 1995 році [5], як частина браузера Netscape Navigator (сучасний Mozilla Firefox).

З того часу, мова перетворилася у повноцінну мову програмування, на якій розробляються додатки, далеко за межами користувацьких сценаріїв з боку веб-сайтів. Завдяки своїй гнучкості, простоті та відкритості, стала промисловим стандартом, витіснивши більш громіздкий Flash від Adobe та QuickTime від Apple.

JavaScript пристосували для виконанні сценаріїв не тільки на боці клієнтів, а і на боці серверу. З'явився Node.JS, який потіснив популярні на серверах мови PHP та PERL, що також використовуються для побудови серверних частин веб застосунків.

Важливим кроком для оформлення сучасних веб-додатків стала поява HTML5 та CSS3, які зробили верстання подібних застосунків більш простою задачею. З'явилися відносні розміри, що дозволило робити сайти адаптивними, та сіточна верстка, на противагу застарілої табличної, яка мала багато недоліків. Це дозволило зробити один із найбільш простих і лаконічних способів верстання інтерфейсів, який використовується далеко за межами інтернету.

Все це дозволяє створювати замітки, зберігати файли в інтернеті, дивитися відео, працювати з документами, таблицями та презентаціями, редагувати фото та відео, та навіть грати у ігри не покидаючи веб-браузер.

2.2 Адаптивність

Проблемам черезмірно широкого спектру пристроїв, через які можна переглядати веб-сторінки, завжди була актуальна для інтернету. На його зародженні, розробники мусили вказувати в якому браузерів, під якою операційною системою, та в якій роздільній здатності, слід переглядати веб-сайти, аби весь зміст передавався правильно.

Часто на сайтах можна було зустріти спеціальні позначки про те, що сайт гарантовано працюватиме, наприклад, в Internet Explorer або Netscape. Але розробник не дає гарантій, що він так само добре відображатиметься, наприклад, в Opera.

Із поширенням КПК, а пізніше смартфонів, ситуація ускладнилася. На ринку почали з’являтися пристрої з обмеженим споживанням пам’яті, низькою роздільною здатністю, а що гірше, із нетрадиційним відношенням сторін. Водночас і на ПК, класичні ЕПТ-монітори з відношенням у 4:3х почали замінювати більш сучасні із відношення 16:9.

Це робило перегляд сайтів часто, в кращому випадку, незручним, а не рідко і просто неможливим. Навіть проста зміна розміру вікна могла зламати структуру сайту.

Спочатку розробники намагалися дублювати сайти для деяких типів пристроїв (такі сайти називаються адаптованими), однак це значно ускладнювало, як розробку, так і підтримку подібних сайтів [7].

Із появою HTML5 та CSS3, ситуація змінилася. Серед UI/UX дизайнерів поширилася концепція колонкової модульної сітки [8]. Такий підхід не тільки допоміг зробити сайти більш охайними та збалансованими, він також допоміг в забезпеченні адаптивності в межах однієї сторінки. Такі сайти називаються чуйними(від англійського Responsive) [35].

За допомогою технології медіа-запитів, що з’явилася в CSS3, появи елементу div (контейнер) в HTML5, а також режимів відображення flex та grid, для створення адаптивності тепер можливо просто приховувати частину колонок або змінювати кількість колонок яку займає той чи інший контент на сторінці, при відображенні у різних умовах, починаючи від нестандартних роздільних здатностей і закінчуючи принтерами.

Багато сучасних фронтенд-фреймворків, такі як Bootstrap, TailWind, Material CSS або Framework підтримують 12-колонкову верстку «прямо з коробки» тощо.

У питанні, як управляти подібною сіткою, існує три основні підходи. Вони визначають як буде організована адаптивність і який підхід використовуватиметься у першу чергу.

– mobile first [9] – в такому разі, в першу чергу, верстається мобільна версія сторінки, адаптована до низької роздільної здатності, обмеженної пам’яті та повільного інтернет-з’єднання. Після чого додаються елементи, що будуть доступні на пристроях з кращими характеристиками і більшими екранами. Це не означає, що сайт має бути заточеним під мобільні пристрої. Mobile First проєктуванням вважається, коли ви орієнтуєтеся на те, як виглядатиме сайт, якщо наприклад, браузер відкритий у маленькому вікні;

– desktop first [9] – в цьому разі спочатку розробляється повноцінна версія сайту адаптована під найбільшу роздільну здатність та максимальні характеристики, а пізніше з нього «викидаються» ті елементи, які не варто відображати на менших пристроях. Цей підхід користується меншою популярність, на відміну від Mobile First, в основному через більшу складність, однак часто, особливо коли стоїть задача створити веб-додаток, де більш прості пристрої можуть виконувати обмежений набір функцій, такий підхід може бути виправданим;

– content-out [10] – підхід при якому, робиться акцент не як змусити зайняти контент максимально доступний простір, а як зробити його максимально зрозумілим. Такий підхід є більш складним, ніж попередні, і часто орієнтується на пропорційність контенту і ступінь його важливості.

2.3 Мови дизайну

Розвиток дизайну неодмінно веде до наробки якихось спільних практик, які виглядають актуально і стильно в певний момент часу. Люди створюють нові дизайни, інші дизайнери та користувачі бачать результат і бажають або очікують, в залежності від ролі відповідно, побачити аналогічні

рішення в інших дизайнах. Таким чином укладаються мови дизайну, і дизайн-системи, як їх проізводні.

Часто подібні мови розробляють великі та впливові компанії, є частинами дизайну тих чи інших мобільних та настільких операційних систем тощо.

Так, наприклад, в другій половині нульових популярність мала мова дизайну Frutiger Aero. Цей стиль використовувався в Windows Vista, та пізніше у Windows 7, у ранніх версіях Mac OS X, IOS, оболонці KDE для UNIX-подібних систем. Також він був характерним і для величезної кількості веб-сайтів того періоду. Йому характерні яскраві кольори, скевоморфизм, градієнти, ефект скла, напівпрозорість.

Її замінили більш мінімалістичні мови дизайну, як відповідь на тренд до мінімалізму. Однак багато хто розумів його по своєму, що породило декілька нових мов дизайну. Це Metro від Майкрасофт, що використовувався в Windows 8, Windows 10, Windows Phone, Xbox 360 і Xbox One. Це Apple Human Interface Guidelines від Apple, що використовувався в IOS версії 7 та вище, а також Mac OS з версії 10.10 по 10.16. Або Material UI від Google, що використовувався на системі Android з версії 5 по версію 11. Особливо остання була дуже популярна, як мова дизайну і зустріти її використання можна було в дуже далеких від Android царинах, в тому числі і в галузі розробки веб-сайтів.

На сьогодні можна також зустріти багато дизайн мов. Їх використання не є чіткими вимогами, а скоріше рекомендаціями, як має виглядати додаток, однак їх використання може значно спростити, як розробку додатку: не треба продумувати, як за різних умов виглядатимуть кнопки, як забезпечити цілісність, а вже використовувати готові, або адаптовані під продукт рішення, так і стандартизувати дизайн, роблячі інтерфейс більш звичним для користувача. Часто актуальні мови дизайну вже відображають актуальні віяння в галузі.

На сьогодні, в моді знову скевоморфізм, мінімалізм став менш радикальним і допускає використання об’ємних образів, текстур та яскравих кольорів, а досягнення об’єму не тільки за допомогою тіней та контрастів [15].

І це відображається і на актуальних мовах дизайну. Наразі існує декілька актуальних стандартизованих мов дизайну:

Google Material You [11] – як зрозуміло із назви, цей стиль розробляється і підтримується корпорацією Google. Ця мова активно застосовується в Android та сервісах пошукового гіганту. Його характерними рисами є мінімалізм, використання яскравих кольорів, велике значення надається тіням.

Своєю мовою взаємодії володіє і Apple [12]. Вона називається Human Interface Guidelines. Її використання є обов’язковим в додатках для техніки Apple. Apple надає більше уваги градієнтам, і значно меншу грі тіней, на відміну від мови Google. Він також допускає обмеженний об’єм, використання 3d та псевдо 3d.

Microsoft Fluent Design [13] – використовується корпорацією Microsoft. Використовується в операційній системі Windows 11, ігровій консолі Xbox S та багатьох інших програмних продуктах корпорації Microsoft. Вона використовує переливання градієнтів, напівпрозорість, як основу інтерфейсу.

Gnome Human Interface Guidelines [14] – активно використовується в UNIX-подібних операційних системах, таких як Linux або FreeBSD. Основою цієї мови є контраст кольорів, скевоморфизм та мінімалістичність.

2.4 Клієнт-серверна архітектура. Технології Webhook, Websocket та Node.js

Як було описано вище, сучасні веб-додатки можуть бути складними і комплексними програмами, які можуть взаємодіяти із широким набором бібліотек і великою кількістю ресурсів.

Більшість веб-додатків мають клієнт-серверну архітектуру [16].

Хоча питанню розробки саме клієнтної частини буде присвячена більша частина даної роботи, треба зачепити також влаштування і серверної частини, без якої розробка подібного додатка просто б не мала смислу.

Клієнт та сервер є двома незалежними процесами, які взаємодіють один з одним. На стороні сервера відбувається обробка даних, його робота здебільшого є непомітною для користувача, який напряму взаємодіє із клієнтською частиною. Графічний інтерфейс (UI), також працює на клієнтській частині. Розробка цієї частини додатку називається Front-end, а його розробник Front-end розробником.

Серверна частина, в свою чергу дозволяє робити складні розрахунки, операції над даними, а також спілкуватися з іншими сайтами, приймати від них повідомлення, та передавати їх конкретним користувачам клієнта.

Ця частина додатку називається Back-end, а його розробник, відповідно Back-end розробником.

Для реалізації подібних процесів існує технологія Node.JS [17].

Ця технологія дозволяє управляти і програмувати роботу сервера.

В нашому випадку, за допомогою цієї технології виконується багато різних функцій: за допомогою медіакодека відбувається декодування утвореного клієнтом відео та передача його по протоколу до місця визначеного призначення.

Відбувається автентифікація користувача для отримання каналів передачі, за допомогою нього відбувається прийом пожертв.

Для останньої задачі застосовується технологія WebHook [18]. Node.JS дозволяє влаштовувати канал, який має конкретну адресу, і по якій можна приймати текстові повідомлення. Таким чином можна отримувати сповіщення про пожертви.

Для спілкуванням із клієнтською частиною додатку, розробка якого буде описано в майбутніх розділах, використовується протокол WebSocket [19]. Він також дозволяє встановлювати канал для спілкуванням між частинами додатку та передавати інформацію в обох напрямках.

3 ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ СТРУКТУРИ ТА ЛОГІКИ ВЗАЄМОДІЇ

3.1 Визначення цільової авдиторії видання

Програмне забезпечення, в тому числі сайти та веб-додатки, як його складові, призначені для того, щоб задовольняти потреби користувача у роботі тих чи інших функцій.

Функціональність продукту визначають потребу в продукті, що розробляється. Тому функціонал програми має бути чітко сформульованим: якщо програма буде виконувати занадто обмежений функціонал, то тоді користувач не відчуватиме в ній потреби, якщо ж функціонал буде надто широким, користувачу буде складно в ньому розібратися і він не розумітиме його суть [3]. Це ускладнюватиме інтерфейс і негативно впливатиме на час, необхідний користувачу задля досягнення мети.

Тож даний етап є дуже важливим і йому потрібно приділити увагу. Для виконання цієї задачі в першу чергу потрібно проаналізувати цільову авдиторію майбутнього продукту.

В UX для визначення цільової авдиторії існує так званий підхід user persona [20]: це портрет одного чи декількох усереднених потенційних споживачів, що має вік, стать, локацію, досвід, власні уподобання. Побудова успішного видання ґрунтується на задоволенні інтересів конкретних споживачів, і дане дослідження допоможе ці інтереси визначити. Розроблені персони визначенні в табл. 3.1. Визначено вік, освіту, місце проживання, мотивацію, освіту мету, болі, та потреби. Вивчення відбувалося за допомогою особистого опитування осіб, що займаються відео-блогінгом.

User Persona надає можливість більш детально вивчити цільову авдиторію, вияснити із якими проблемами вони стикаються.

Таблиця 3.1 – User persona потенційних користувачів

Ім’я	Антон	Олена	Максим
Вік	22	28	18
Стать	Чоловік	Жінка	Чоловік
Освіта	Бакалавр за спеціальністю історія	Магістр за спеціальністю графічний дизайн	Повна середня освіта
Місце проживання	Київ	Дніпро	Львів
Інтереси	Політика, історія	Графічний дизайн	Відеоігри
Особливість	Має канал в Telegram та Youtube на політичну тематику.	Має портфорліо на Behance	Має канал на Youtube і акаунт в Twitch.
Бажання	Проводити відео-трансляції для того, щоб обговорити актуальні суспільні проблеми разом із своєю аудиторією.	Бажає організовувати вебінари на Behance.	Грати в ігри разом із друзями, демонструвати власний прогрес на широку аудиторію
Мотивація	— збирати кошти на підтримку Збройних Сил України; — підтримувати контакт з авдиторією; — залучати підписників до підтримки власного проєкту.	— підняти популярість власного портфоліо; — залучити потенційних клієнтів.	– підтримувати контакт із авдиторією; — залучати кошти на розвиток власного проєкту.

Продовження таблиці 3.1

Ім’я	Антон	Олена	Максим
Болі	— організувати трансляцію на декілька джерел водночас складно; — іноді при залученні коштів, треба використовувати декілька рахунків (наприклад, розділити по цілям збору). Налаштувати стандартизовані сповіщення в такому разі складно.	— Під час роботи складно читати чат, аби швидко відповідати на питання глядачів.	— Для того щоб читати чат, треба тримати вікно чату відкритим, що проблематично в умовах гри. — Щоб отримати донати на різні рахунки (підписка на сервіс монетизації, прийом донатів, переказ на картку), складно налаштувати єдине сповіщення.

3.2 Аналіз аналогів

Дуже рідко буває, щоб який-небудь сайт або веб-застосунок не мав би конкуруючих аналогів.

А це значить, що потреби користувача вже частково задоволені. Аналіз аналогів дозволить не тільки визначити доцільність створення нового продукту, а ще й розглянути їх недоліки, що дозволить визначити речі, які варто врахувати, аби досягнути кращого продукту серед існуючих.

В даній галузі також існує декілька аналогічних сайтів та програм, що конкурують між собою. Розглянемо їх аби визначити недоліки.

1. OBS Studio (Open Broadcaster Software) [21] на сьогодні є найбільш популярним програмним забезпечення для організації інтернет трансляцій (рис. 3.1). З’явився як невеликий проєкт одного розробника, однак швидко розрісся через підтримку небайдужих. На сьогодні має найбільш широке

ком’юніті, яке забезпечує підтримку сервісу. Має досить широкий набір функцій, широкий набір протоколів передачі, що підтримуються. Має широкий набір платформ: підтримує Windows, Linux, macOS, FreeBSD.

Однак разом із тим має величезну кількість недоліків. Головний із них, що водночас є і його перевагою, це наявність виключно ПК-версії. Це означає, що найбільш ресурсоємний процес, а саме декодування відео, відбувається повністю на боці клієнтського ПК. Що робить програму досить вибагливою до користувача. Також за допомогою OBS складно організувати декілька трансляцій. Досить велика кількість функцій робить інтерфейс дуже перевантаженим, що призводить до складності у налаштуванні серед новачків. Немає роботи із чатом.

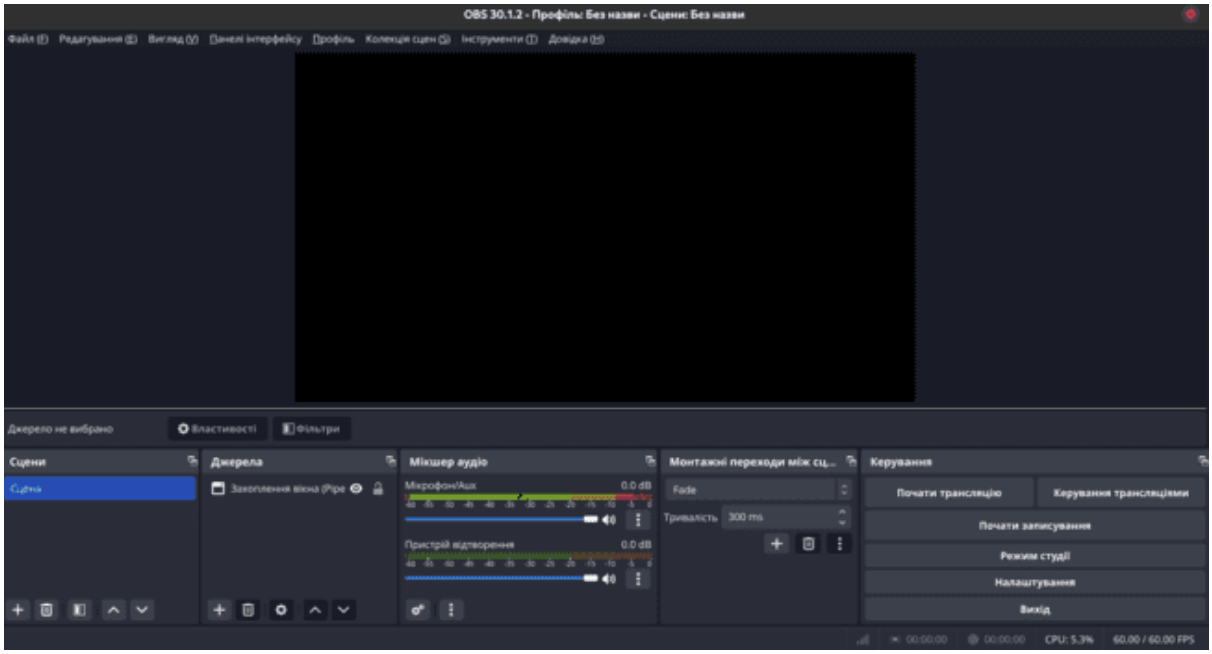

Рисунок 3.1 – OBS Studio

2. Restream [22] продукт від української фірми, що також отримав певне визнання. На відміну від попереднього пункту він є повноцінним веб-додатком (рис. 3.2). Його основною фішкою є просте налаштування трансляції на декілька платформ водночас. Він має досить простий та інтуїтивно зрозумілий інтерфейс. Присутні ілюстрації, інтерфейс виглядає

досить зрозумілим, наявна адаптивність. Складається з декількох розділів, кожен який розділений за функціоналом. Однак натомість пропонує досить обмежений функціонал в плані кастомізації вигляду вашої трансляції, та високу вартість підписки.

На головній сторінці додатка потрібна створити трансляцію, після чого ви потрапите в систему редагування трансляції. Досить зручним здається система переключень сцени, однак вона досить обмежена в плані налаштувань, так що простота компенсується функціональністю. Незрозуміло, як налаштовувати вивід звукового сигналу.

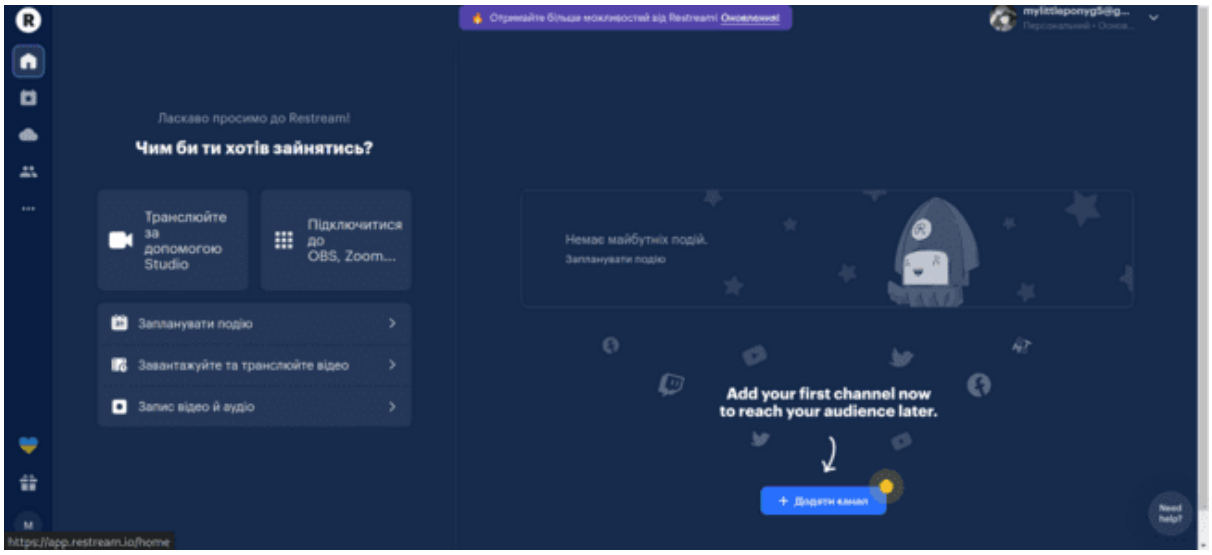

Рисунок 3.2 – ReStream

3. Streamyard [23] ще один схожий на попередній продукт. І має схожий на попередній пункт недоліки. Його інтерфейс (рис. 3.3) виконано в білій колірній гамі. На відміну від ReStream, його інтерфейс виглядає простіше, най і мінімалістічніше.

Використовується світла колірна гама, що, враховуючи необхідність працювати із відео, є скоріше недоліком. Використання світлої колірної гами коли йде робота з насиченими кольорами, чим є відео, що транслюється є не дуже оптимальним рішенням [38].

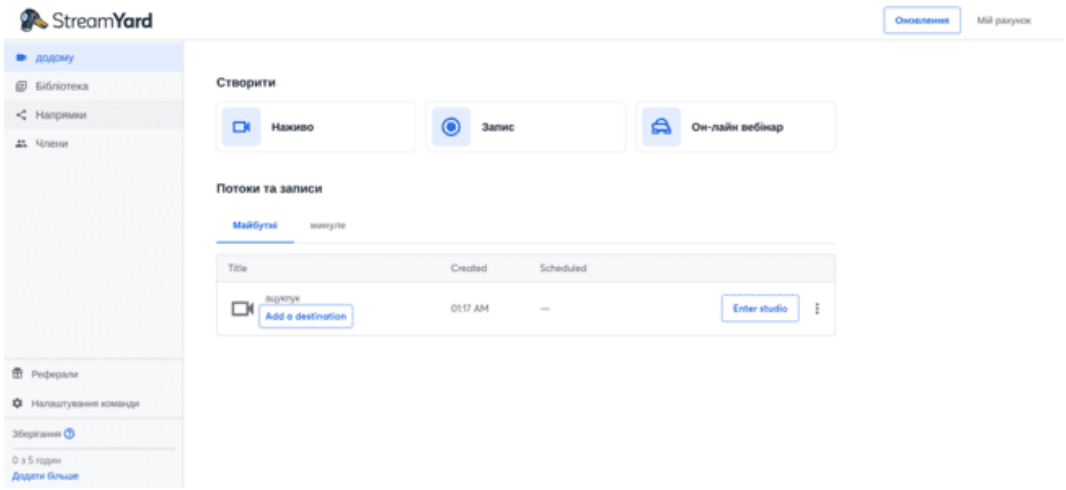

Рисунок 3.3 – StreamYard

3.3 Визначення функціональності

Після того, як досліджено цільову аудиторію та аналоги від конкурентів, потрібно визначити саме способи вирішення існуючих проблем та покращити взаємодію. Для цього необхідно побудувати CJM-таблицю (Customers Journey Map). Вона дозволяє визначити чіткіше із якими проблемами стикається користувач на яких етапах (табл. 3.2).

Таблиця 3.2 – Customers Journey Map

Крок	Планування трансляції	Налаштування трансляції	Управління трансляцією
Дія	— визначення теми та часу майбутнього відеопотіку; — визначення на якій платформі його проводити; — пошук рішень, що задовольнятиме.	— створити трансляцію на обраній платформі; — налаштувати сцени для користувача; – налаштувати вивід звуку; — налаштувати прийом пожертв.	— запуск або зупинка трансляції; — тимчасове вимкнення однієї із звукових доріжок; — робота із чатом; — перемикання між сценами.

Продовження таблиці 3.2

Крок	Планування трансляції	Налаштування трансляції	Управління трансляцією
Потреба	Вивчити доступні програмні рішення	Налаштувати майбутній контент під власні потреби	Управляти випадками, які можуть виникнути під час відеотрансляції
Болі	— складності у налаштуванні програмного рішення; — наскільки потужним мусить бути ПК для цього рішення? — ціна.	— звук буде занадто гучним/тихим; — пожертви не надсилатимуться; — я не зможу правильно розташувати зображення.	— трансляція працюватиме не стабільно; — що вдастся залучити пожертви.
Тачпоїнт	Пошук в інтернеті	Сайт веб-додатку	Сайт веб-додатку
Як покращити взаємодію?	Зручна, оптимізована для пошуку, сторінка із описом переваг продукту, та ціною.	— додати простий інтерфейс управління звуком; — створити налаштування сцени, перемикання між ними; — створити просте меню налаштування трансляції.	— помічати інформацію про якість з’єднання; — надати інструментарій для перегляду та управління чатом.

3.4 Визначення логіки взаємодії

Отриманих в попередніх підпунктах даного розділу даних, а саме аналізу цільової авдиторії, аналіз аналогічних рішень, та побудова CJM, достатньо аби зробити певні висновки про потреби користувача.

Користувач потребує доступ до таких функцій, як створення трансляцій із можливістю транслювати на декількох майданчиках водночас, аби залучати більше аудиторії, можливість управляти звуковими доріжками, можливість легко приймати пожертви, управляти чатом, а також розміщувати віджети, щоб отримувати інформацію про чат, отримані пожертви. Також існує потреби в тому, щоб користувач міг побачити в якому стані сигнал, що передається. І чи відбувається потік взагалі.

Інтерфейс, що пропонується OBS здається перевантаженим, однак він також має кілька переваг. Інтерфейси ReStream та StreamYard розділяють саме налаштування трансляції і меню планування трансляції. Таким чином створюється хибний причино-наслідковий зв’язок у логіці взаємодії. Користувач в першу чергу хоче, підлаштувати під власні потреби, або використовувати вже заготовлені до цього шаблони, і лишень внісши у них зміни, або використавши ще раз, створити нову трансляцію, в той час як останні два аналоги пропонують спочатку створити трансляцію, а потім її налаштувати у спеціальному вікні.

Воно також має досить обмежений функціонал в плані налаштування звуку. Незрозуміло, з яких джерел отримується звук і наскільки він наразі гучний або слабкий.

Коли трансляція почалася, OBS показує корисну інформацію про якість трансляції, а решта сервісів ні. Однак саме вона дає змогу переконатися, що із потіком усе добре.

Також вищеописаний розділ між створення трансляції та студією, в якій вона створюється розвиває цілісність додатку.

Попри весь бракуючий функціонал суть майбутнього продукту має бути все ще ясною [36]: це веб-додаток для організації інтернет трансляції, тож на першому місці мають бути функції, що пов’язані із досягнення цієї мети, в той час як до інших має бути зрозумілим доступ, однак цей аспект має займати провідну роль.

На основі проаналізованої інформації була визначена логіка взаємодії, для її опису була побудована UseCase-діаграма, як показано на рис. 3.4

Для вирішення вищеописаних задач, пропонується створити додаток, що складатиметься із двох сторінок.

Перша буде головною сторінкою додатку: вона буде описувати переваги веб-програми, аби користувач відразу зрозумів, куди він потрапив. І також можливість увійти та перейти до другої сторінки.

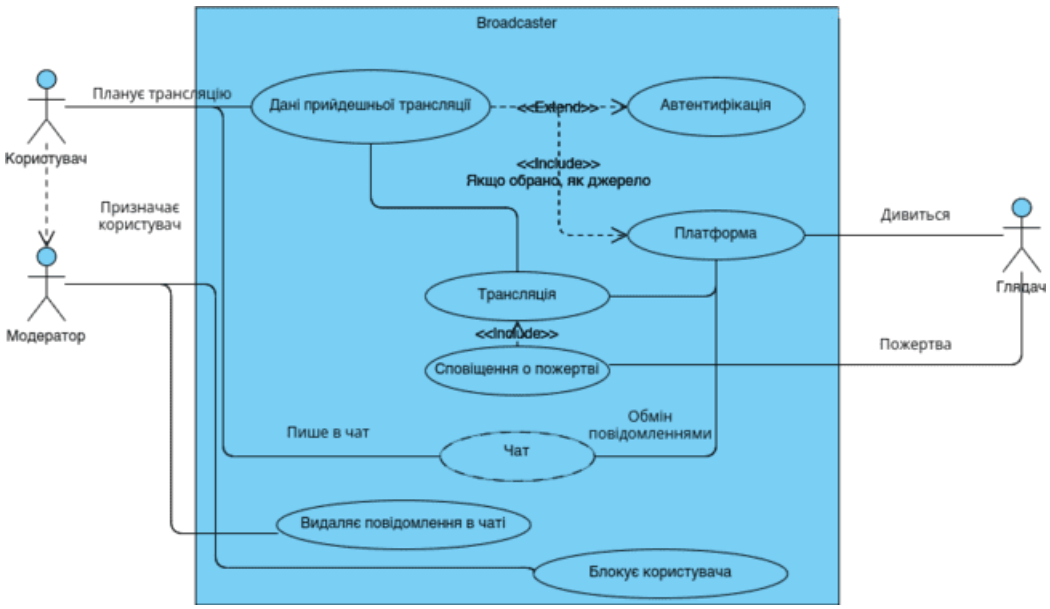

Рисунок 3.4 – UseCase діаграма

Друга сторінка буде поділена на дві частини: зліва буде зображено передперегляд того, що бачитиме користувач і декілька кнопок, а справа чат майбутньої відеотрансляції, в який можна буде писати, видаляти повідомлення, призначати модераторів, та блокувати користувачів.

У верхній частині буде розміщено передперегляд трансляції, а знизу буде доступна панель, для налаштування авдіовиводу.

Після додавання джерела звуку, буде можливість його видалення, приглушення, та зміни гучності.

Також тут буде розміщено 4 функціональні кнопки: “Показати сцену”, “Створити трансляції”, “Пожертви” та “Почати трансляцію”.

При натисненні на ці кнопки, з’являтимуться спливаючі вікна, в яких можна буде виконати потрібну функцію. Таким чином, ми зможемо забезпечити розділення за функціями, при тому не порушуючи цілісність інтерфейсу, та не перенавантажуючи його.

При натисканні «Показати сцену» буде можливість налаштувати вивід трансляції, натискаючи по даному вікну, можна буде обрати серед віджетів потрібний, або змінити вже існуючий. Для переключення між сценами, на

верхній панелі існуватиме спеціальний перемикач. Все вищеописано зформовано до карти сайту (рис 3.5).

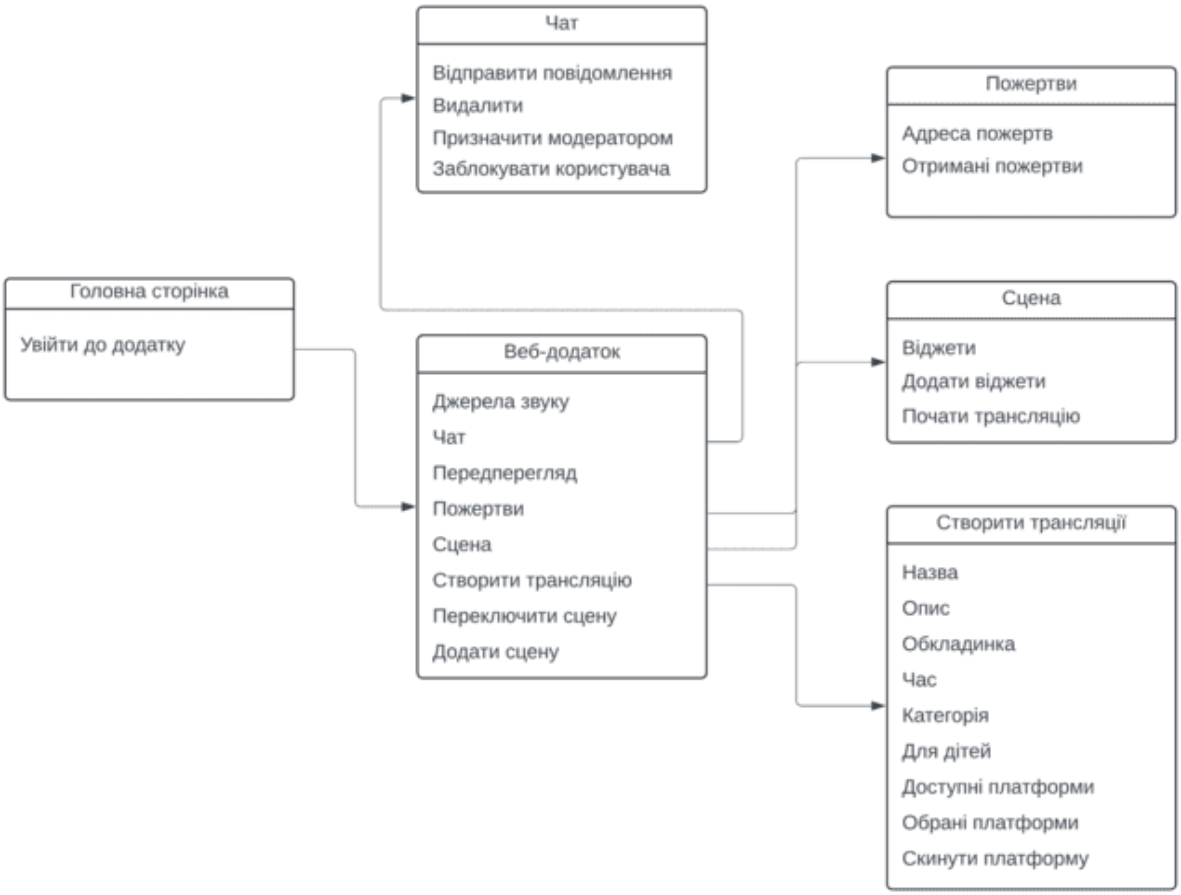

Рисунок 3.5 — Карта сайту

4 ОБҐРУНТУВАННЯ ВИБОРУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

При розробці веб-сайту застосовується інструментарій із декількох різних царин. Для розробки графічного дизайну необхідними є растрові та векторні процесори, для підготовки ілюстрацій.

Також необхідним є програмне забезпечення, завдяки якому можлива розробка дизайн-макету. Для розробки і програмування веб-сайту потрібна також спеціалізована середа розробки, що буде спрощувати верстання і робити програмування.

Растрові графічні процесори, допомагають редагувати растрову графіку, тобто таку, що складається із пікселів. Такими є скріншоти, фотографії, відскановані зображення та малюнки.

Серед растрових процесорів найбільш відомими є Adobe Photoshop, GIMP та Protopea.

Усі вони пропонують досить широкий вибір інструментарія.

Найстаріший та найбільш широко відомий, назва якого стала ім’ям прозивним, є Adobe Photoshop [24]. Саме в межах цього графічного процесору, наприклад, з’явилася концепція шара.

Він пропонує величезний інструментарій, який дозволяє робити із растровою графікою практично усе що завгодно, а з недавніх пір в ньому інтегрована нейронна мережа, що часто спрощує роботу.

GIMP (GNU Image Manipulation Program) [25] є другим за популярністю растровим процесором.

Спочатку розроблявся як редактор зображень для операційних систем сімейства Linux. Він розроблявся, ще тоді коли в останніх не існувало власного набору елементів для побудови інтерфейсів, тож саме цій програмі Linux зобов'язаний величезною кількістю графічних програм.

За своїм функціоналом практично не поступається Adobe Photoshop, хоча очевидно, що GIMP знаходиться в ролі доганяючого. Також його

інтерфейс та клавіатурні скорочення можуть здатися незвичним для користувачів попередньої програми.

Protopea [26] з’явився досить нещодавно і є спробою створити альтернативну версію Adobe Photoshop.

Його головним плюсом є можливість працювати онлайн, без прив’язки до конкретної платформи.

Хоча по функціональності він все ще поступається Photoshop.

Іконки, логотипи, графічні ілюстрації являють собою векторну графіку. Їх основною перевагою перед растровою є можливість масштабування без втрати якості, так як, на відміну від растрової складаються не з обмеженої кількості пікселів, які при збільшенні роблять зображення розмитим, а при зменшенні занадто чітким, а з геометричних примітивів.

Хоча це при тому значно обмежує можливості цього формату. З векторної графіки можна створити растрову, а з растрової векторну – ні.

Серед найбільш поширених програм для редагування векторної графіки існують Adobe Illustrator, InkScape та CorelDRAW.

Так само, як і Adobe Photoshop, Adobe Illustrator [27] є промисловим стандартом. Він має дуже широкий функціонал, більшість векторної графіки орієнтовано саме на Adobe Illustrator.

CorelDRAW [28] також є поширеним векторним редактором, хоча значно менше ніж Adobe Illustrator. Його інтерфейс видастся незручним для тих, хто звик працювати у Adobe Illustrator.

CorelDraw має менший функціонал.

Так само як і Adobe Illustrator є платним.

InkScape [29] багато в чому схожий на GIMP. Так само вільно розповсюджується, і так само спочатку розвивався як програма для Linux. Його інтерфейс є досить застарілим, помітно, що програма розвивається повільніше ніж конкуренти, що не дивно враховуючи, що на відміну від попередніх, InkScape підтримується добровольцями.

Для проєктуваня макетів також існує досить широкий вибір додатків: Figma, PenPot, Adobe XD.

Figma [30] є промисловим стандартом, що має досить широкий набір функцій. Вона дозволяє працювати над проєктом декільком особам, переглядати код CSS, що позбавляє від необхідності спеціального софту по типу AvoCode чи Grippin для спрощення верстання. Розповсюджується на умовно платній основі.

PenPot [31] є реалізацією безкоштовної та вільної альтернативи, що намагається повторити Figma.

Хоча він має певні проблеми із стабільністю, він вже підтримує досить багато функцій, а також надає багато дизайн-мов, як основу для майбутнього інтерфейсу, так само, як і Figma.

На відміну від, попередніх Adobe XD [32] є самостійним додатком, а не веб-сервісом. Він має досить зручну систему мокапів та сітки, що повторюються, розумну анімацію, що добре пришвидшує розробку.

На жаль, після покупки Figma, Adobe значно більше приділяє уваги підтримці саме їй, і ймовірно намірена відмовитися від нього в майбутньому.

Також важливим є вибір середи розробки для верстання та програмування:

WebStorm [33] – продукт, що розробляється чеською корпорацією JetBrains. Підтримує підкреслення синтаксису в CSS, HTML, JavaScript. Пропонує автоматичне підвантаження фреймворку Bootstrap, а також CMS Wordpress та Joomla.

Має інтеграцією із системою контролю версій git. Є кросплатформним. Підтримує Window, mac Os, Linux, хоча в останньому має деякі проблеми із сумісністю. Є платним і комерційним продуктом.

Visual Studio Code [34] – кросплатформене IDE від корпорації Microsoft. Має підтримку величезної кількості різних синтаксів, які можна розширювати за допомогою плагінів. Має інтеграцію із системою контролю версій git. Добре працює на всіх сучасних платформах.

Для обробки векторної графіки було обрано Adobe Illustrator, для роботи із растровою графікою, буде використовуватися Adobe Photoshop.

Ці програми досить зручні, зарекомендували себе на ринку, та можна вважати є промисловими стандартами.

Для створення прототипу використовується програма Penpot. Вона є повністю безкоштовною.

Для верстання веб-сайту використовується середа програмування Visual Studio Code через її безкоштовність та широкий вибір функціонала.

5 РОЗРОБКА ДИЗАЙН-МАКЕТУ МАЙБУТНЬОГО ВЕБ-САЙТУ

5.1 Розробка модульної сітки

Модульна сітка [8] є конче важливим компонентом сучасного графічного дизайну, в тому числі під час проєктування інтерфейсу. Використання модульної сітки дає декілька важливих переваг. По-перше вона дозволяє робити дизайн більш збалансованим: інтерфейс розділяється на рівні частини, що дає можливість забезпечити єдину систему відступів та полів, так звану консистентність.

Модульна сітка може бути зручним способом забезпечення адаптивності – можна змінювати порядок відображення в блоках сітки, скільки колонок займає той чи інший елемент, контент в деяких колонках можна приховувати при різних умовах відображення.

Веб-додаток, що розробляється заточується під комп’ютери. Операційні системи IOS та Android просто не підтримують можливість захоплення екрану, що необхідно для коректної роботи, в силу правил безпеки, та деякі інші функції. Однак в майбутньому є можливість адаптувати проєкт і під мобільні пристрої.

Тим не менш, це не позбавляє необхідності створити адаптивний дизайн. Користувач може змінити розмір вікна на комп’ютері, розміри екранів також є різними.

Як зазначалося вище, в продукті присутні дві сторінки.

Одна з них є веб-додатком, інша буде головною сторінкою, що видаватиметься при пошуку.

Обидві будуть використовувати 12 колонкову систему (рис. 5.1), однак друга також буде поділятися і на горизонтальні блоки (рис. 5.2). Сторінка буде складатися із декількох блоків, що прокручуватимуться згори вниз, і розбиття інформації на блоки допоможе розділити її і зробити дизайн збалансованим. Відстань між колонками становить 40 пікселів, цієї відстані

буде достатньо, аби забезпечити візуальне розділення інформаційних блоків. Також її колонки є більш вузькими, аби забезпечити більші бічні відступи.

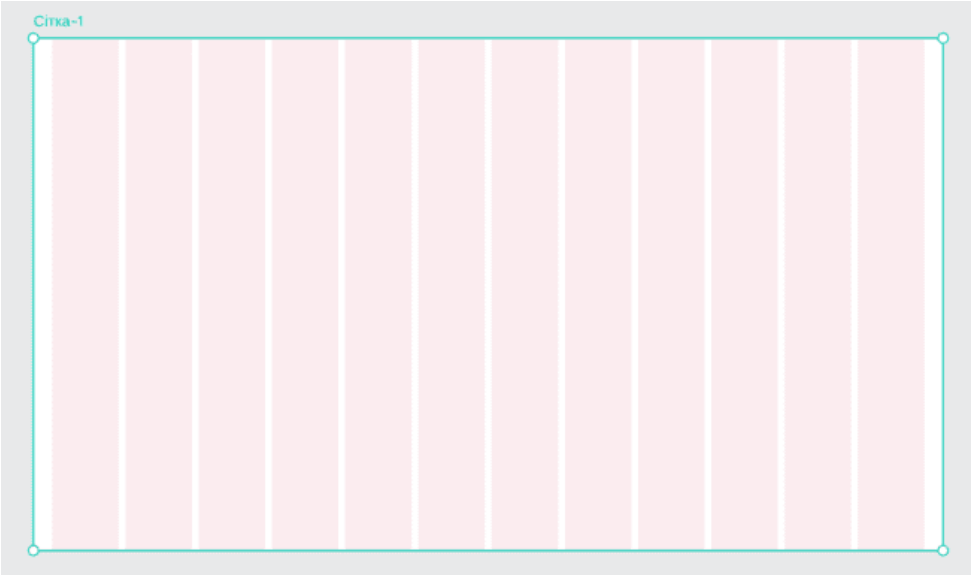

Рисунок 5.1 – 12-колонкова модульна сітка основного веб-додатку

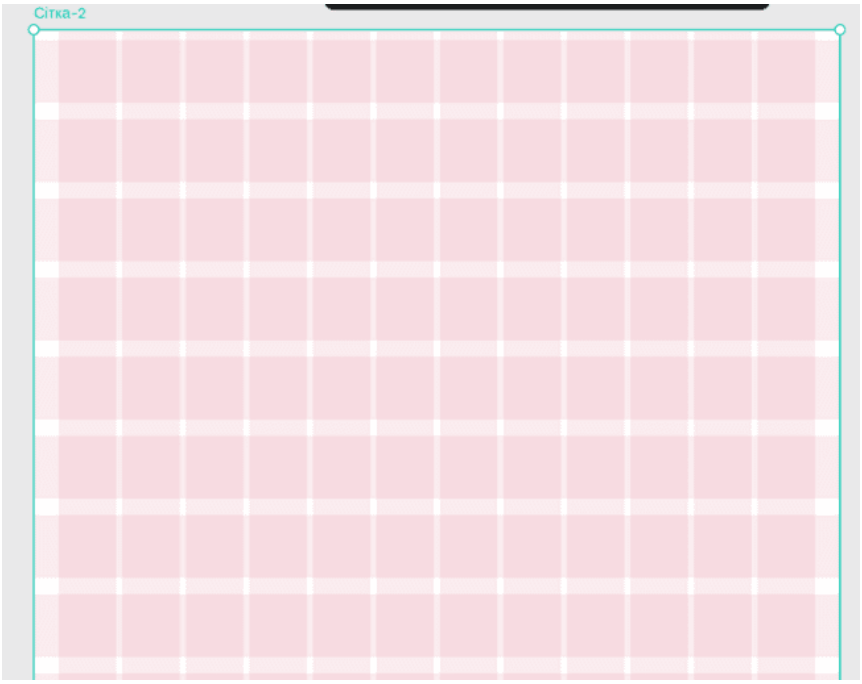

Рисунок 5.2 – Модульна сітка головної сторінки

5.2 Обґрунтування дизайнерського рішення

Після визначення навігаційної системи та логіки взаємодії майбутнього веб-сайту, створення модульної сітки, що використовуватиметься, важливо визначити колірне та шрифтове рішення, вигляд елементів управління.

Саме на цьому етапі потрібно визначити естетичні засади продукту, що створюється. Навіть якщо інтерфейс має бути інтуїтивним, але естетично не привабливим, користувач все ще ним не користуватиметься.

Також, саме дизайнерське рішення визначатиме наскільки користувачу буде легко досягати мети і його досвід буде приємніше.

Для спрощення розробки та вигляду, за основу дизайну була використана Gnome Human Interface Guideline [14], як готове рішення.

Він пропонує декілька принципів та готових рішень, набір піктограм, кольорів, які можуть бути використані для отримання фінального результату, а саме:

– підкреслення елементів управління збільшенням яскравості та тінями. Кнопка виділяється використанням більш світлого кольору, світлість збільшується при наведенні, для того щоб підтвердити вибір;

– усі елементи управління мусять містити візуальні (в формі піктограм) та текстові підказки, аби користувачу було зрозуміло, для чого потрібен той чи інший елемент;

– окремі елементи вводу додатково виділяються для того, щоб користувач отримав візуальне підтвердження, де він вводить дані на даний момент. Найбільш важливі елементи керування, які потребують додаткового виділення підкреслюються акцентним кольором;

– спливаючі вікна виділяються обведенням, аби візуально розділити інформацію;

– інтерфейс має використовувати закруглені кути, це знижує когнітивне навантаження, і пом’якує сам дизайн, дозволяючи сконцентруватися на змісті [43].

На основі цих принципів потрібно сформувати колірне рішення. Теорія кольору — окрема дисципліна, яка має багато нюансів. Успішне колірне рішення може забезпечити покращення користувацького досвіду, і навпаки.

Наш зір не є ідеальним, а наш глядацький досвід може відрізнятися. Ми також маємо враховувати, що деякі люди мають різні особливості, такі як різні форми дальтонізму або дислексія.

Їх особливості також варто врахувати.

Колір має підкреслювати, і правильно розставляти акценти [37]. Це досягається за допомогою контрастів, однак занадто яскравовиражений контраст може мати негативний вплив, через когнітивне перенавантаженя, і невиправдано виділятися.

За основу колірного рішення обрана темна тема. Це є виправданим, так як під час організації трансляції людина фокусує увагу на яскравих кольорах [38], що надаються передпереглядом, та іншими яскравими елементами, такими як віджети, людина звертає увагу на аватари користувачів, що пишуть в чат, і використання білого кольору може перенавантажити і так повний кольорів інтерфейс.

Акцентним кольором обрано червоний та блакитний. Першим буде виділена кнопка почати трансляцію, аби користувач відразу розумів, як це зробити, і «чипляючись» міг усвідомити де він зараз знаходиться.

За основу рішення взято темно сірий колір та його відтінки. Фінальне колірне рішення засноване на особливості мови інтерфейсу [14] та особливостях роботи із темною темою [39] наведено на рис. 5.3

Рисунок 5.3 – Колірне рішення

Важливо також визначити шрифтове оформлення. Згідно з обраною мовою дизайну обрану рекомендовану гарнітуру Cantarell (рис 5.4).

Це гуманістична гротеск гарнітура була розроблена Дейвом Крослендом у 2009 році, він є частиною сімейства Droid, з самого початку розроблявся як екранна гарнітура, що означає, що вона буде добре зчитуватися з екрану. Додатковою перевагою даного шрифту, є те що він також вільний навіть для комерційного використання.

Шрифт Cantarell [40] добре відображається на екрані, він є досить простим, однак водночас елегентним, сучасним, та чистим.

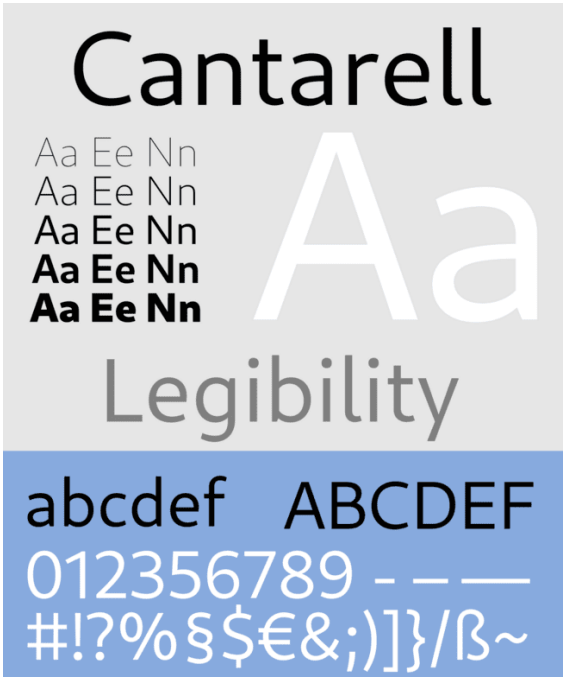

Рисунок 5.4 – Cantarell

Буде використано декілька накреслень. Для заголовків використано чорне накреслення, для підказок і кнопок використано жирне, для решти елементів управління використано звичайне накреслення.

Згідно із визначеними рекомендаціями розроблена і іконка нового сайту. Вона являтиме собою супутникову тарілку, що випромінює червоні хвилі. Вона виконанна у стилі скевоморфизму (рис. 5.5).

Рисунок 5.5 – Іконка

На основі іконки розроблена ілюстрація для головної сторінки (рис. 5.6).

Рис 5.6 – Ілюстрація

5.3 Розробка дизайн-макету

Після визначення попередніх пунктів цього розділу, на їх основі можна переходити до розробки дизайн-макету майбутнього веб-додатку.

Необхідно скористатися вищеобранним інструментом PenPot [32] і перейти на його сайт penpot.app (рис. 5.7).

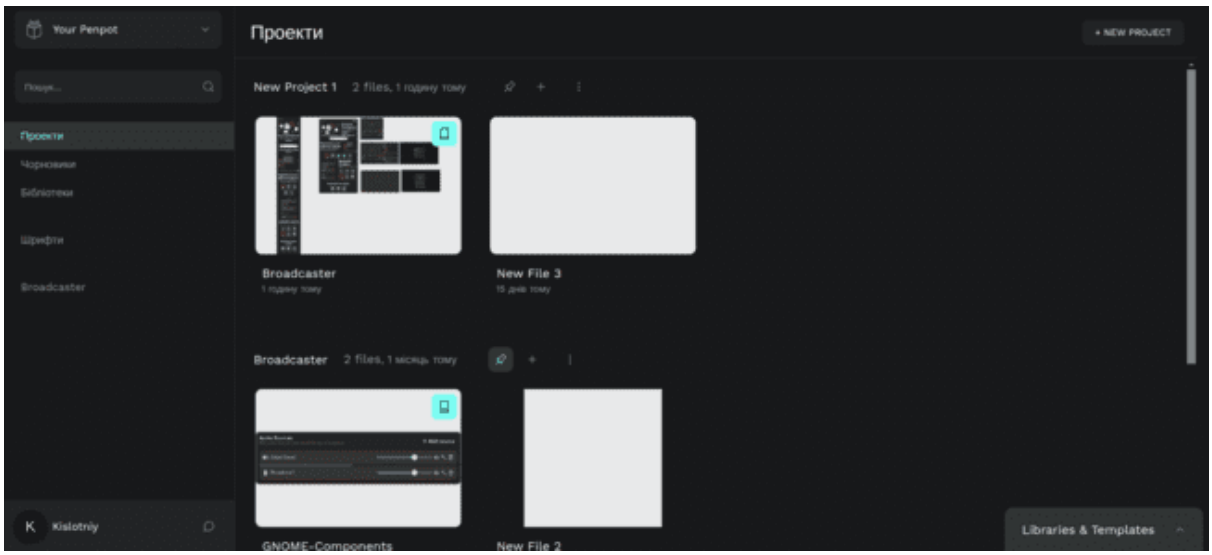

Рисунок 5.7 – Головне меню PenPot

Для розробки було обрано бібліотеку GNOME-Components. Це бібліотека компонентів: кнопок, полів введення , та інших елементів керування, що відповідає рекомендаціям обраної мови дизайну.

За допомогою доступного інструментарією, за допомоги модульної сітки було створено дизайн-макет, згідно із сформованою картою сайту, що відповідає дизайнерському рішенню, що описане в попередніх розділах.

На цьому етапі важливо також продумати аспект адаптивності: як виглядатиме веб-додаток при різних розмірах екрану.

Для цього скористаємося принципом Desktop First. Для початку було створено дизайн-макет повноцінної версії додатку, а потім скопіювавши варфрейм, змінено його розмір, і адаптовано сторінку, що наповнена контентом під нові умови.

Результатом буде декілька макетів, що відповідають до цього поставленим умовам (рис. 5.8). Тепер їх можна використовувати для прототипування і привести до стану, що уможливлює розробку.

Рисунок 5.8 – Дизайн макет

Результати даного етапу розташовані у додатку А.

5.4 Прототипування дизайн-макету

Результатом даного етапу буде прототип веб-сайту, що необхідно розробити. Він вказуватиме, як елементи дизайн-макету мають взаємодіяти між собою.

Це дозволяє перевірити функціональність та протестувати, наскільки добре інтерфейс впорується із своїми функціями.

Для проведення прототипування у багатьох систем для побудови дизайн-макетів, існує спеціальний режим прототипування. PenPot тут не виключення. Необхідно перейти в цей режим, після чого між усіма елементами керування можна встановлювати зв’язки, таким чином показуючи, як елементи керування взаємодіятимуть між собою (рис 5.9).

Прототип має відповідати карті сайту, що була збудована в третьому розділі даної роботи. Тепер вона реалізована в готовий прототип, що взаємодіє між собою.

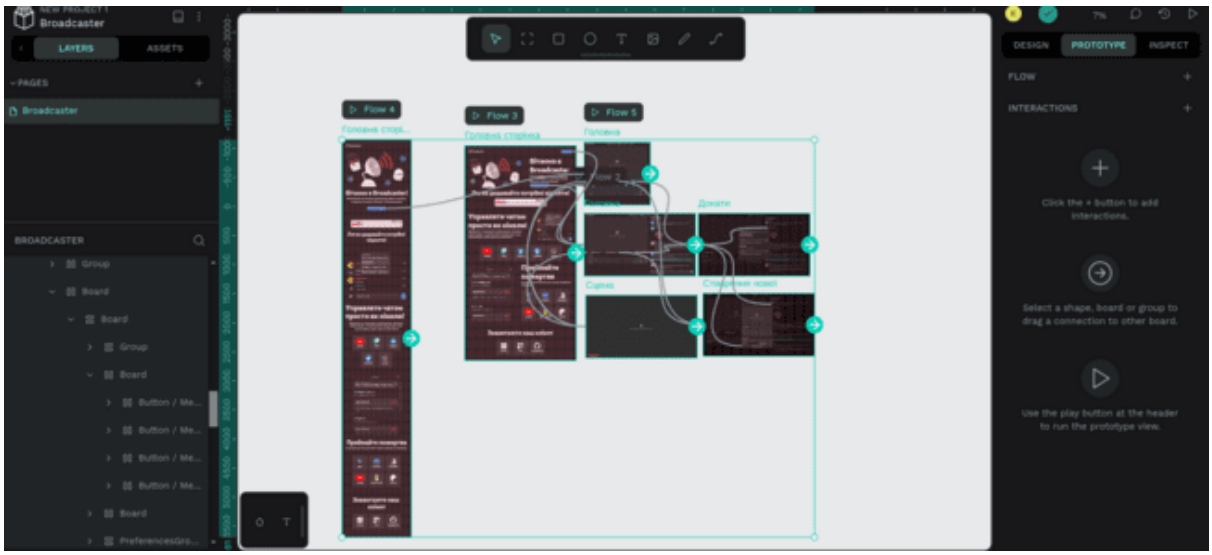

Рисунок 5.9 – Прототип

6 РОЗРОБКА ВЕБ-САЙТУ НА ОСНОВІ ДИЗАЙН-МАКЕТУ

6.1 Наповнення контентом сторінок видання

Після створення дизайн макету необхідно перейти до розробки на основі оригінал макету.

Розробка відбувається в середі розробки Visual Studio Code.

Як зазначалося вище, веб-додаток складається із серверною та клієнтської частини. Серверна частина написана на Node.JS та спілкується із клієнтом за допомогою протоколу WebSocket.

В рамках цієї роботи, буде зосереджена увага на розробці клієнтської сторони програми.

Для розробки веб-сторінки був застосований фреймворк Material CSS [41]. Він дозволяє спростити процес верстання та розробки.

Він вже надає широкий вибір готових рішень, такі як модульна сітка, набір скриптів, стилів, елементів керування.

Він має широку і зрозумілу документацію, є простим у вивченні.

Попри використання даного фреймворка обрана мова дизайну відрізняється, тож існує потреба у створенні додаткових стилів, які будуть змінювати надані.

Для розробки веб-сайту використовується HTML, CSS та JavaScript. Фрагменти коду будуть надані у Додатку Б.

В першу чергу розробка почалася із екрану вітання.

Тут описані переваги веб-додатку, кнопка для авторизації, та пропозиція скачати веб-додаток.

Вгорі розташовано шапка сторінки (рис. 6.1).

Тут зображено логотип (рис 5.5) та назву веб-додатку. В правому куті, знаходиться кнопка увійти у додаток.

Рисунок 6.1 – Шапка сторінки

В наступному блоці розташована ілюстрація (рис 5.6), заголовок з привітанням, та підзаголовок.

Ще раз розташовано кнопку для автентифікації (рис 6.2).

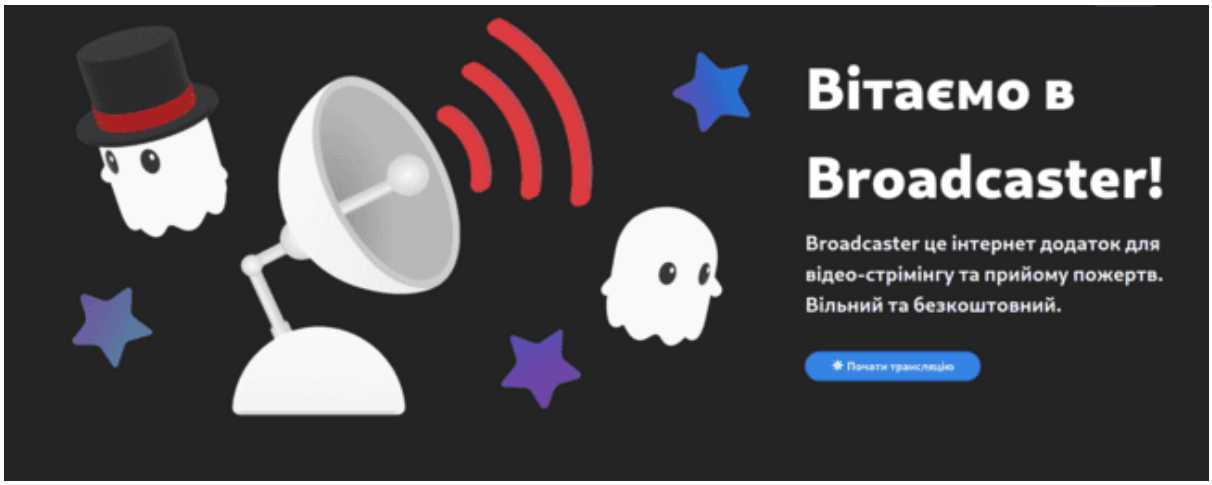

Рисунок 6.2 – Перший інформаційний блок

В наступних інформаційних блоках перераховані переваги веб-додатку (рис. 6.3 – рис. 6.6).

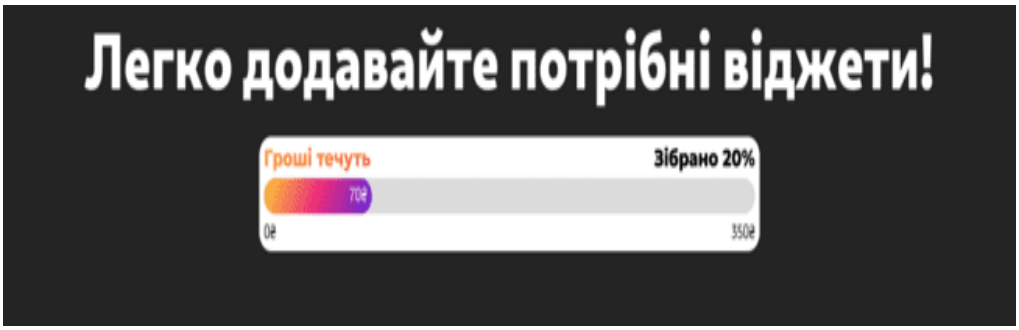

Рисунок 6.3 – Інформаційний блок

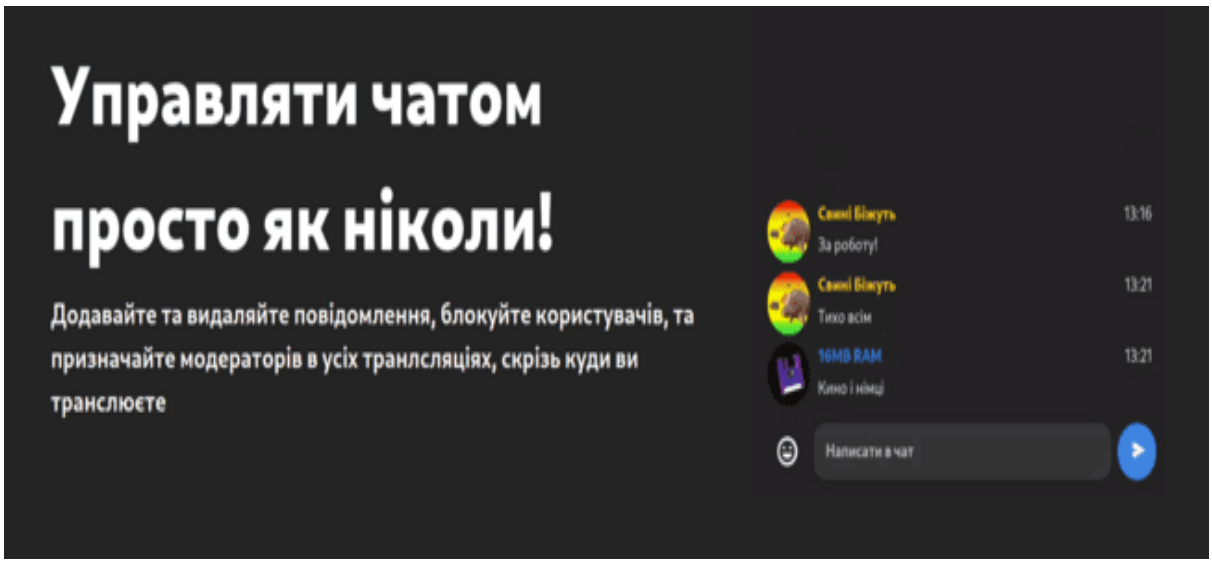

Рисунок 6.4 – Інформаційний блок

Нижче продемонстровані джерела трансляція на які підтримується. Кожен елемент виділено квадратом навкруг, що підсвічується при наведенні.

Наступні пункти виконані в аналогічний спосіб.

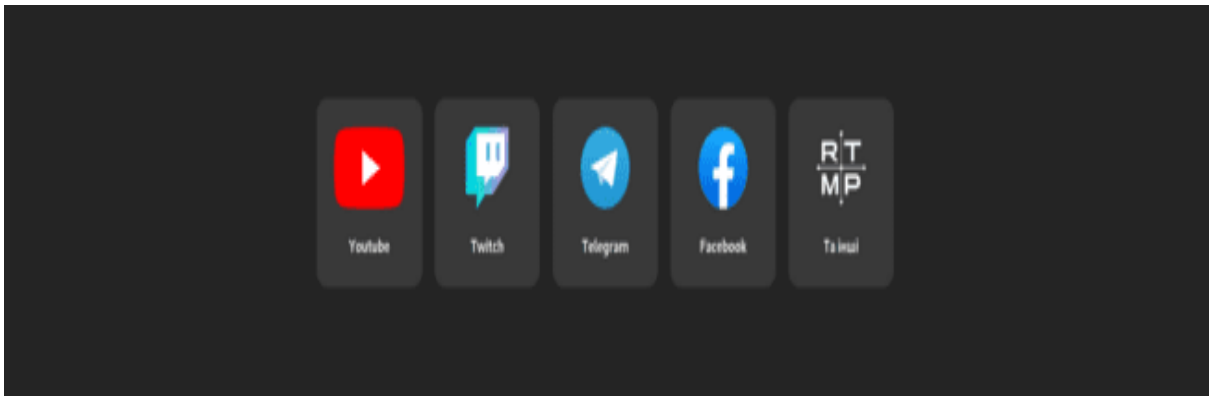

Рисунок 6.5 – Інформаційний блок

Після завершення сторінки Головної сторінки, перейдемо до сторінки веб-додатку. Доступ до неї відбувається після автентифікації одним із засобів, тобто за допомогою кнопки “Почати”, що створена вище. Згідно з вище описаним дизайнерським рішенням побудуємо інтерфейс.

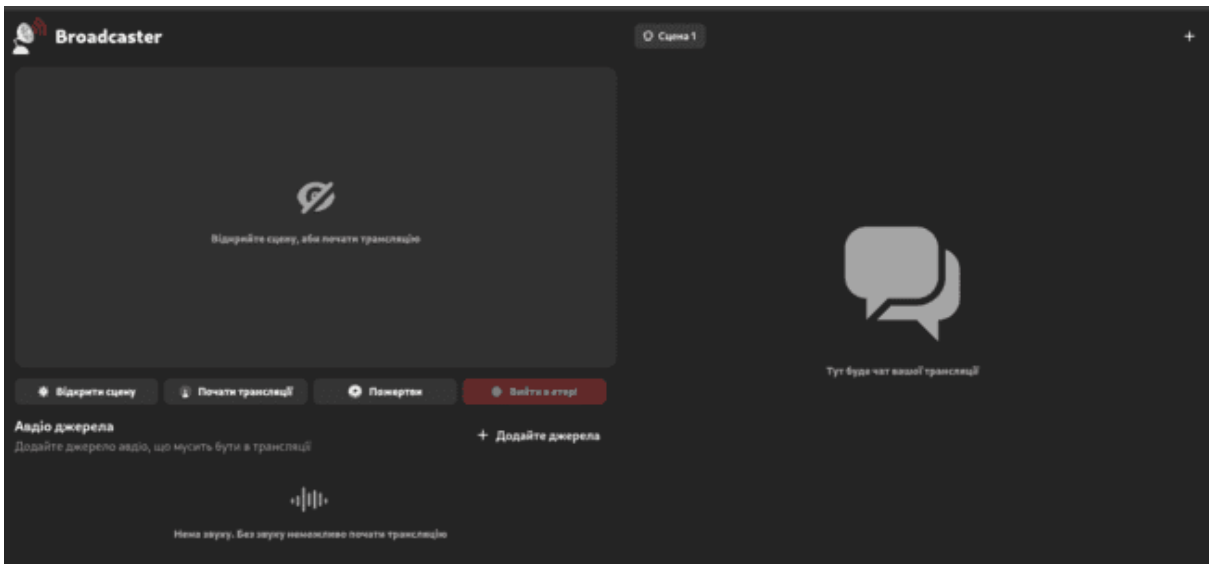

Рисунок 6.6 – Сторінка веб-додатку

6.2 Тестування і публікація

Після завершення розробки, треба задуматися про публікацію веб-сайту. Зазвичай для цього використовується так звані хостінги. Вони допомагають розгорнути сайт в мережі.

Існує широкий вибір серверів для розгортки веб-сайтів. Так як розробляється Web-додаток, де серверна частина збудована на Node.Js, то вибір обмежується тима хостингами, що підтримують відповідний сервер. Вибір було зупинено на хостингу Render. Наразі він єдиний на ринку, що відповідає поставленим вимогам та має безкоштовний тарифний план.

Після того, як сайт було успішно завантажено в мережу, необхідно перевірити його працездатність. Треба провести тест на кросбраузерність та протестувати адаптивність.

Для цього, за допомогою веб-браузера було перевірено сайт при різних масштабах, та розмірах вікна.

Для перевірки на кросбраузерність необхідно перевірити наскільки коректно веб-сайт відображається на різних браузерних рушіях. Наразі поширені браузери двох типів: ті що засновані на Mozilla та ті, що засновані на базі Chromium.

Для перевірки сайту на браузерах типу Mozilla необхідно протестувати його в Mozilla Firefox, для перевірки в браузерах типу Chromium, необхідно протестувати його в Google Chrome. Тестування проводилося на комп’ютерах під керуванням Fedora Linux 40 та Windows 11 в Google Chrome версії 125.0.64 та Mozilla Firefox 125.0.3.

Обидва типи браузерів добре сумісні із сучасними стандартами, такими як HTML, CSS та JavaScript. Так Google Chrome [42] на Linux набрав 576 балів з 594, а Firefox 544 з 594. Google Chrome для Windows набрав 584 бали, а Firefox 564. Однак кожен з них мають властивості, що не відповідають стандартам, однак часто до них звертаються при розробці.

Як правило вони помічаються спеціальними префіксами, webkit в випадку Chromium, і Moz в випадку Mozilla.

Також варто враховувати різницю в підтримцю різних функцій JavaScript. Наприклад, у веб-додатку використовується захоплення екрану, однак методи для його виклику відрізняється, що також важливо врахувати.

Додатково проведено тестування на Android. Веб-додаток коректно відображається, хоча очікувано не виконує недоступні під цією ОС функції.

В результаті тестування працездатність успішно доведена. Під час тестування були виявлені деякі помилки, однак вони були виправленні.

7 ЕКОНОМІЧНА ЧАСТИНА

У результаті виконання кваліфікаційної роботи був створений веб-додаток для проведення відеотрансляцій та прийому пожертв. Розробка веб-сервісів дає можливість користувачам виконувати задачі, які раніше брали на себе традиційні застосунки. Це дає можливість бути мобільним, а також не прив’язує користувача до конкретної платформи.

Багато в чому подібні додатки є більш зручними, тож в майбутньому їх ставатиме все більше.

Потреба у створенні подібного додатку існує, так як на даний момент не існує єдиного сервісу, що пропонує разом усі вищеописані функції в одному місці. Поява подібного рішення допомогла б спростити організацію інтернет-трансляцій. Якщо їх досвід буде приємним, вони будуть досить часто звертатися до цього продукту. Розробка подібного веб-додатку із зручним UX-інтерфейсом є метою цього проєкту.

Але за таких умов все одно треба переконатися в тому, що розробка дійсно необхідна і має економічну доцільність. Для цього проводиться оцінка економічної ефективності майбутнього проєкту. Вона проводиться перед проєктуванням і дозволяє визначити чи має розробка сенс. Необхідно розрахувати собівартість та ціну майбутнього проєкту.

В інтернеті вже існують сайти, метою яких є організація веб-трансляцій, але їх функціонал не повністю відповідає потребам користувача, їх UX мають недоліки. Аналіз переваг, чи то функціональних, чи то в плані організації користувацького досвіду, надає можливість визначити конкурентні переваги майбутнього продукту. Задачею є створення веб-додатку, що задовольняє їх потреби, зробленого таким чином, аби забезпечити кращий користувацький досвід.

Перевагами сервісу, що розробляється є:

1) функціональність – можливість приймати сповіщення про пожертви, наявність гнучкого налаштування трансляції;

2) користувацький досвід – простий, легкий, інтуїтивно зрозумілий інтерфейс;

3) дизайн – колірне та шрифтове оформлення є компонентами приємного користувацького досвіду, що дозволяє звертатися до інструмента ще раз.

Після визначення конкурентних переваг майбутнього продукту необхідно проаналізувати етапи, необхідні для розробки проєкту. Розробка веб-сайту складається із таких елементів як:

– аналіз поставленого завдання, виконується вивчення питань, що необхідно дослідити перед початком розробки;

– аналіз цільової авдиторії, її потреб, проведення UX-досліджень;

– визначення функціонала і задач, ієрархію їх важливості;

– визначення логіки взаємодії, побудова каркасу, що описує взаємодію, на який можна накласти майбутній дизайн;

– розробка програмного рішення для виконання поставлених функцій back-end розробником;

– визначення колірного та шрифтового оформлення, розробка іконок та необхідних ілюстрацій;

– розробка дизайну на основі попереднього пункту;

– створення прототипу, що підійде для програмування веб-інтерфейсу;

– прототип передається Frond-End інженеру для тестування та отримання відгуку;

– верстання веб-інтерфейсу за допомогою HTML, CSS та JavaScript;

– прив’язка елементів керування до реалізованих функцій, аби вони виконували своє призначення;

– тестування веб-додатку на працездатність;

– тестування на сумісність із актуальними браузерами та відповідними операційними системами;

– публікація в інтернет.

Після визначення етапів розробки майбутнього проєкту необхідно перейти до розрахунку собівартості програмного рішення.

У собівартість розробки даного проєкту входять такі елементи як:

1) основна заробітна плата;
2) додаткова заробітна плата;
3) єдиний соціальний внесок;
4) плата за електроенергію.

Розробку проєкту проводять чотири фахівці: дизайнер, front-end інженер, back-end розробник, front-end розробник. Веб-додаток розробляється протягом двох з половиною місяців. Після аналізу середньої заробітної плати на ринку праці було встановлено наступні годинні ставки за умови тривалості робочого дня 8 годин для кожного виконавця:

– front-end інженер (middle) – 187,00 грн/год;

– front-end розробник (junior) – 156,00 грн/год;

– back-end розробник (middle): 281,00 грн/год;

– дизайнер (junior): 140,00 грн/год.

На основі цих даних проведено розрахунок витрат на заробітну плату. Результат наведено в табл. 7.1.

Таблиця 7.1 – Розрахунок витрат на заробітну плату

Етап	Вид робіт	Виконавець		Годинна ставка, грн	Тривалість виконання, дні	Заробітна плата, грн
		кіль-кість, осіб	посада			
1	2	3	4	5	6	7
Аналіз завдання	Формування вимог до продукту	1	дизайнер	140,00	1	1120,00
Аналіз цільової авдиторії, UX-досліджень	Аналіз потреб користувача	1	дизайнер	140,00	3	3360,00
Проєктування інтерфейсу	Визначення логіки взаємодії між користувачем та продуктом	1	дизайнер	140,00	2	2240,00
Розробка варфреймів	Розробка каркаса майбутнього інтерфейсу	1	дизайнер	140,00	3	3360,00

Продовження таблиці 7.1

1	2	3	4	5	6	7
Розробка програмного рішення	Програмна реалізація визначенних функцій	1	back-end розробник	281,00	25	56200,00
Розробка дизайнерського рішення	Визначення графічного оформлення	1	дизайнер	140,00	7	7840,00
Створення дизайн-макету	Підготовка рішення готового для розробки	1	дизайнер	140,00	2	2240,00
Оцінка Front-End інженером	Огляд роботи	1	front-end інженер	187,00	1	1496,00
Верстання інтерфейсу	Розробка графічного інтерфейсу	1	front-end розробник	156,00	35	43680,00
Прив’язка елементів керування	Об’єднання інтерфейсу із програмним рішення	1	front-end розробник	156,00	2	2496,00
Разом					81	124032,00
Додаткова заробітна плата (10%)						12403,20
Усього						136435,20

Додаткова заробітна плата – це важливий компонент системи оплати праці, який спрямований на стимулювання працівників до продуктивної роботи та компенсування особливих умов праці. Вона включає різноманітні види виплат, що відображають додаткові зусилля, вкладені працівником, або специфічні аспекти його роботи.

У даному випадку додаткова заробітна плата становить 10 %.

$$124032{,}00 \times 10\% = 12403{,}20 \text{ грн.}$$

Ставка єдиного соціального внеску відповідно до чинного законодавства України становить 22 % та складає:

$$136435{,}00 \times 22\,\% = 30015{,}74 \text{ грн.}$$

До інших витрат слід віднести витрати на обслуговування комп’ютерів та оплату за електроенергію. Витрати на електроенергію розраховуються на основі споживаної потужності пристрою та тарифу на електроенергію. У цьому випадку передбачається використання трьох комп'ютерів з потужністю 0,65 кВт/год кожен. Вартість однієї кВт/год електроенергії становить 2,64 грн. Час використання електроенергії під час розробки становить 8 годин на день.

$$0{,}65 \times 2{,}64 \times 8 \times 81 \times 3 = 3335{,}90 \text{ грн.}$$

Також необхідно врахувати витрати на обслуговування обладнання. При розробці використовувалося три ноутбуки, середня ціна кожного – 16000,00 грн. Ноутбуки варто замінювати раз на 3 роки, вони використовуються 254 робочих дні на рік. Отже, дана стаття витрат складе:

$$((16000{,}00 / (3 \times 8 \times 254)) \times 81 \times 8) * 3 = 5102{,}36 \text{ грн.}$$

Отже, собівартість становить 174889,21 грн. Виходячи з рівня рентабельності в 30 % прибуток становить:

$$174889{,}21 \times 30\,\% = 52466{,}76 \text{ грн.}$$

Тож ціна розробки без урахування податку на додану вартість (ПДВ) становить:

$$174889{,}21 + 52466{,}76 = 227355{,}97 \text{ грн.}$$

З урахуванням ПДВ (20 %) ціна складе:

$$227355{,}97 \times 120\,\% = 272827{,}17 \text{ грн.}$$

Остаточний розрахунок наведено у табл. 7.2

Таблиця 7.2 – Розрахунок витрат на розробку та ціни додатку

Стаття витрат	Сума, грн
Основна заробітна плата	124032,00
Додаткова заробітна плата	12403,20
Єдиний соціальний внесок	30015,74
Витрати на обслуговування обладнання	5102,36
Витрати на електроенергію	3335,90
Собівартість розробки сайту	174889,21
Прибуток	52466,76
Ціна без ПДВ	227355,97
Податок на додану вартість (ПДВ)	45471,19
Ціна із урахуванням ПДВ	272827,17

Отже, розробка командою, що складається з дизайнера, front-end інженера, front-end розробника та back-end розробника, буде відбуватися 81 день та коштувати 272827,17 грн з можливістю отримання прибутку у розмірі 52466,76 грн, що свідчить про доцільність реалізації проєкту.

ВИСНОВКИ

У період розвитку мультимедійних систем веб-сайти займають все більшу і більшу роль. Сьогодні часто це не просто сторінки із певною інформацією. Сучасні технології, такі як JavaScript та HTML5 дозволяють веб-сайтам замінювати справжні додатки і мати перед ними переваги. Їх не потрібно встановлювати, вони однаково добре працюють на всіх актуальних операційних системах, за потреби їх легко адаптувати під мобільні пристрої.

Паралельно розвивається і відеоконтент, щороку набуває популярності формат відеотрансляцій, де користувач може підтримувати контакт з авдиторією, отримувати від них пожертви на підтримку власних проєктів або інші задачі.

Однак сучасні рішення не пропонують усього потрібного функціоналу, до користувацького досвіду, який вони пропонують можна поставити питання, часто він є надто громіздким або недостатньо цілісним.

У даній кваліфікаційній було розроблено веб-додаток для організації відеотрансляцій та отримання інформації про пожертви під час прямого етеру. Детально розглянуті потреби потенційних користувачів, була проаналізована цільова авдиторія продукту, аналогічні рішення, виділенні їх проблеми в плані функціоналу, та користувацького інтерфейсу.

На основі цього була розроблена логіка взаємодії із застосунком, і пізніше розроблена модульна сітка і дизайнерське рішення, що визначило колірне, графічне та шрифтове оформлення.

На основі даних параметрів був розроблений клікабельний прототип майбутнього веб-додатку.

В подальшому за допомогою фреймворка Material CSS, а також HTML, CSS та JavaScript було розроблено клієнту майбутнього веб-додатку.

Також була розрахована вартість розробки, вона буде становити 268 761 гривень.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. What is Web-Application – Web Application Explained. URL: https://aws.amazon.com/what-is/web-application/ (дата звернення: 21.05.2024).

2. Uijun Park. Introduction to Design Thinking for UX Beginners: 5 Steps to Creating a Digital Experience That Engages Users with UX Design, UI Design, and User Research. Start Building Your UX Career. Нью-Йорк: UIJUN Company, 2023. 164 p.

3. Jenifer Tidwell, Charles Brewer, Aynne Valencia. Designing Interfaces: Patterns for Effective Interaction Design 3rd Edition. Себастополь.: O`Reily, 2018. 599 p.

4. Streamers: the new wave of digital entrepreneurship? Extant corpus and research agenda / Maria Törhönen // Electronic Commerce Research and Applications. 2021. № 46. P. 1–13.

5. History of JavaScript and its Evolution. URL: https://www.linkedin.com/pulse/history-javascript-its-evolution-suleman-elahi (дата звернення: 22.05.2024).

6. Peter Gasston. The Modern Web: Multi-Device Web Development with HTML5, CSS3, and JavaScript. Сан-Франциско: No Starch Press. 2013. 464 p.

7. Responsive web design vs. adaptive: Which should you use? URL: https://www.wix.com/blog/responsive-vs-adaptive-design (дата звернення: 23.05.2024).

8. 12–8–4 Column System for Responsive Grids. URL: https://bootcamp.uxdesign.cc/12-8-4-column-system-for-responsive (дата зверненя: 23.05.2024).

9. Mobile-First vs. Desktop-First Design. URL: https://theandreflores.medium.com/mobile-first-vs-desktop-first (дата звернення: 23.05.2024).

10. Content-out Layout. URL: https://alistapart.com/article/content-out-layout/ (дата звернення: 23.05.2024).

11. Material Design. URL: https://m3.material.io/ – (дата звернення: 17.05.2024).

12. Human Interface Guideline URL: https://developer.apple.com/design/human-interface-guidelines (дата звернення: 17.05.2024).

13. Fluent 2. URL: https://fluent2.microsoft.design/ (дата звернення: 17.05.2024).

14. Gnome Human Interface. URL: https://developer.gnome.org/hig/ (дата звернення: 17.05.2024).

15. Graphic design trends for 2024. URL: https://www.adobe.com/express/learn/blog/design-trends-2024 (дата звернення: 24.05.2024).

16. What is Client-Server Architecture? Everything You Should Know. URL: https://www.simplilearn.com/what-is (дата зверненя: 24.05.2024).

17. Node.js — Run JavaScript Everywhere. URL: https://nodejs.org/en (дата звернення: 24.05.2024).

18. What is a webhook? URL: https://www.redhat.com/en/ (дата звернення: 24.05.2024).

19. The WebSocket API (WebSockets). URL: https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API (дата звернення: 24.05.2024).

20. User Personas: Your Guide to Building Personas for UX. URL: https://maze.co/guides/user-personas/ (дата звернення: 25.05.2024).

21. OBS Studio. URL: https://obsproject.com/ (дата звернення: 25.05.2024).

22. Restream: Create and Multistream Live Video. URL: https://restream.io/ (дата звернення: 25.05.2024).

23. StreamYard. URL: https://streamyard.com/ (дата звернення: 25.05.2024).

24. Adobe Photoshop. URL: https://www.adobe.com/products/photoshop.html (дата звернення: 17.05.2024).

25. GIMP – The GNU Image Manipulation Program. URL: https://www.gimp.org/ (дата звернення: 17.05.2024).

26. Protopea URL: https://www.photopea.com/ (дата звернення: 17.05.2024).

27. Adobe Illustrator | Розкішна графіка, створена для вас. URL: https://www.adobe.com/ua/products/illustrator.html (дата звернення: 17.05.2024).

28. CorelDRAW Graphic Suite. URL: https://www.coreldraw.com/ (дата звернення: 17.05.2024).

29. Inkscape – Draw Freely. URL: https://inkscape.org/ (дата звернення: 17.05.2024).

30. Figma – The Colaborative Interface Designer. URL: https://www.figma.com/ (дата звернення: 17.05.2024).

31. Penpot – The Design Tool for Design Beuty Product. URL: https://penpot.app/ (дата звернення: 17.05.2024).

32. AdobeXD. URL: https://helpx.adobe.com/xd/get-started.html (дата звернення: 17.05.2024).

33. WebStorm: The JavaScript and TypeScript IDE, by JetBrains. URL: https://www.jetbrains.com/webstorm/ (дата звернення: 26.05.2024).

34. Visual Studio Code - Code Editing. Redefined. URL: https://code.visualstudio.com/ (дата звернення 26.05.2024).

35. Ethan Marcotte. Responsive Web Design (Brief Books for People Who Make Websites, No. 4). Нью-Йорк: A Book Apart, 2012. – 150 p.

36. Steave Krug. Don't Make Me Think, Revisited: A Common Sense Approach to Web Usability (3rd Edition). Себастополь: O`Reily, 2013. 216 p.

37. Accessibility is our responsibility. URL: https://medium.muz.li/accessibility-is-our-responsibility-5f9627a89177 (дата звернення: 27.05.2024).

38. Dark Mode vs Light Mode in UX: Which Optimizes User Experience Better? URL: https://freedium.cfd/https://medium.com/@sannanmalikofficia (дата звернення 27.05.2024).

39. How to Design Accessible Dark Mode Interfaces. URL: https://medium.com/@tundehercules/designing-effective (дата звернення: 27.05.2024).

40. Cantarell TypeFace Family. URL: https://cantarell.gnome.org/ (дата звернення: 27.05.2024).

41. Materialize. URL: https://materializecss.com/ (дата звернення 28.05.2024).

42. HTML5test - How well does your browser support HTML5? URL: https://html5test.co/ (дата звернення: 29.05.2024).

43. “The Subtle Art of Rounded Corners: A UX/UI Perspective”. URL: https://bootcamp.uxdesign.cc/the-subtle-art-of-rounded-corners-a-ux-ui-perspective-a0274a90f27b (дата звернення: 26.05.2024).