МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Острозька академія»
Економічний факультет
Кафедра економіко-математичного моделювання та інформаційних технологій

КВАЛІФІКАЦІЙНА РОБОТА/ПРОЄКТ
на здобуття освітнього ступеня бакалавра

на тему: «Розробка модулю індивідуального навчання для освітньої платформи з елементами гейміфікації»

Виконав: студент 4 курсу, групи КН-41
першого (бакалаврського) рівня вищої освіти
спеціальності 122 Комп'ютерні науки
освітньо-професійної програми «Комп'ютерні науки»
Романюк Анна Сергіївна

Керівник: *викладач кафедри ЕММІ Т*
Красюк Богдан Віталійович

Рецензент: *Front-end Developer "DOODLE", LLC*
Місай Володимир Віталійович

РОБОТА ДОПУЩЕНА ДО ЗАХИСТУ

Завідувач кафедри економіко-математичного моделювання та інформаційних технологій _____ (проф., д.е.н. Кривицька О.Р.)

Протокол № 11 від «18» травня 2022 р.

Острог, 2022

Міністерство освіти і науки України
Національний університет «Острозька академія»

Факультет: *економічний*
Кафедра: *економіко-математичного моделювання та інформаційних технологій*
Спеціальність: 122 Комп'ютерні науки
Освітньо-професійна програма: Комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри економіко-математичного моделювання
та інформаційних технологій

_____ Ольга КРИВИЦЬКА
«____» _____20__ р.

З А В Д А Н Н Я
на кваліфікаційну роботу/проєкт студента

Романюк Анни Сергіївни

1. Тема роботи: "Розробка модулю індивідуального навчання для освітньої платформи з елементами гейміфікації"

керівник проєкту: Красюк Богдан Віталійович, викладач.

Затверджено наказом ректора НаУОА від 31 жовтня 2022 року №77.

2. Термін здачі студентом закінченої роботи/проєкту: 31 травня 2023 року.

3. Вихідні дані до роботи/проєкту: дана робота полягає у розробленні онлайн-платфоми для навчання, з використанням наступних сервісів: Trello, Milanote, Figma, Code with me.

4. Перелік завдань, які належить виконати: Проведення аналізу наявних аналогів, проведення аналізу потреб користувачів, створення User Flow та високодеталізованих скетчів, розробка дизайну інтерфейсу сторінок, реалізація дизайну сторінок у коді. Проведення тестування веб-доступності та тестування функціональності роботи платформи.

5. Перелік графічного матеріалу: рисунки, таблиці.

6. Консультанти розділів роботи:

Розділ		Підпис, дата

	Прізвище, ініціали та посада Консультанта	Завдання видав	Завдання прийняв
1	Красюк Б. В.	01.12.2022	01.12.2022
2	Красюк Б. В.	01.12.2022	01.12.2022
3	Красюк Б. В.	01.12.2022	01.12.2022

7. Дата видачі завдання: 01.12.2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів	Примітка
1	Затвердження теми роботи/проєкту	До 31.10.22	
2	Постановка технічного завдання	До 01.12.22	
3	Проведення аналізу аналогів та користувачів, створення User Flow	До 15.12.22	
4	Створення високодеталізованих скетчів для основних сторінок	До 06.02.23	
5	Створення інтерфейсу користувача, графіки	До 03.04.23	
6	Створення компонентів	До 17.04.23	
7	Імплементація в коді	До 05.05.23	
8	Тестування веб-доступності	До 05.05.23	
9	Функціональне тестування	До 15.05.23	
10	Попередній захист кваліфікаційної роботи/проєкту	До 18.05.23	
11	Здача кваліфікаційної роботи/проєкту на кафедру	До 31.05.23	

Студент: _____ Анна РОМАНЮК

Керівник кваліфікаційної роботи: _____ Богдан КРАСЮК

АНОТАЦІЯ
кваліфікаційної роботи
на здобуття освітнього ступеня бакалавра

Тема: Розробка модулю індивідуального навчання для освітньої платформи з елементами гейміфікації

Автор: Романюк А. С.

Науковий керівник: Красюк Б. В.

Захищена «……»…… 20__ року.

Пояснювальна записка до кваліфікаційної роботи: *63 с., 25 рис., 2 табл., 4 додатки, 15 джерел.*

Ключові слова: *онлайн-навчання, веб-сервіс, розроблення платформи, прототипування, інтерфейс користувача, тестування дизайну, доступність.*

Короткий зміст праці:

Завданням кваліфікаційної роботи/проєкту було розроблення освітньої онлайн-платформи, основним фокусом на забезпечення якості освіти та доступності користувачів. Для розробки макетів та імплементації дизайну були використані графічний редактор Figma та бібліотека Material UI, що дозволило створити зручний та привабливий інтерфейс користувача. Клієнтська частина проекту була описана та розроблена за допомогою інструменту Code with me. Робота передбачає виконання кількох етапів, включаючи затвердження теми роботи, постановку технічного завдання, проведення аналізу аналогів та користувачів, створення високодеталізованих скетчів для основних сторінок, створення інтерфейсу користувача та графіки, розробку компонентів, імплементацію в коді, тестування веб-доступності та функціонального тестування. Особлива увага приділяється розробці, що відповідає вимогам доступності, зокрема забезпеченню збільшення веб-контенту та зручної взаємодії для користувачів з різними потребами. Результатом цієї роботи буде повноцінна платформа для онлайн-навчання, яка дозволить користувачам отримувати якісну освіту зручно та ефективно. Застосування графічного редактора Figma та бібліотеки Material UI допомогло забезпечити злагоджений та привабливий дизайн інтерфейсу, використовуючи сучасні технології та стандарти. Клієнтська частина проекту, описана за допомогою інструменту Code with me.

The task of the qualification work/project was to develop an online educational platform, with the main focus on ensuring the quality of education and user accessibility. The Figma graphic editor and the Material UI library were used to develop layouts and implement the design, which allowed us to create a user-friendly and attractive user interface. The client side of the project was described and developed using the Code with me tool. The work involves several stages, including approving the topic of work, setting up a technical task, analyzing analogs and users, creating highly detailed sketches for the main pages, creating a user interface and graphics, developing components, implementing them in code, testing web accessibility, and functional testing. Particular attention is paid to development that meets accessibility requirements, including increasing web content and convenient interaction for users with different needs. The result of this work will be a full-fledged online learning platform that will allow users to receive quality education conveniently and efficiently. The use of the Figma graphic editor and the Material UI library helped to ensure a coherent and attractive

interface design using modern technologies and standards. The client side of the project described using the Code with me tool.

ЗМІСТ

ВСТУП

Мета

За останні роки підхід до навчання значно змінився і багато установ переходять до онлайн-освіти. Це в більшій мірі пов’язано із пандемією COVID-19 у 2020 році.

Сьогодні дистанційне навчання популярне серед молоді, оскільки формат традиційного навчання з одногрупниками не є доступним для усіх студентів. Дистанційне навчання надає можливість навчатися без переїзду в інше місто, заощаджує час та кошти.

Досі навчальні заклади стикаються з проблемою вибору зручних платформ для студентів. На сьогодні існує багато платформ як українських, так і іноземних де можна навчатися, у всіх них є свої переваги та недоліки. Наприклад популярними є Prometheus, EdEra, Coursera, Classroom, Udemy та ін.

Під час пандемії було виявлено, що багато навчальних закладів не мають достатньої кількості технічних ресурсів для забезпечення якісного онлайн-навчання. Також, не було підготовлено інфраструктури для переходу на дистанційне навчання, що створило серйозні труднощі в проведенні занять.

Створення власної платформи для онлайн-освіти дозволяє розширити можливості доступу до навчальних матеріалів та забезпечити студентів зручним та інтуїтивним інтерфейсом. Така платформа робить навчальний процес зручним та гнучким.

Створення такої платформи є важливим завданням у сучасному світі. Вона забезпечує доступ до навчання для різних категорій студентів, розширює можливості навчання та сприяє підвищенню ефективності процесу.

РОЗДІЛ 1. ЗАГАЛЬНІ ПОЛОЖЕННЯ

1.1. Опис предметного середовища

Робота була поділена на декілька етапів: дизайн, розробка, тестування. Розділення на етапи дозволить збільшити ефективність та якість розробки

На етапі дизайну ми встановлюємо вимоги до інтерфейсу платформи та розпочинаємо розроблення дизайн. На етапі програмування ми розробляємо функціональність платформи згідно дизайну. І останній етап це тестування. На цьому етапі ми перевіряємо як працює платформа та окремі її модулі.

1.2. Постановка задачі

Повна постановка завдання полягає у створенні зручної платформи для дистанційного навчання, що дозволить студентам отримувати нові навички. Для досягнення цієї мети необхідно вирішити наступні завдання:

1. Розробка зручного та легкого в користуванні інтерфейсу користувача, що дозволить знаходити необхідну інформацію без зайвих проблем.
2. Забезпечення різноманітності навчального матеріалу.
3. Забезпечення можливості створення нових навчальних курсів, що дозволить користувачам вибирати найбільш цікаві курси.
4. Створення можливості відстеження успішності студентів, для кращої ефективності навчання.

З метою вирішення поставлення завдань ми плануємо розробити платформу яка включатиме в себе наступні компоненти:

- Інтуїтивно зрозумілий та зручний інтерфейс
- Різноманітний навчальний матеріал
- Розклад зустрічей та календар
- Можливість створення навчальних курсів
- Можливість створення та редагування навчальних матеріалів
- Можливість моніторингу успішності студентів

Сучасна платформа для онлайн-навчання повинна відповідати потребам студентів та викладачів. Оскільки користувачі часто використовуватимуть платформу для дистанційного навчання, то затримки та повільна робота будуть негативно впливати на досвід користувачів.

1.2.2. Вимоги з боку викладача

- Зручний інтерфейс
- Створення власних курсів та уроків
- Легкий доступ до курсів
- Можливість перевіряти роботи
- Можливість оцінювання та моніторинг рейтинг

1.2.3. Вимоги з боку студента

- Зручний інтерфейс
- Швидкий та легкий доступ до матеріалів
- Легка подача виконаних завдань
- Відгуки від викладачів

ВИСНОВОК

У цьому розділі було з’ясовано, що для ефективної роботи необхідно ділити великий проєкт на декілька етапів, ми поділили на дизайн, розробку та тестування. Це дозволить ефективно керувати та швидше знаходити проблеми та їх рішення. Дизайн зосереджений на створення привабливого та зручного дизайну, розробка на реалізації, а тестування на перевірці якості розробки.

Було описано постановку задачі, з конкретними завданнями для створення зручної платформи онлайн-освіти. Сформовано основні вимоги з сторони викладача та сторони студента щодо функціональності платформи.

РОЗДІЛ 2. ІНФОРМАЦІЙНЕ ТА МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

2.1. Процес впровадження нового компонента

Процес впровадження нового компоненту — це певна послідовність кроків, які необхідно виконати для додавання компоненту в існуючий проєкт. Такий процес включає в себе такі етапи: збір вимог, розробка скелетону, розробка компоненту у середовищі Figma, імплементація в коді, тестування та верифікація, збір побажань або вад та внесення правок при необхідності.

Впровадження нового компоненту допомагає зробити проєкт більш ефективним швидким та зручним для користувачів. Компоненти можуть використовуватися багатьма різними частинами проєкту, тому їхнє впровадження допомагає покращити якість розробки та пришвидшити її.

Для зручності впровадження компонента було створено UML-діаграму (див. Додаток А) з описом усіх етапів, які необхідно пройти:

- Збір необхідних вимог
- Розробка скетчу компонента
- Розробка дизайну компонента
- Імплементація в коді
- Тестування
- Збір побажань та внесення правок

2.1.1. Збір вимог

Компонент бокової навігації — це ефективний інструмент для навігації користувачів на сайті.

Компонент має наступні вимоги:

1. Заголовок курсу. Заголовок має бути розташовано у верхній частині компонента та містити назву курсу, який проходить студент. Типографіка має бути чіткою та зрозумілою.

2. Пункти меню. Пункти мають бути розташовані під заголовком курсу та містити наступні назви: уроки, студенти, рейтинг, завдання. Кожний пункт має мати відповідну іконку.

3. Зручна навігація. При наведенні на певний пункт меню, користувач має бачити взаємодію з пунктами меню. Тобто при наведенні пункт меню змінює колір, а при натисканні колір робиться темнішим. Також, при натисканні на пункт меню, користувач має перейти на відповідну сторінку.

4. Стиль інтерфейсу. Компонент має відповідати єдиному стилю платформи та мати чіткий та сучасний дизайн.

2.1.2. Розробка скетчу компоненту

Створення скетчу передбачає створення компоненту, який містить лише основні структурні елементи. Під час розробки низькодеталізованого скетчу, макет компонента відображає основні елементи та їх взаємодію. Для компонента бокової навігації скелетон містить заголовок курсу та декілька пунктів меню (див. Рис. 2.1).

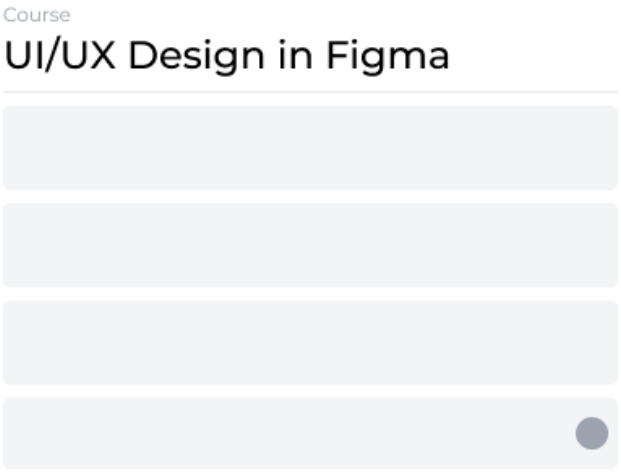


Рис. 2.1. Низькодеталізований скетч навігації
Джерело: Розроблено автором

Завдяки розробці скетчу було попередньо оцінено відповідність вимогам, побачити недоліки на ранньому етапі та швидко внести необхідні правки ще до початку розробки дизайну.

2.1.3. Розробка компоненту у середовищі Figma

Після створення скетчу компонента, наступним кроком є розробка максимально детального дизайну компонента в середовищі Figma. Дизайн компонента має відповідати вимогам, зібраним на попередньому етапі, і повинен бути зрозумілим для користувачів.

При створенні дизайну компонента навігації було додано різні елементи, іконки, текстові поля, бейдж та стан при наведенні (див. Рис. 2.2)

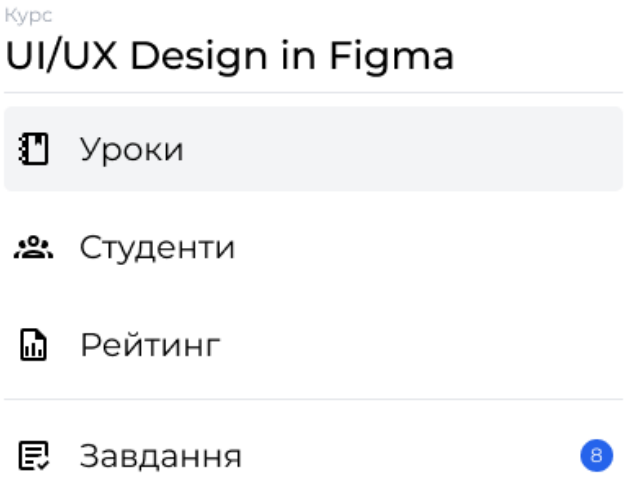


Рис. 2.2. Дизайн компоненту навігації
Джерело: Розроблено автором

Оскільки, такий компонент буде використовуватися на декількох сторінках, його було додано до бібліотеки компонентів проєкту, щоб його можна було швидко знаходити та використовувати.

2.1.4. Імплементація в коді

Однією із зручних можливостей фреймворку Nuxt 3 є наявність автоматичного імпорту багатьох файлів, таких як, наприклад, файли компонентів. Хоча й це накладає певні обмеження, такі як необхідність слідувати певній структурі організації файлів, але є беззаперечною перевагою при використанні.

На початку роботи над проектом було прийняте рішення створити окремі папки для різних груп компонентів. В даному випадку, оскільки компонент бокової навігації

стосується роботи з курсами, то й зберігатися він повинен у папці для компонентів курсів.

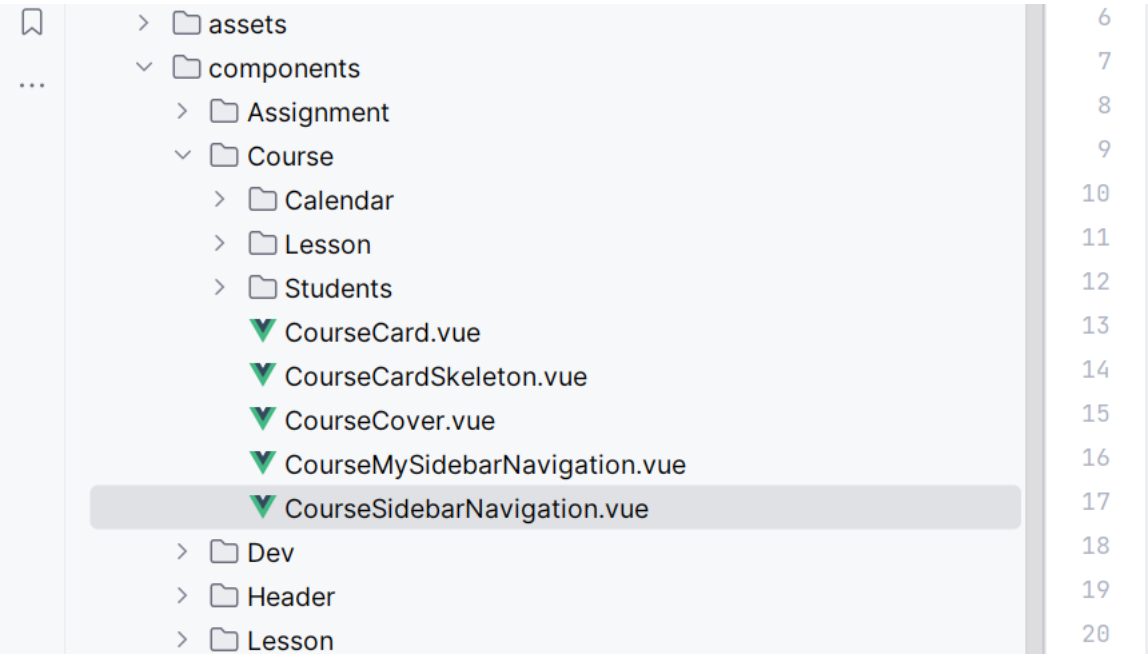


Рис. 2.3. Компонент у дереві файлів
Джерело: Фото з екрану

Як видно з фото, компонент має назву CourseSidebarNavigation там має розширення .vue. Це спеціальне розширення для компонентів Vue. Цей файл зберігається у папці Course, що в свою чергу знаходиться у папці components. Папка components це спеціальна папка у які потрібно зберігати усі компоненти проекту на Nuxt 3. Це дасть можливість використовувати їх без попереднього імпорту. У такому випадку, необхідно розуміти, що ім’я компоненту буде мати назву, яка складається з назви папки у якій він знаходиться, та назви файлу. Але якщо назва папки є в назві файлу, то ця частина не дублюється, тому незважаючи що шлях до файлу є Course/CourseSidebarNavigation.vue, назва компоненту буде CourseSidebarNavigation.

Лістинг 2.1

```
<div class="basis-1/5">
 <CourseSidebarNavigation
  :course-id="route.params.id"
  :course-title="course?.data.title"
  class="w-full"
```

```
 />
</div>
```

Компоненти Vue складаються з 3 частин:

1. Template – html код компоненту (шаблон), опис структури елементів які будуть відображатися на сторінці
2. Script – JavaScript або TypeScript код компонента, відповідає на логіку та поведінку цього компонента
3. Style – CSS код компонента, відповідає за стилізацію компонента.

Лістинг 2.2

```
<template>

</template>

<script setup lang="ts">

</script>

<style scoped>

</style>
```

У кожної з цих секцій є також додаткові налаштування, наприклад lang=”ts” у секції script відповідає за вибір мови на якій буде написано логіка компоненту, у цьому випадку це TypeScript.

Розглянемо детальніше секції у компоненті CourseSidebarNavigation

2.1.4.2. Шаблон компонента

Як і інші компоненти Vue, секція template повинна мати лише один дочірній тег, у цьому випадку це тег div. У цій секції дозволено використовувати усі функції та змінні які створені у секції script. Для виведення змінних використовуються подвійні фігурні дужки

Лістинг 2.3

```
<div class="text-2xl font-medium">{{ props.courseTitle }}</div>
```

2.1.4.3. Логіка компоненту

Як було описано раніше, секція script використовується для опису JavaScript коду для створення логіки компонента. Конкретно у компоненті CourseSidebarNavigation ця секція невелика і має наступний вигляд.

Лістинг 2.4

```
<script lang="ts" setup>
const props = defineProps<{
  courseId: string;
  courseTitle?: string;
}>();
</script>
```

Тут з використанням функції defineProps оголошуються параметри компонента які можуть бути передані при його використанні (див. Лістинг 2.1).

2.1.4.4. Стилі компонента

Секція style використовується для опису стилів. У компоненті CourseSidebarNavigation вона має наступний вигляд:

Лістинг 2.5

```
<style scoped>
.link {
  @apply cursor-pointer space-x-4 whitespace-nowrap rounded px-1.5 py-3 text-xl transition-colors
hover:bg-gray-200;
}

.active {
  @apply bg-gray-100;
}
</style>
```

При її оголошенні також передається налаштування scoped. Цим налаштовується обмеження дії стилів лише на ті теги, які описані в секції template цього компонента.

Стилі для класів описуються не зі звичайний кодом CSS, а з використанням класів TailwindCSS та її директиви @apply.

2.1.5. Тестування компонента

Для того, щоб переконатися у правильності функціонуванні компонента, проводиться ручне тестування функціональності. Етап тестування функціональності компонента - це процес перевірки, чи працює компонент згідно з очікуваннями та відповідає специфікаціям. При тестуванні відбувається перевірка вимог, які були зазначені на першому етапі.

При наявності вад, вони записуються у Trello, для зручного відслідковування та роботи. У компоненті навігації вади відсутні, та робота відбувається згідно зазначених вимог. Тому, нема необхідності записувати вади

2.1.6. Збір побажань та внесення правок

Після завершення тестування компонента можуть виникнути різні ситуації, які вимагають внесення певних покращень або виправлень. Оскільки, у випадку компонента навігації не було виявлено проблем, можуть обговорювати покращення компонента.

Процес збору побажань та внесення правок може включати в себе кілька етапів. Спочатку необхідно визначити, які саме зміни потрібно внести в компонент, на основі результатів тестування, та вияснити чи дійсно зміни необхідні.

Наступним кроком є внесення покращень у компонент. Після внесення змін необхідно перевірити компонент знову, щоб переконатися, що всі виправлення та покращення були реалізовані правильно та не порушують роботу компоненту.

2.2. Підготовка до розробки дизайну

Створення дизайн-концепцій платформи “Versite” є важливим етапом, та допомагає вирішити ряд проблем. Дизайн-концепції допомагають визначити стиль платформи, функціональність та інтерфейс користувача. Дизайн має бути зроблений з урахуванням тенденцій та інновацій у сфері дизайну та онлайн-освіти.

Задля ефективності процесу розроблення дизайну, весь процес варто поділити на дві фази — підготовка (аналіз ринку, моделювання процесу взаємодії користувача з системою) та розроблення інтерфейсів та їх елементів. У цьому розділі буде описано

підготовчі процеси та планування подальшої реалізації створених моделей у спеціалізованому програмному забезпеченні.

2.2.1. Етапи розробки дизайну

Повний процес розробки дизайну було розділено на наступні етапи:

1. Дослідження аналогів та користувачів: дослідження допомагає дизайнерам зрозуміти, які функції та інтерфейси уже присутні на ринку, та які з них є найбільш ефективними. Це дозволяє уникнути повторення помилок та зробити оптимізований продукт.
2. Створення скетчів: скетчі — це швидкі ескізи, які малюються на папері або у електронних програмах з метою візуалізації ідей та концепцій дизайну. Основні переваги використання скетчів:
   - Швидко та ефективно відобразити ідеї, що в подальшому допоможе зберегти час на вирішенні проблем із дизайном.
   - Встановлення якісної співпраці, що дозволить краще зрозуміти як реалізовуються ідеї та як проєкт буде виглядати.
   - Виявити проблеми на ранньому етапі, та сприяти швидкому їх рішенню.
3. Розробка дизайн-макету: цей етап передбачає створення детальної візуальної концепції продукту, включаючи вибір кольорової гами, типографіки та графічних елементів.
4. Тестування дизайну: це процес оцінки ефективності дизайну з точки зору користувачів, щоб забезпечити максимально якісну взаємодію між користувачем та продуктом. Мета такого тестування перевірити, наскільки дизайн відповідає потребам та очікуванням користувачів.

2.2.2. Дослідження аналогів

На сьогоднішній день на українському ринку є багато онлайн платформ, які допомагають отримувати нові навички чи отримати нову освіту. Для реалізації власної

платформи для онлайн-освіти було вивчено як і популярні платформи такі як Prometheus та EdEra, так і доволі нову платформу Creo.ua. Розглянемо кожну з них детальніше.

Prometheus — українська онлайн-платформа яка містить в собі понад 50 тис. курсів. На платформі можна обрати як безкоштовне так і платне навчання на різні теми. В основі навчання лежить використання відео-уроків, тестів, та можливість спілкування з викладачами та іншими студентами на форумі.

Інтерфейс Prometheus доволі простий та зрозумілий. Головна сторінка містить курси з навчальними матеріалами, які можна відсортовувати за різними категоріями. Це дозволяє швидко знайти необхідний курс.

EdEra — українська онлайн-платформа, яка доволі давно знаходиться на українському ринку. Основна їхня мета — надавати якісну освіту для усіх бажаючих на різні теми.

EdEra має важкий для розуміння та користування інтерфейс. Головна сторінка містить інформацію про нові проєкти та власний блог. Окремої уваги заслуговує система оповіщення про нові матеріали. Проте, інформація не структурована та важко знаходити власний кабінет з курсами.

Creo — це нова та успішна платформа для навчальних курсів, яка пройшла на заміну російської платформи Creo.games. Навчання відбувається у форматі відео-уроків та текстових матеріалів.

Платформа Creo має інтуїтивно зрозумілий інтерфейс, що складається з головної сторінки з курсами та персональної сторінки користувача. Інтерфейс дещо схожий з EdEra, проте, більш зрозумілий.

Для зручності аналізу створено таблицю у якій описано основні переваги та недоліки наявних аналогів (див. Таблиця 1).

Таблиця 1. Порівняльна таблиця аналогів онлайн-платформ

Особливості	Versite	Prometheus	Creo	EdEra
Формат навчання	онлайн	онлайн	онлайн	Онлайн

Модуль індивідуального навчання	+	-	+	+
Розклад зустрічей/календар	+	+	-	+
Можливість створювати курси зі своїми матеріалами	+	+	+	+
Простий та інтуїтивний інтерфейс	++	+-	++	--
Гейміфікація	+	-	-	-

Як можна помітити, кожна з платформ має як переваги так і недоліки. Кожна платформа має онлайн-навчання. Розклад зустрічей є не у всіх, але розклад у EdEra гнучкий, тоді як на Creo та Prometheus доволі жорсткий, та необхідно дотримуватися встановлених дедлайнів. Якщо говорити за інтерфейс, у Creo він найбільш зрозумілий та інтуїтивний, у інших необхідна додаткова навігація та деякий час що зрозуміти призначення елементів інтерфейсу.

Отже, порівнявши три платформи можна зробити такі висновки:

1. Оскільки Versite нова розробка, у ній нема всього необхідного функціоналу порівняно з іншими платформами.
2. Versite відповідає необхідним потребам користувачів, оскільки вона містить вимоги, які засновані на попередньому досвіді використання інших платформ.
3. Versite має свої переваги у порівнянні з існуючими платформами на ринку, однак ми можемо налаштувати її під потреби, робити більш ефективною та зручною для користувачів.

2.2.3. Дослідження користувачів

Після проведення дослідження аналогів необхідним етапом є дослідження проблем користувачів на цих платформах.

Prometheus — основні проблеми з цією платформою здебільшого пов’язані з технічними проблемами, великим навантаженням системи, що призводить до повільної роботи платформи, це негативно впливає на користувача.

Creo — найбільш поширеною проблемою є проблеми з навігацією та пошуком необхідних матеріалів. Також є технічні проблеми з переходами на сторінках та відображенням матеріалів.

EdEra — найбільша проблема платформи — це надмірний обсяг інформації, який є заважким для користувача та складним для його сприйняття. Для користування цією платформою необхідний деякий час на вивчення її інтерфейсу.

В процесі взаємодії користувачі постійно стикаються з проблемами та незручностями, які впливають на їх мотивацію користування платформою. Одним із способів зменшення цих проблем є використання User Flow (див. Додаток С). User Flow — це детальна діаграма, яка показує послідовність кроків, які проходить користувач для досягнення певної мети на веб-сайті. Після детального дослідження аналогів ми створили максимально зручну карту шляху користувача, це допомогло нам виявити можливі проблеми, такі як:

1. Помилки у навігації. Карта користувача допомагає швидко виявити складнощі в навігації та показати, як користувач переходить між сторінками.
2. Велика кількість інформації. Карта користувача допомагає виявити надмірну кількість інформації на сторінках.
3. Помилки в дизайні та розташування елементів. Карта користувача допомагає виявити відсутність необхідних елементів або неправильне розташування.

На створеному User Flow є різні геометричні фігури:

- Ромб (початок) — подія, яка визначає початок шляху, коли користувач щойно зайшов на платформу та вибір який він має зробити, вказати, чи хоче користувач зареєструватися чи авторизуватися.
- Квадрат — вказує кроки, які здійснює користувач, наприклад шукає інформацію.
- Прямокутник із іконкою блискавки — означає, що дія була виконана, наприклад створено курс, або урок.

ВИСНОВОК

В даному розділі було описано різні етапи впровадження нового компонента, та детально описано кожен з них, а саме етап збору вимог (формування запиту на новий функціонал), розробка скетчу (низькодеталізованого каркасу компонента), розробка дизайну (готового вигляду частини інтерфейсу), імплементація в коді (створення компонентів з використання фреймворку), тестування компонента (перевірка функціональності компонента), збір побажань та внесення правок (при наявності формуються побажання для покращення та впроваджуються).

Було проведено аналіз аналогів, таких як Prometheus, EdEra та Creo.ua. Платформи є популярними серед користувачів, але мають свої як переваги, так і недоліки. Основні проблеми, які можна виокремити, включають незручний інтерфейс, перевантаженість сторінок інформацією та проблеми з продуктивністю. Такі проблеми негативно впливають на користувачів, призводячи до зниження задоволення від використання платформи, збільшення кількості помилок.

Також було проведено аналіз користувачів, визначено їхні основні проблеми при користуванні наявними аналогами. Було виявлено що проблеми користувачів пов’язані з перезавантаженістю інформації та відсутністю інтуїтивного дизайну. Велика кількість інформації, яка відображається на сторінках є заплутаною та важкою для сприйняття. Інтуїтивний дизайн сприяє легкості та зрозумілості взаємодії користувача з платформою, дозволяючи їм легко знаходити необхідні функції та виконувати завдання без зайвих зусиль. Проблема відсутності інтуїтивного дизайну полягає у тому, що користувачам може бути складно орієнтуватися та розуміти, як користуватися платформою, що може призводити до погіршення продуктивності та незадоволення користувачів.

Після проведення аналізу аналогів та користувачів було розроблено user flow, щоб виявити проблеми та недоліки на ранніх етапах роботи з платформою. User flow допоміг визначити, що перевантаженість інформацією та незручний інтерфейс є ключовими

факторами, які впливають на задоволення та продуктивність користувачів. Розробка user flow сприяла виявленню проблем, пов'язаних з перезавантаженістю інформацією та відсутністю інтуїтивного дизайну, що в свою чергу надало підґрунтя для подальших зусиль у вирішенні цих проблем та покращенні користувацького досвіду.

РОЗДІЛ 3. ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1. Засоби розробки

Для розробки онлайн-платформи було використано різноманітні засоби розробки, що включають у себе інструменти для планування, проектування та програмування. Основним критерієм вибору засобів розробки є їхні можливості відповідати вимогам завдання, а також їхня доступність та популярність серед розробників.

3.1.1. Trello

Trello — це онлайн-інструмент для розподілу та моніторингу виконання завдань. Сервіс надає можливість спільно працювати над проєктом з колегами та коментувати завдання. Також має мобільний додаток, що робить його доволі зручним у використанні.

Головною перевагою використання цього сервісу є можливість створення завдань та розподіл між різними стадіями виконання. Для відображення кожної з цих стадій створюється окрема дошка, на яку необхідно прикріпити необхідну картку завдання. Для розроблення проєкту було створено три дошки які відображають різні стадії реалізації певного запланованого функціоналу (див. Рис. 3.1):

1. Треба зробити — перша дошка на яку потрапляє новостворене завдання.
2. Робиться — на цю дошку перетягуються завдання, над якими наразі введеться робота. Це дозволяє бачити які з завдань виконуються та не робити подвійну роботу.
3. Виконано — це фінальна дошка, на яку перекидаються усі виконанні завдання.

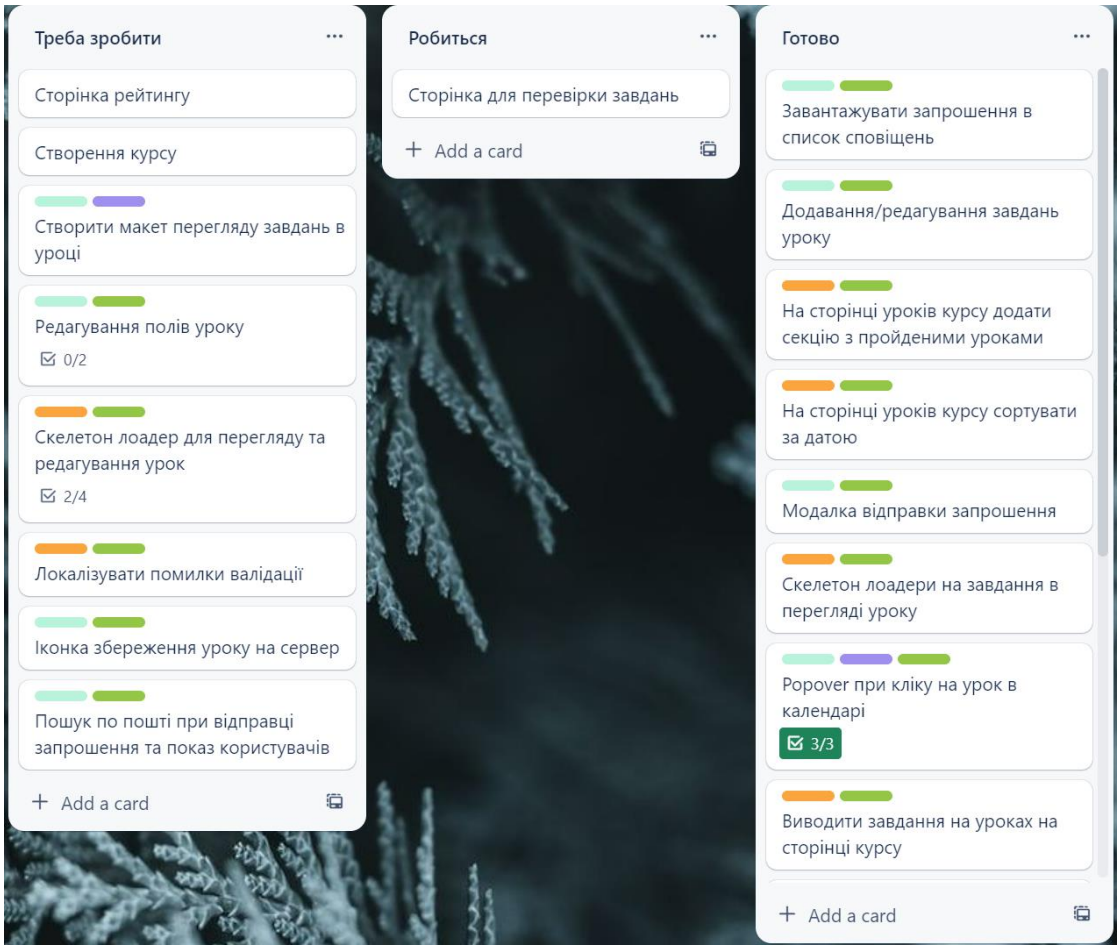


Рис. 3.1. Список завдань у Trello

Джерело: Фото з екрану

Можливості сервісу Trello:

- Створення дошок
- Створення списків та карток
- Призначення завдань для різних виконавців
- Додавання коментарів та файлів
- Встановлення пріоритетів завдань
- Перегляд завдань у календарному поданні
- Моніторинг процесу
- Інтеграція з іншими сервісами

Trello є доволі потужним інструментом в керуванні проєктом, який дозволяє швидко та зручно організовувати робочі завдання та проєкти різної складності.

3.1.2. Milanote

Milanote — це візуальний онлайн-інструмент, який призначений для роботи з ідеями, дозволяє швидко організовувати всі свої матеріали. Це зручний простір, у якому можна легко організовувати проєкт, а інтерфейс перетягування дозволяє розміщувати необхідні матеріали так, як буде зручно для користувача в нескінченному, віртуальному просторі. Всі дошки користувач може стилізувати на свій смак (див. Рис. 3.2).

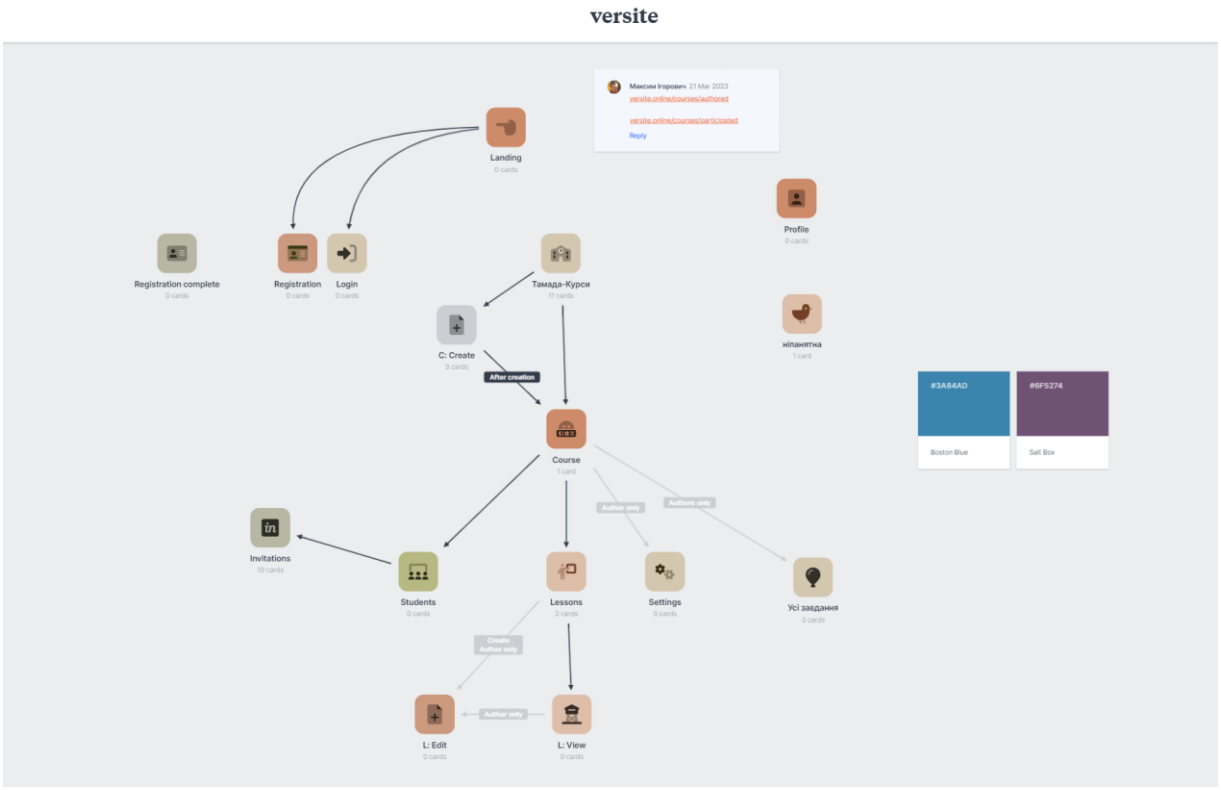


Рис. 3.2. Дошки проєкту

Джерело: Фото з екрану

Під час роботи з ідеями було створено декілька дошок та стилізовано під потреби. На кожну дошку можна перейти на працювати всередині них. Перейшовши на дошку “Invitation” (див. Рис. 3.3), можна переглянути її вміст.

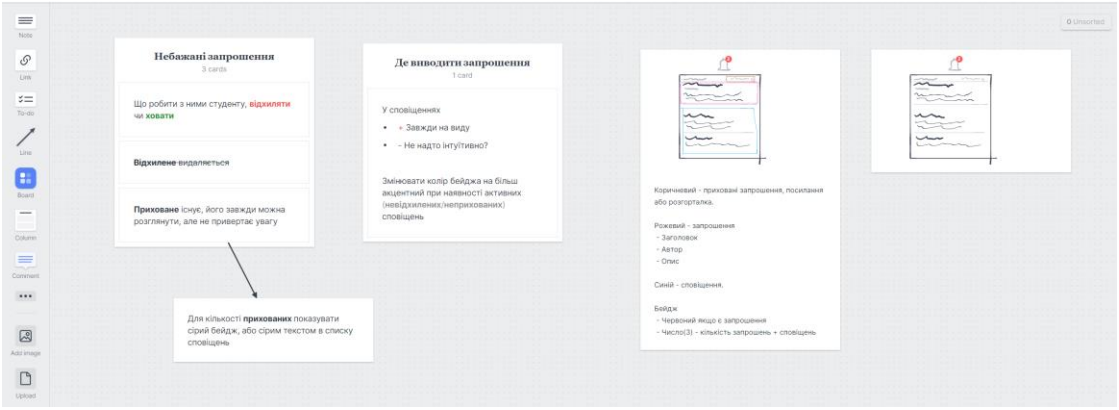


Рис. 3.3. Скетчі використання запрошень на платформі

Джерело: Фото з екрану

У дошці створені декілька колонок для зручного опису запрошень, наприклад опис запрошень, які користувач не хотів би бачити, у якому місці їх краще вивести. Також, можна побачити схематичне зображення виведення сповіщення (див.), з описом до зображення.

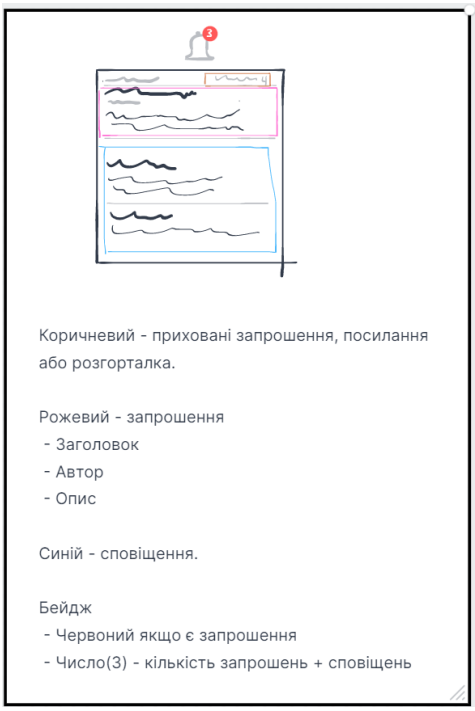


Рис. 3.4. Схематичне зображення запрошення на курс

Джерело: Фото з екрану

3.1.3. Figma

Для дизайну було використано онлайн-інструмент Figma. Доцільно буде розглянути чому саме цей інструмент, адже відомо що ж є інші інструменти, наприклад Adobe XD чи Adobe Photoshop (див. Таблиця 2).

Таблиця 2. Порівняння інструментів для створення дизайну

Особливості	Figma	Adobe XD	Adobe Photoshop
Тип програми	Онлайн-інструмент	Десктоп-програма	Десктоп-програма
Наявність автоматичного збереження	+	+	-
Робота з векторними елементами	+	+	+
Робота з растровими елементами	-	+	+
Робота з векторною типографікою	+	+	+
Робота з повторювальними елементами	+	+	-
Підтримка командної роботи у реальному часі	+	+	-
Система контролю версій	+	+	-
Прототипування	+	+	-
Експорт у зручних форматах	+	+	+
Підтримка роботи з бібліотеками	+	+	-

Отже, оскільки Figma є онлайн-іструментом, це надає більшої гнучкості та спрощує роботу в команді, оскільки може бути використана з будь-якого пристрою з підключенням до інтернету, навіть з мобільного телефону.

Також було використано ряд плагінів, таких як:

- Iconify — це плагін, який дозволяє легко знаходити і використовувати іконки. За допомогою плагіну можна переглядати безліч наборів іконок з популярних бібліотек. Однією з ключових особливостей iconify є можливість пошуку іконок за ключовими слова, що економить багато часу.
- Font Fascia — це типографічний плагін, який надає надає простий інтерфейс для вибору та керування шрифтами з такими можливостями, як попередній перегляд шрифтів у реальному часі, регулювання розміру шрифту, ваги шрифту та інтервалу між літерами. Він також дозволяє впорядковувати шрифти, створюючи власні групи шрифтів, що полегшує доступ до ваших улюблених шрифтів або шрифтів, які ви використовуєте найчастіше.

3.1.4. Code with me

Code with me — це сервіс для спільної роботи над кодом, надана компанією JetBrains, розробниками WebStorm та інших IDE. Сервіс дозволяє віддалено працювати спільно над кодовою базою в реальному часі, незалежно від фізичного місцезнаходження. Code with me спрямована на поліпшення співпраці та спрощення процесу командного програмування та віддаленої роботи.

Основні функції Code With Me:

- Розробка в реальному часі. Code with me дозволяє спільно та одночасно працювати над проєктом. Учасники командної роботи можуть редагувати файли, переходити по коду та використовувати налагодження.

- Безпечне підключення. З’єднання між учасниками встановлюється за допомогою криптографічного end-to-end шифрування. Це робить безпечними сесії спільної роботи та запобігає витоку даних.
- Швидкий доступ. Code with me, спрощує процес початку спільної роботи над проєктом. Після запуску плагіну, всі запрошенні учасники отримають унікальне посилання, за допомогою якого вони можуть швидко приєднатися до сесії без необхідності додаткових кроків.
- Переадресація портів (port forwarding). У сервісі Code with me веб-доступ до віддаленого робочого середовища зазвичай забезпечують шляхом переадресації портів. Це означає, що коли один розробник починає сесію спільної роботи, його локальний порт, на якому запущений веб-сервер або додаток, може бути перенаправлений, щоб інші учасники могли мати доступ до нього через цю сесію.
- Доступ до терміналів. Code with me також забезпечує доступ до терміналів. Учасники можуть використовувати спільний термінал, де кожний може вводити команди і бачити результати їх виконання у реальному часі.
- Налаштування дозволів. Сервіс дозволяє проводити налаштування дозволів для окремих користувачів. Наприклад, дозвіл на редагування файлів, дозвіл на взаємодію з публічними терміналами, дозвіл на перегляд та/або виконання існуючих конфігурацій запуску, дозвіл на перегляд та/або взаємодію з іншими інструментами (Gradle, Ant, Maven).
- Вбудовані аудіодзвінки та відеодзвінки. Code with me надає вбудовані застосунки для аудіо- та відеозв’язку, що дозволяють учасникам обговорювати проєкт у режимі реального часу під час спільної роботи.
- Підсвічування коду та навігація. Code with me надає синхронне підсвічування коду, що дозволяє учасникам бачити зміни у реальному часі. Він також дозволяє синхронну навігацію по коду.

3.2. Розробка дизайну

3.2.1. Створення скетчів

Як вже було описано раніше (див. 2.2.1), перший етап практичної реалізації дизайну — це створення скетчів, які допомагають визначитися з дизайн-концепцією. Мета цього етапу полягає у візуалізації ідеї та перевірки на практиці.

У проєкті було використано високодеталізовані скетчі. Це варіант, при якому ми створюємо скетчі, які дозволяють створити майже завершений дизайн перед розробкою. Такий варіант дозволяє візуалізувати дизайн платформи з усіма можливими деталями, та переглядати з різних кутів.

Наприклад, завдяки створенню скетчу сторінки “запрошення на курс” (див. Рис. 3.5), ми оцінили розміщення графічних елементів, наявність картинок та використання типографіки.

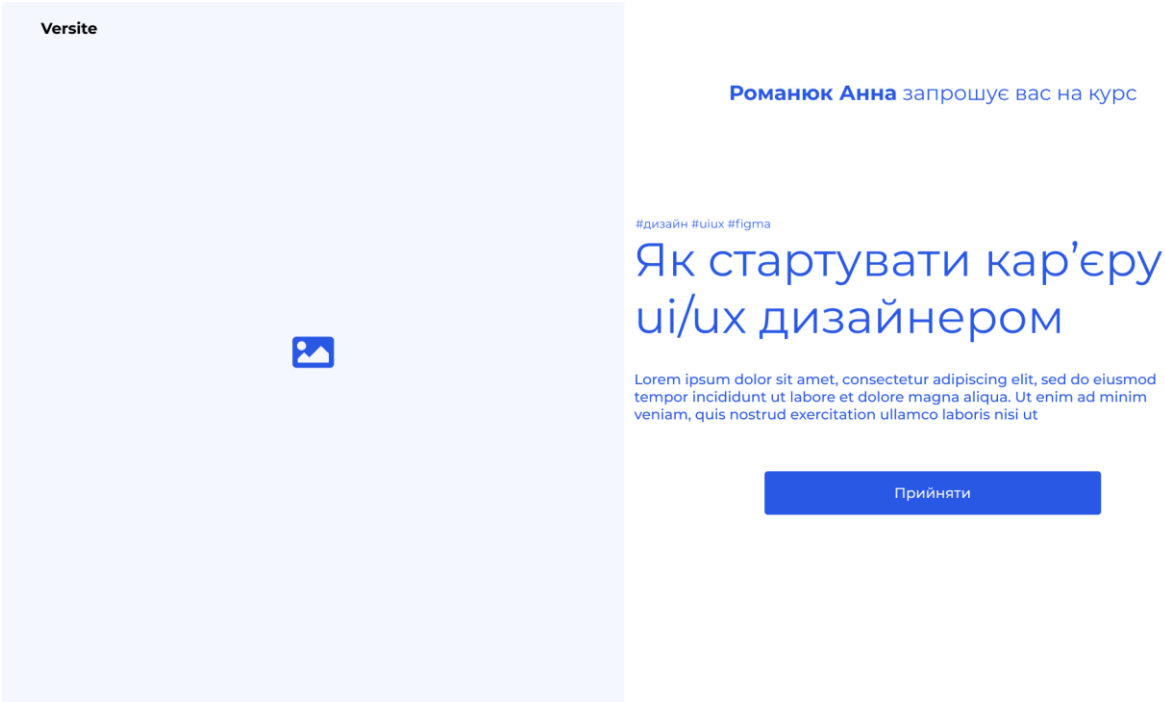


Рис. 3.5. Скетч сторінки “запрошення на курс”

Джерело: Розроблено автором

Після створення скетчу сторінки, було визначено загальну концепцію та з’ясовано як будуть розміщені графічні елементи. Функціонально, ми отримали робочу сторінку,

на якій є обкладинка курсу, відправник запрошення, заголовком, коротким описом та закликом до дії.

Під час створення скетчу, ми зосередилися на основних елементах, таких як відправник запрошення, опис курсу та заклик до дії. Це дозволило швидко визначитися із дизайном та необхідними додатковими елементами.

Після створення декількох таких скетчів, ми створили стилі, для єдиного корпоративного стилю платформи та забезпечення швидшої та ефективнішої роботи:

- Типографіка (єдиний шрифт, та стандартизація розмірів)
- Кольорова палітра
- Єдиний сет іконок

3.2.1.1. Типографіка

Вибір шрифтів це один з найважливіших етапів у роботі. Для збереження єдиного стилю та уникненню проблем при розробці, був обраний шрифт Montserrat. Також, для кращої командної взаємодії, розміри шрифтів були стандартизовані (див. Рис. 3.6).

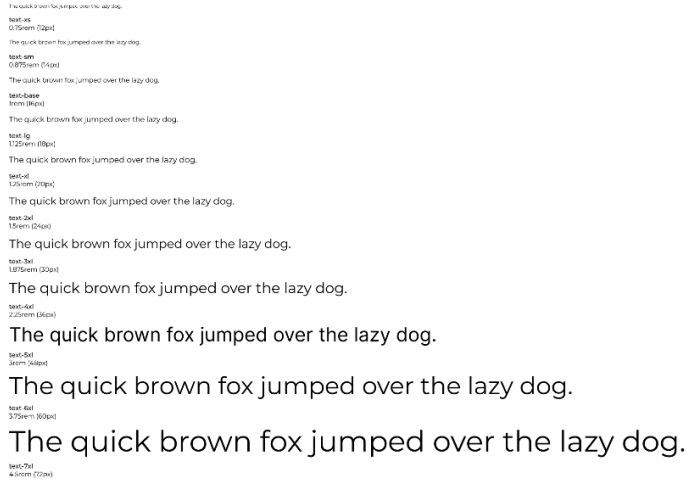


Рис. 3.6. Типографіка

Джерело: Розроблено автором

Стандартизація типографіки це про зручність як при розробці дизайн-макетів так і при верстці. Такий крок робить роботу зручною, розмір шрифту, інтервали між

символами та словами, у кожному блоці буде саме той шрифт, який попередньо задано стилями.

3.2.1.2. Створення кольорової палітри Tailwind

Базова палітра кольорів Tailwind включає в себе основні кольори, які будуть використано під час розробки. Tailwind - це CSS фреймворк, який містить набір готових класів для швидкої розробки інтерфейсів.

Використання відповідної кольорової гами (див. Рис. 3.7) дозволяє зробити дизайн більш привабливим та легким для сприйняття.

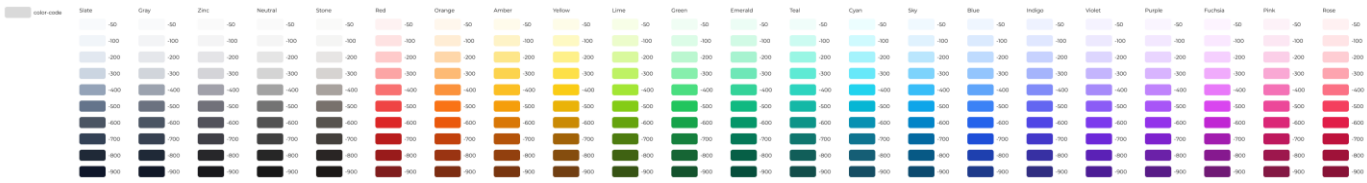


Рис. 3.7. Базова палітра Tailwind

Джерело: Розроблено автором

Також, створення палітри кольорів дозволяє забезпечити консистентність дизайну та уникнути проблем зі змішуванням кольорів, що може вплинути на єдиний стиль та зручність використання платформи.

3.2.1.3. Іконки

У процесі розробки будь-якого проєкту, важливою частиною є підбір іконок, оскільки вони допомагають у покращенні користувацького досвіду, а також допомагають зробити інтерфейс більш зрозумілим. У дизайні платформи ми використовували іконки з сету Google Material Design (див. Рис. 3.8).

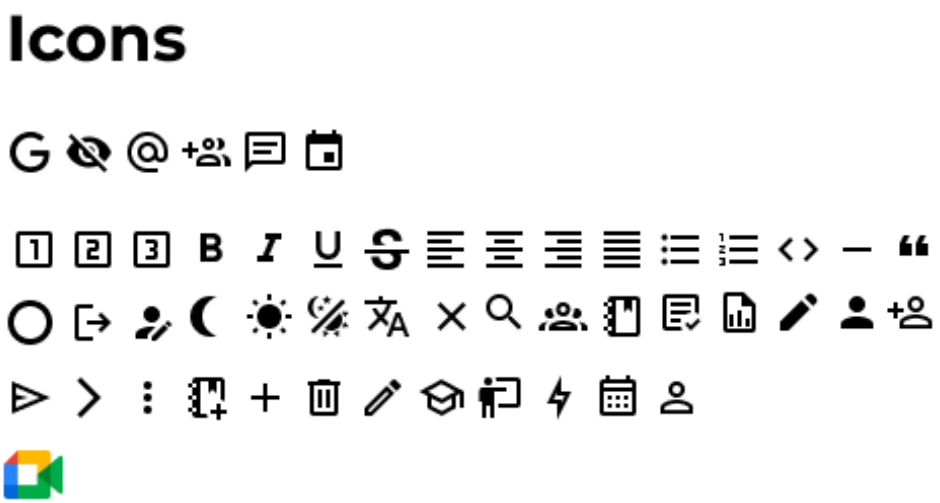


Рис. 3.8. Іконки

Джерело: Розроблено автором

Дизайн розроблений Google з метою створення єдиної системи дизайну. Він базується на принципах матеріального дизайну. Основною перевагою використання таких іконок є те, що такі іконки є добре знайомі для користувачів. Саме це полегшить розуміння та взаємодію з платформою. Також, використання саме цих іконок дозволить підвищити рівень довіри користувача до платформи.

3.2.2. Розробка дизайну

Розробка дизайн-макетів — це процес створення візуального зображення майбутньої платформи, створення зручного інтерфейсу.

Після визначення основного функціоналу за допомогою високодеталізованих скетчів та стандартизували стилі, тепер ми розробляємо дизайн основних сторінок. Головна мета створення дизайн-концепцій — це розробка детального дизайну платформи, який забезпечить зручну та якісну взаємодію між користувачем та платформою.

Розглянемо детальніше деякі сторінки платформи та їх функціонал.

Однією з основних та цікавих сторінок платформи, є сторінка усіх курсів (див. Рис. 3.9). Основною метою створення дизайн-макету сторінки курсу є забезпечення зручного та ефективного використання сторінки. Окрім того, сторінка дозволяє створювати викладачам нові курси та здійснювати навігацію за допомогою бокового меню.

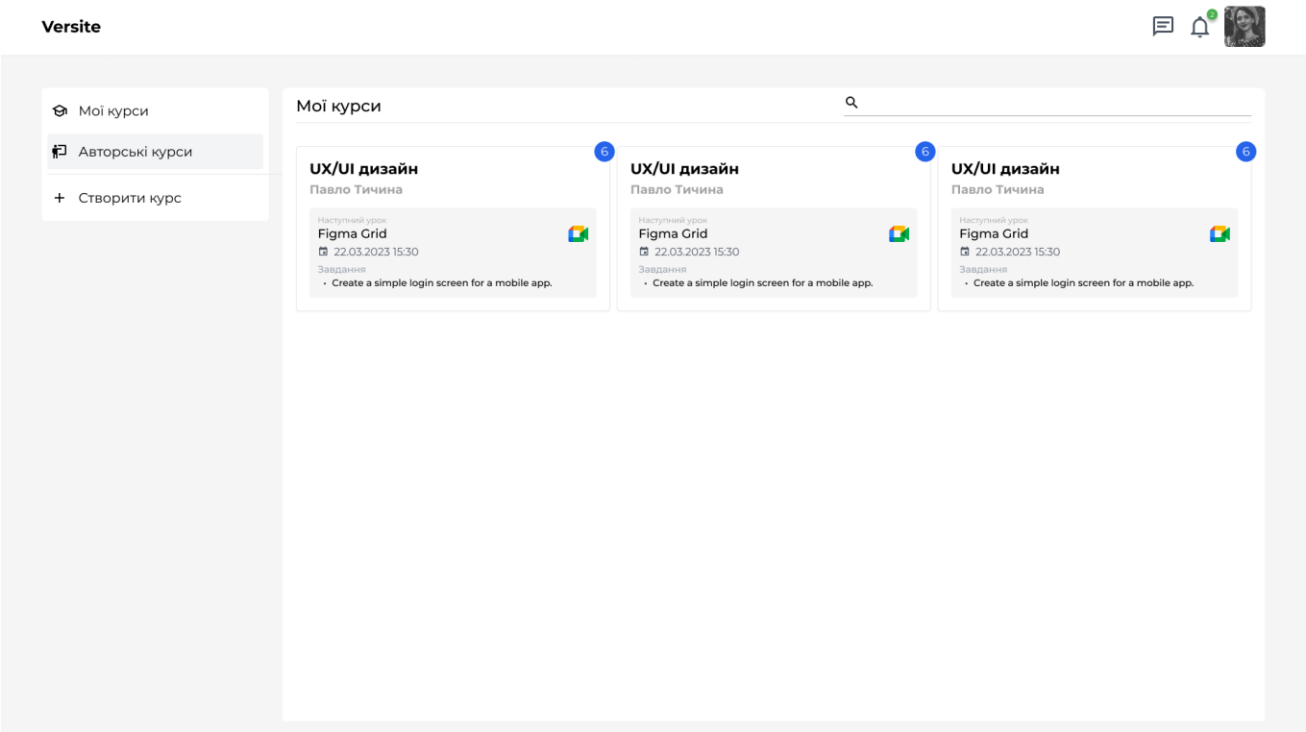


Рис. 3.9. Сторінка “Курси”

Джерело: Розроблено автором

На прикладі сторінки розглянемо основний функціонал.

Шапка сторінки

На всіх сторінках шапка сайту не буде змінюватися. Вона містить логотип, та навігацію. При кліку на дзвіночок користувач бачить свої сповіщення, а при натискані на фото профілю у нас з’являється меню (див. Рис. 3.10), у якому ми бачимо наші дані та можемо зробити налаштування.

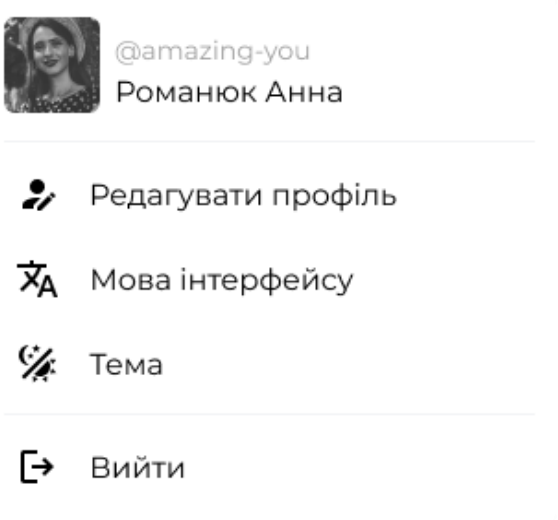


Рис. 3.10. Випадне меню

Джерело: Розроблено автором

Ліва бокова навігація

Одним із ключових елементів будь-якої платформи це зручна навігація, яка дозволяє користувачам швидко переходити між розділами. У нашому випадку ми маємо ліве бокове меню, за допомогою якого користувач може здійснювати навігацію між курсами які проходить та авторськими курсами (тими курсами, який користувач створив самостійно). Щоб зробити цей процес ще зручнішим та більш інтуїтивним необхідно показати різні стани елементів навігації, такі як неактивний, наведений та активний (див. Рис. 3.11).

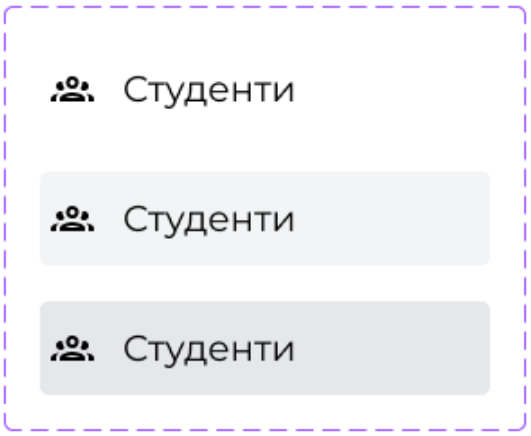


Рис. 3.11. Різні стани елементів меню

Джерело: Розроблено автором

Для цього був створений варіативний компонент, який дозволяє змінювати стиль елементів в залежності від їх стану. Це дозволяє забезпечити зручну та інтуїтивну навігацію для користувачів платформи.

Основна частина сторінки

У основній частині ми бачимо курси, які є у користувача. Якщо розглянути окремо карточку курсу (див. Рис. 3.12) ми маємо не багато графічних елементів.

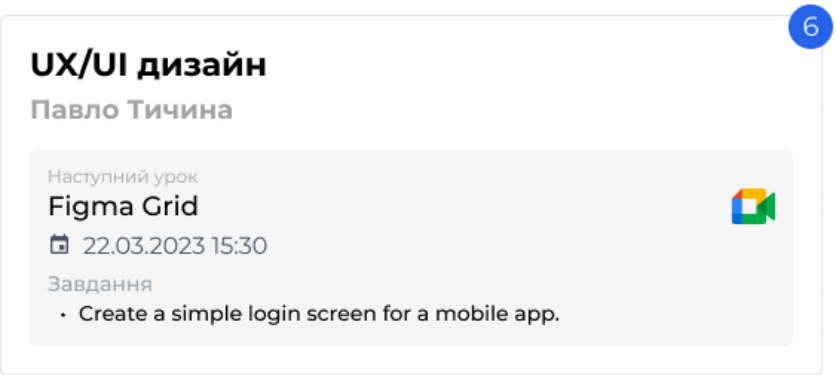


Рис. 3.12. Картка курсу

Джерело: Розроблено автором

Це заголовок курсу, його автор, опис найближчого або наступного уроку та домашнє завдання. У верхньому правому куті розміщується бейдж, який допомагає швидко передати інформацію користувачеві. Студент бачить кількість невиконаних завдань, а викладач — кількість неперевірених завдань.

Ще однією цікавою сторінкою є сторінка уроків (див. Рис. 3.13). Основною метою створення дизайну сторінки уроків курсу є забезпечення студентам доступу до списку уроків та їх завдань, допомогти їм зорієнтуватися в курсі та забезпечити зручний спосіб навігації по курсу.

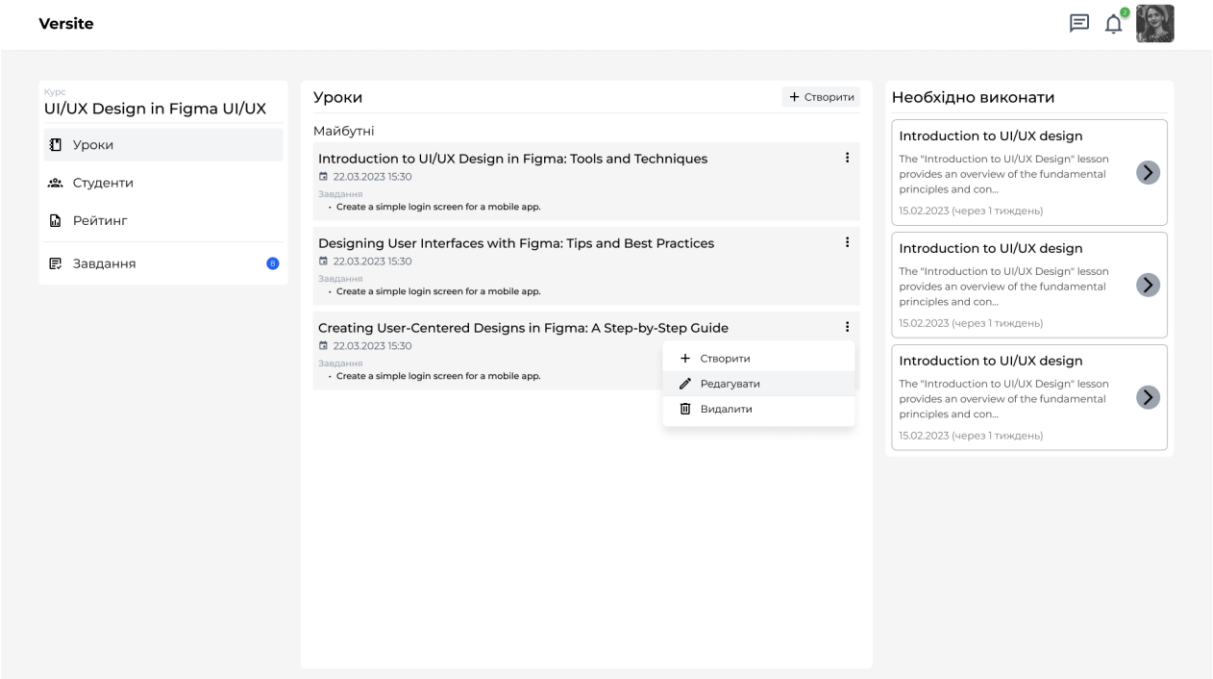


Рис. 3.13. Сторінка уроків курсу

Джерело: Розроблено автором

Серед основних функцій сторінки ми маємо:

Ліва бокова навігація

Навігація на сторінці уроку мають схожі елементи з навігацією на сторінці курсів (див. Рис. 3.9). Однак, на сторінці уроків ми додали новий пункт “завдання”, який є важливим як для студента так і для викладача. Викладач має змогу бачити скільки завдань йому необхідно перевірити, а студент може перевіряти скільки завдань йому необхідно зробити. Для того щоб показувати кількість завдань, був створений бейдж, який збільшує зручність користування та дозволяє швидко звернути на себе увагу.

Основна частина

На сторінці ми маємо повторювані компоненти уроків (див. Рис. 3.14), що схоже до компонентів на сторінці з курсами.

Рис. 3.14. Компонент уроку

Джерело: Розроблено автором

Розглянемо компонент уроку та наявний функціонал. Компонент містить у собі заголовок уроку, дату проведення, домашнє завдання. Заголовок та дата допомагаю ідентифікувати та відслідковувати уроки, а домашнє завдання дає користувачам можливість підготуватися до заняття.

У компоненті уроку, ми також маємо контекстне меню (див. Рис. 3.15), яке дозволяє викладачу створювати, редагувати а також видаляти уроки.

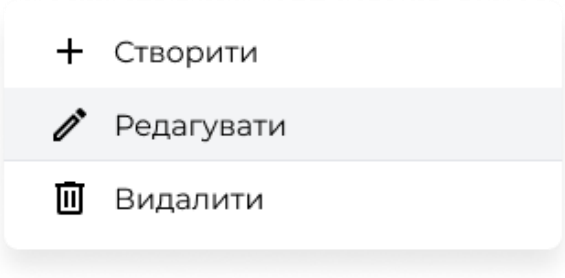


Рис. 3.15. Контекстне меню для налаштувань уроку

Джерело: Розроблено автором

Цей компонент є важливим елементом платформи, оскільки дозволяє користувачам легко знаходити та керувати своїми уроками.

Праве бокове меню

У правому меню (див. Рис. 3.16) користувач бачить найближчі завдання які необхідно виконати.

Рис. 3.16. Компонент домашньої роботи

Джерело: Розроблено автором

У компоненті у нас є заголовок завдання, короткий опис, терміни здачі та кнопка по якій можна перейти щоб почитати завдання детальніше.

Отже, для забезпечення зручної навігації та візуальної доступності на різних сторінках платформи “Versite”, було розроблено дизайн-концепції, які базують на принципах чіткості, лаконічності та доступності. Компоненти дизайну були створені для відображення різних станів елементів.

3.3. Тестування

3.3.1. Веб-доступність та тестування доступності

Фундаментально мережа інтернет створено для того, щоб у ній могла працювати кожна людина незалежно від обладнання, програмного забезпечення, мови та здібностей. Коли інтернет відповідає цій меті, він стає доступним для людей з обмеженими можливостями. Веб-доступність означає, що веб-сайти, веб-інструменти проектуються і розробляються таким чином, щоб ними могли користуватися люди з обмеженими можливостями. Якщо говорити більш конкретно, то люди з інвалідністю можуть сприймати, розуміти, орієнтуватися, взаємодіяти з інтернетом та веб-ресурсами.

Доступність має важливе значення не лише для організацій, а й для розробників, які хочуть створювати веб-сайти та веб-інструменти на високому рівні. Тоді вони не будуть перешкоджати людям з обмеженими можливостями користуватися їхніми продуктами.

Веб-доступність охоплює усі види інвалідності, серед них:

- Слухові. Люди, що мають проблеми із слухом, можуть мати проблеми з розумінням мовлення. Для забезпечення доступності веб-сайту для користувачів із вадами слуху використовують такі практики, як підказки до зображень, або відео із субтитрами та описами.
- Зорові. Люди із вадами зору мають проблеми із баченням, включаючи кольорову сліпоту, нездатність розділяти деталі або контрастність. Для забезпечення доступності використовують графіку високого контрасту, або використання розширеного тексту та маркерів для забезпечення зручного читання.
- Мовленнєві. Люди із вадами мовлення здебільшого мають проблеми із комунікацією та розумінням мовлення. Для забезпечення доступності використовують відео- та аудіо-матеріали з субтитрами та описом для допомоги в розумінні контенту, а також використовують інтерактивні елементи.
- Неврологічні. Люди із неврологічними вадами можуть мати проблеми із концентрацією, розумінням та обробкою інформації. Для забезпечення доступності використовують такі практики, як спрощення складних процесів та інформації, використання різних, допоміжних засобів та елементів для покращення концентрації.
- Фізичні. Користувачі із фізичними вадами можуть мати проблеми з опорно-руховою системою та використання рук. Для забезпечення доступності веб-сайту використовують клавішні скорочення, що допомагають використовувати веб-продукт без користування комп’ютерною мишею.
- Когнітивні. Користувачі із когнітивними вадами можуть мати проблеми із розумінням та запам’ятовуванням інформації. Для забезпечення доступності веб-сайту, використовують такі практики, як спрощення мовлення, використання інтерактивних елементів, які допомагають у запам’ятовуванні та розумінні. Також, використання візуальних засобів допомагають зосередитися на важливому контенті.

Веб –доступність корисна не лише людям з інвалідність, а й здоровим людям у різних, життєвих ситуаціях, наприклад:

- Люди з “ситуативними обмеженнями”, наприклад при яскравому сонячному освітленні
- Люди з “тимчасовою інвалідністю”, наприклад з зламаними кінцівками
- Людям чиї здібності зменшуються у зв’язку із старінням

Ініціатива з веб-доступності (Web Accessibility Initiative) розробляє технічні специфікації, керівництва, допоміжні ресурси, які описують рішення для забезпечення доступності. Вони вважаються міжнародними стандартами веб-доступності, наприклад, WCAG 2.0 або WCAG 2.1.

Тестування доступності (Accessibility testing) — це тестування веб-продукту з метою визначення рівня доступності для людей з різними вадами інвалідності. Для тестування доступності веб-сайтів використовують допоміжні інструменти. Це тестування доволі кропітке, має багато нюансів та стандартів. Для першої перевірки дуже добре підійде метод Easy Checks. Це первинний огляд веб доступності, при якому відбувається перевірка тексту, взаємодії та деяких загальний моментів.

3.3.1.1. Перевірка тексту

Заголовки

Перше що бачить користувач заходячи на сторінку — це назва сторінки. Змістовні назви сторінок особливо важливі, для орієнтації, вони допомагають людям швидко розуміти де вони знаходяться, і переміщатися між сторінками у браузері.

Таку перевірку було проведено без сторонніх ресурсів та розширень. Основне завдання перевірити чи коректно відображаються у рядку заголовка вкладці (див. Рис. 3.17).

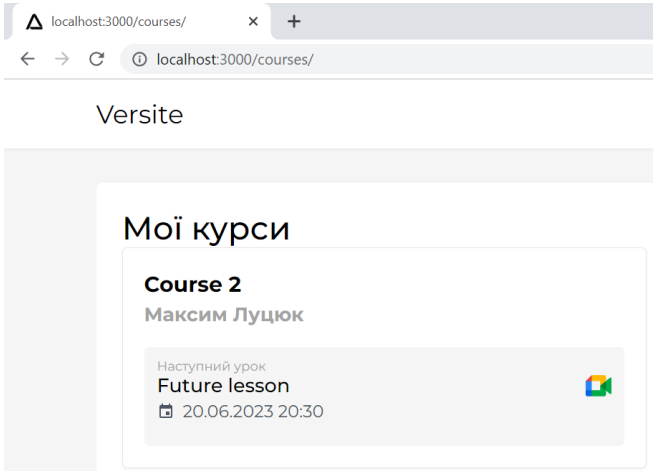


Рис. 3.17. Зображення коректності зображень
Джерело: Фото з екрану

На всіх сторінках нема проблем із відображенням заголовків і така назва коротко описує про що сторінка. Також сторінка має заголовок із назвою сторінки, тому користувачам не потрібно здогадуватися на якій сторінці вони знаходяться.

Рівні заголовків

На сторінках часто є розділи з різноманітною інформацією, які відокремлені візуальними заголовками, наприклад, текст заголовка більший і виділений жирним шрифтом. Рівні заголовків повинні мати правильну ієрархію, від найбільшого до найменшого, але це не є абсолютною вимогою.

При перевірці необхідно звернути увагу:

1. Майже на усіх сторінках повинен бути хоча ю один заголовок.
2. Весь текст, який має вигляд заголовка, оформлюється як заголовок.
3. В ідеалі сторінка починається з заголовка першого рівня, і надалі рівні не пропускаються. Але це не є обов’язковою умовою.

Щоб швидко перевірити ієрархію, використаємо розширення Wave Evaluation Tool. Це інструмент оцінки веб-доступності, який надає візуальний зворотній зв’язок про доступність веб-контенту, вставляючи іконки та індикатори на сторінці (див. Рис. 3.18).

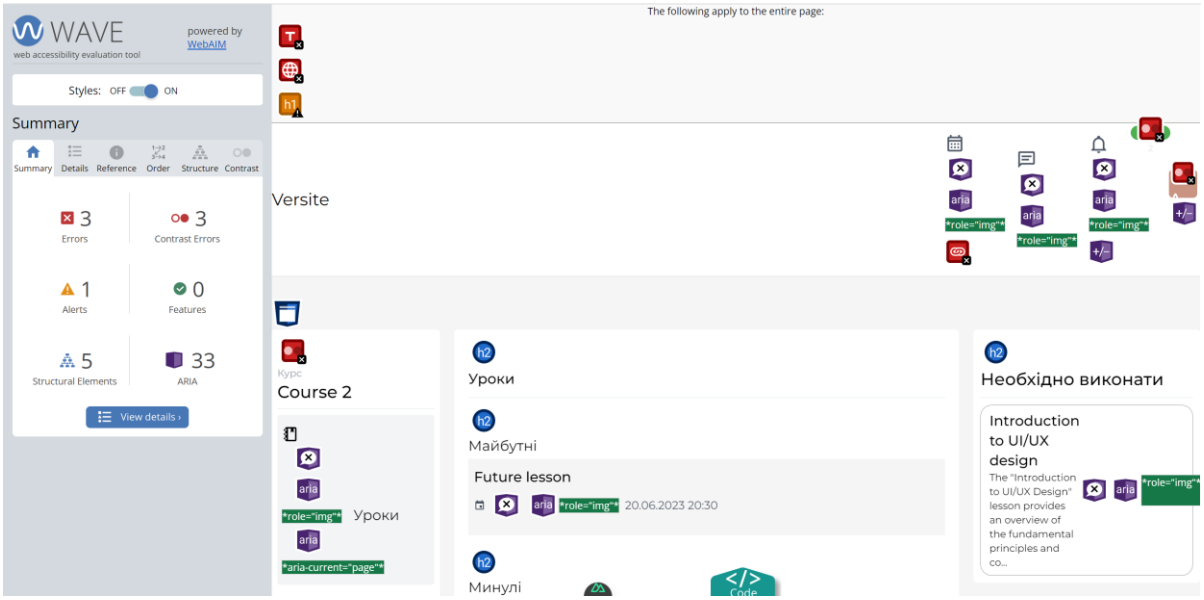


Рис. 3.18. Звіт Wave про веб-доступність
Джерело: Фото з екрану

При оцінювані ієрархії є попередження про відсутності заголовка першого рівня, як ми визначили це не є абсолютною вимогою. Також, у звіті є окрема вкладка “структура” (див. Рис. 3.19).

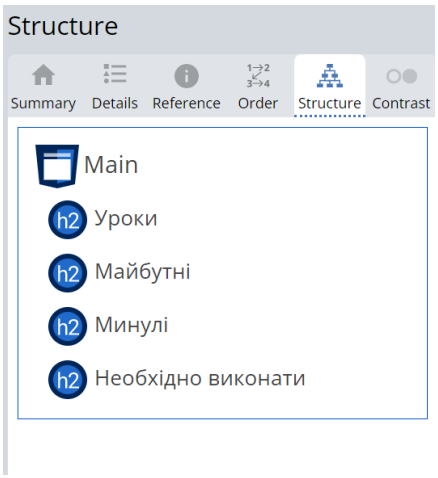


Рис. 3.19. Структура заголовків
Джерело: Фото з екрану

У структурі присутні заголовки другого рівня. Це значно полегшує сприйняття інформації на сторінці.

Коефіцієнт контрастності тексту

Деякі люди не можуть читати текст, якщо між текстом і фоном недостатній контраст. Високий контраст потрібний людям з порушенням зору, літнім людям які з часом втрачають контрастну чутливість, або здоровим людям за обставинами. У той час, наприклад, людям які мають дисклексію необхідна низька контрастність.

Для перевірки контрастності скористуємося розширенням для Google Chrome “Color Contrast Analyzer”

Color Contrast Analyzer — це розширення, яке допомагає визначити рівень контрастності згідно WCAG 2.0. Розширення оцінює сторінку так, як вона відображається у браузері, тому може обробляти текст поверх розширених атрибутів CSS та градієнтів.

Після проведення аналізу ми отримаємо звіт та мапу (див. Рис. 3.20), де межі між кольорами достатньо відрізняються, щоб задовільнити вимоги контрастності.

Рис. 3.20. Звіт контрастності тексту фрагменту сторінки уроків
Джерело: Фото екрану

Провівши аналіз наявних сторінок можна сказати що коефіцієнт контрастності доволі хороший. З наявних проблем є маленький текст сірого кольору на сторінках, та на кнопках з текстом білого кольору на кнопках (див. Рис. 3.21)

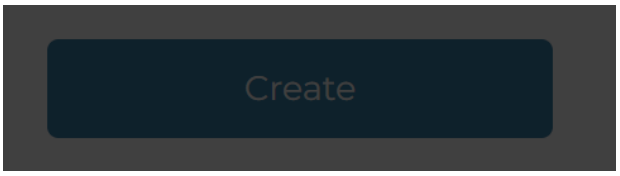


Рис. 3.21. Контрастність тексту на фоні
Джерело: Фото з екрану

На рисунку нема контрастності шрифту, але варто зауважити що кнопки не використовуються часто, тому це не є критично.

Контраст кольору при різновидах дальтонізму

Додаткове розширення для Google Chrome, яке допоможе перевірити сторінки на розрізнення кольорів, це colorblindly. Це розширення, яке допомагає розробникам створювати веб-сайти для людей з дальтонізмом, дозволяючи їм імітувати досвід таких користувачів на веб-сайтах. У розширенні існує 8 різних налаштувань (див. Рис. 3.22), на які можна випробувати сторінки.

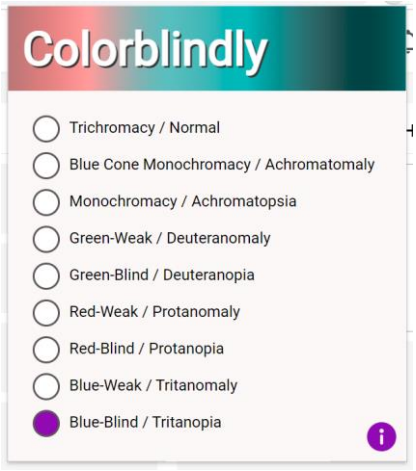


Рис. 3.22. Різновиди дальтонізму у розширенні
Джерело: Фото з екрану

Перевіривши усі доступні налаштування, можна зробити висновок, що люди з різними видами дальтонізму добре розрізняють кольори на сторінка. До того ж, при розробці проєкту основним способом донесення інформації є текстова інформація, кольорові елементи зустрічаються у іконках та бейджах (наприклад, показує кількості сповіщень), та фото.

Зміна розміру тексту

Деяким людям необхідно збільшувати веб-контент, щоб якісно прочитати, а іншим потрібно змінювати різні аспекти тексту, наприклад розмір шрифту чи інтервали. Одним із критеріїв доступності є збільшення веб-контенту до 200% без допоміжних технологій. Збільшення усього контенту вважається більш ефективним чим збільшення тексту.

Кожну сторінку перевіримо окремо, збільшивши масштаб до 200% стандартними інструментами Google Chrome (див. Рис. 3.23).

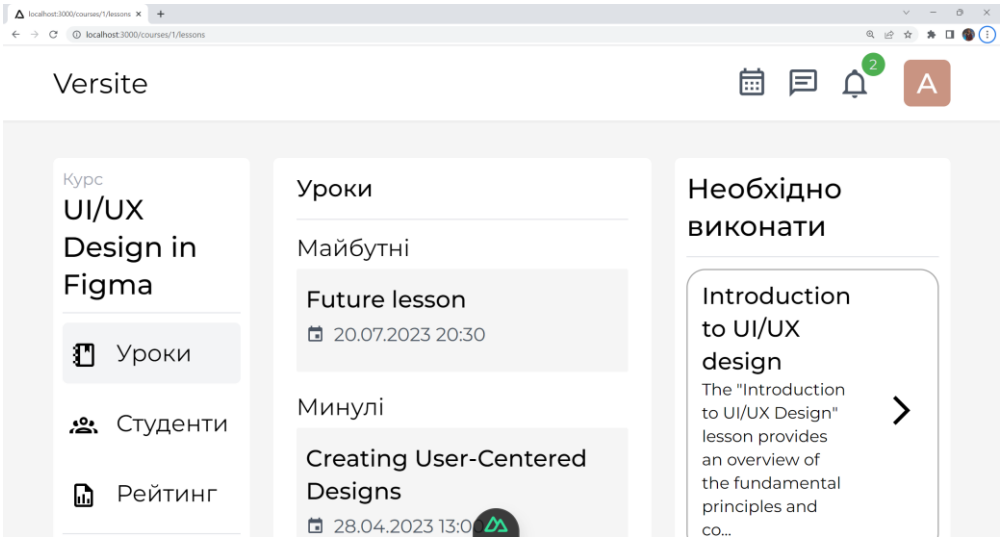


Рис. 3.23. Сторінка при масштабуванні 200%
Джерело: Фото з екрану

Перевіривши сторінки, було виявлено що весь контент відображається якісно при такому великому масштабуванні, і до того ж не з’являється горизонтальний скроллер. За такої умови не буде проблем з переглядом контенту.

Недоліки стандартів WCAG

1. Суперечливі вимоги

У деяких випадках стандарти можуть містити суперечливі вимоги або ж недостатню кількість вказівок щодо того, як саме дотримуватися стандартів. Наприклад, стандарти рекомендують забезпечувати контрастність між текстом та фоном для полегшення читання, але не уточнюють, як саме робити це у конкретних ситуаціях.

2. Застрілі приклади

Деякі приклади, які наведено у стандартах надто складні для розуміння або неадекватно відображати рольові ситуації. Це може призвести до різного сприйняття стандартів дизайнерами та розробниками веб-сайту. Також не завжди їх можна правильно застосовувати.

3. Рекомендації щодо шрифтів

Деякі стандарти є недостатньо точними або надто обмеженими. Наприклад, рекомендація використовувати шрифти з мінімальним розміром 16 пікселів для забезпечення читабельності може не підходити для всіх випадків, зокрема для мобільних пристроїв або для людей із заниженим зором.

Отже, WCAG 2.0 є найбільш поширеним стандартом веб-доступності, і у більшості випадків варто використовувати при розробці не лише платформ онлайн-освіти, а й інших веб-продуктів. Використання стандарту забезпечить доступність платформи для різних користувачів з різними вимогами щодо доступності. Також, необхідно враховувати недоліки стандарту та при розробці бути уважними щодо використання WCAG 2.0.

3.3.2. Тестування функціональності

Наразі платформу Versite не можна назвати великим продуктом, тому було доцільним проводити ручне тестування. Ручне тестування — це процес перевірки програмного продукту, який виконується без застосування автоматизованих засобів. Процес такого тестування часто буває складним та потребує багато часу, але це допомагає забезпечити кращу якість веб-сайту.

Одним із найпопулярніших типів тестування є функціональне тестування. Це тестування, яке спрямоване на перевірку функціональності, тобто на здатність відповідати заданим вимогам та виконувати очікувані функції. Зазвичай, при функціональному тестуванні використовують метод чорної або білої скрині.

Метод чорної скрині — це тип тестування, що виконується без знань внутрішньої роботи продукту, зосереджуючись лише на вхідних та вихідних даних.

Метод білої скрині — тип тестування, що проводиться з урахуванням структури продукту.

Для якісного тестування та виправлення вад, всі вони записували у звіт з відповідною структурою:

1. Заголовок (коротка назва проблеми або вади)
2. Опис проблеми (короткий опис вади)
3. Кроки до відтворення проблеми (детально відтворені кроки, опис того що було зроблено перед тим, як була знайдена вада)
4. Фактичний результат (опис, що сталося на сторінці)
5. Очікуваний результат (опис, що має статися після виконання описаних кроків)
6. При наявності фото екрану або відео, як був знайдений баг (додаткова інформація, для кращого відтворення вади)
7. Посилання на сторінку з вадою
8. Пріоритет (наскільки важлива проблема, та як швидко необхідно вирішити)
9. Дані про середовище (яка операційна система на пристрої та який браузер було використано)

При виявленні вади, такий звіт описувався у сервіс Trello. Цей сервіс вражає своїми функціями, тому після детального огляду, було прийнято рішення використовувати не лише для введення завдань, а й для звітності вад.

Розглянемо ситуацію на прикладі. На одній із сторінок, ми маємо секцію “необхідно виконати” (див. Рис. 3.24), саме у цій секції у нас не працює кнопка, при кліку на яку, ми маємо перейти на іншу сторінку і більше прочитати про завдання.

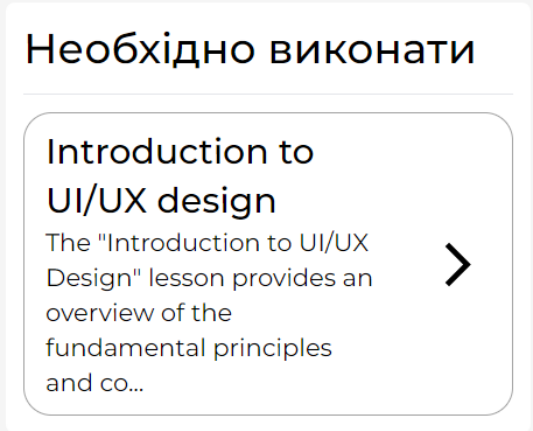


Рис. 3.24. Секція із завданнями

Джерело: Фото з екрану

Після виявлення проблеми, необхідно у Trello створити нову карточку та оформити у звіт, згідно структури (див. Рис. 3.25).

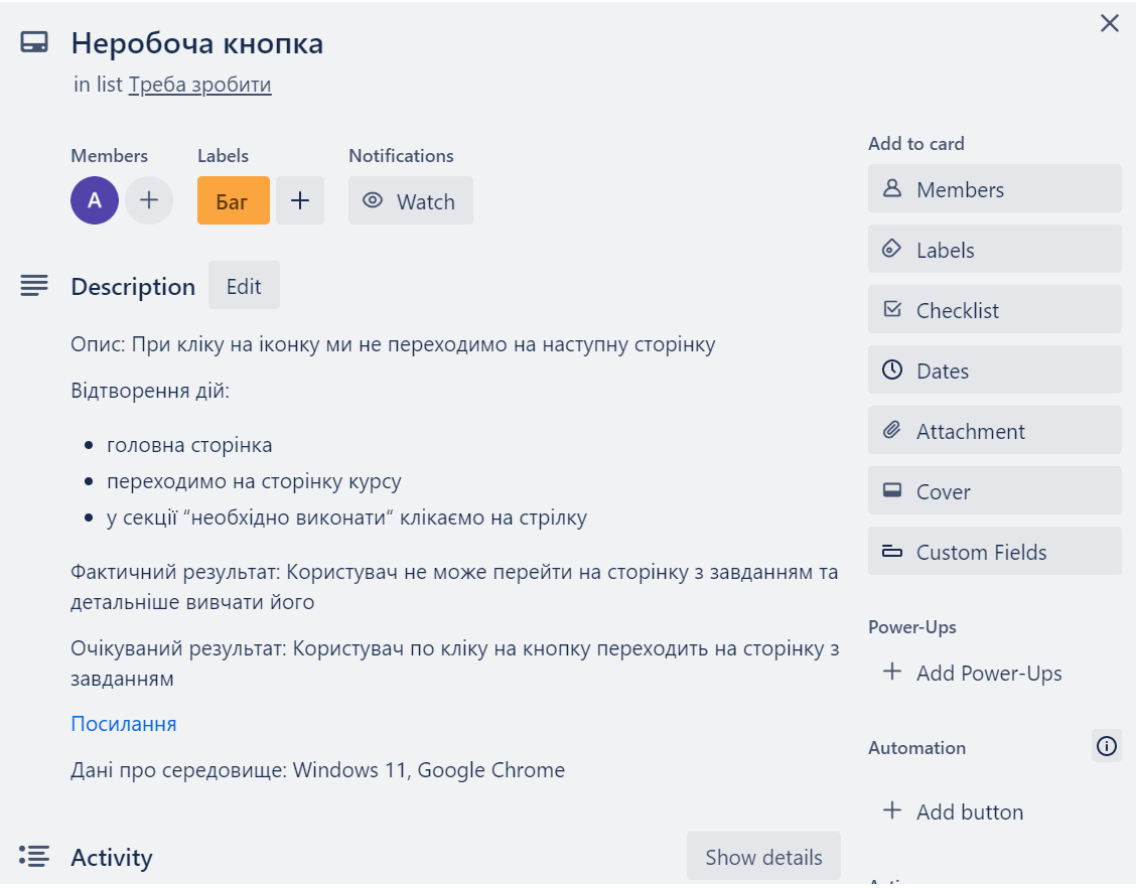


Рис. 3.25. Опис вади

Джерело: Фото з екрану

У системі Trello легко стежити за вирішенням таких вад. Коли вада буде у процесі виконання, виконавець надасть статус “робиться”, при необхідності можна додавати коментарі та спілкуватися щодо проблеми, або якщо у процесі будуть запитання. Після завершення роботи над вадою, її статус зміниться на “готово”.

ВИСНОВОК

У даному розділі було описано використані засоби розробки такі як Trello (планування та відслідковування виконання задач), Milanote (візуалізація ідей), Figma (створення скетчів та дизайну), Code with me (сервіс для спільної роботи над кодом).

Першим етапом практичної реалізації було створення високодеталізованих скетчів, за допомогою яких було оцінено розташування графічних елементів та іконок, наявність картинок та використання типографіки. Такий прототип виглядає як готова та функціональна сторінка, але монохромна. Наступний етап — це створення візуального зображення та зручного інтерфейсу.

Також було розглянуто проблематику веб-доступності для веб-застосунків, а саме відповідність розробленої платформи згідно міжнародних стандартів WCFG 2.0. Зокрема, було перевірено тексти (заголовки та ієрархію), перевірено коефіцієнт контрастності, контраст кольору при різних видах дальтонізму та зміну розміру тексту та контенту до масштабу 200%.

При тестуванні функціональності було проведено ручне тестування для якісного виявлення вад. Ручне тестування використовується для якісного виявлення вад у програмному забезпеченні шляхом ретельного перевіряння функціональності, взаємодії та відповідності результатів очікуванням. Цей процес включає в себе розуміння вимог, планування тестів та виконання тестових сценаріїв для ефективного виявлення помилок.

ВИСНОВОК ДО КВАЛІФІКАЦІЙНОГО ПРОЄКТУ

Під час виконання кваліфікаційного проєкту було сформовано та розроблено веб-платформу для онлайн-навчання. Задля виконання цього завдання необхідно було виконати наступні етапи:

1. Збір вимог
2. Аналіз аналогів та потреб користувачів
3. Створення скетчів
4. Розробка дизайну
5. Програмна розробка
6. Тестування

В результаті було створено готовий веб-продукт. По результатах виконання роботи можемо зробити наступні висновки:

1. Аналіз аналогів допомагає зрозуміти, які платформи та додатки існують на ринку онлайн освіти, які функції вони надають, як вони організовані та як взаємодіють з користувачами. Це дає можливість отримати уявлення про те, які рішення вже існують. Вивчення їхніх проблем, побажань та вподобань допомагає створити більш ефективні рішення.
2. Прототипи сторінок допомагають візуалізувати та конкретизувати ідеї щодо вигляду та функціональності платформи. Вони дозволяють отримати більш чітке уявлення про те, як будуть виглядати та працювати окремі елементи інтерфейсу. Також, допомагає краще зрозуміти логіку та послідовність дій користувача на платформі, що дозволяє виявити проблеми на ранньому етапі, та внести зміни до того, як буде розроблено детальний макет сторінки.
3. Code with me є зручним інструментом для спільної роботи над кодом, який має чималий список переваг, таких як повна інтеграція із середовищами розробки, вбудовані дзвінки, налаштування прав доступу. Особливо варто зазначити

можливість переадресації портів, завдяки чому можна надати доступ навіть до запущених програм.

4. Figma є потужним онлайн-інструментом, який надає багато інструментів для спільної розробки прототипів, скетчів, дизайну та різних мап та брейнштормінгу.
5. Онлайн-інструмент для організації своїх завдань Trello, дозволяє створювати дошки, списки та картки для контролю виконання завдань та спільної роботи. Інструмент можна налаштувати індивідуально під свій проєкт, та доступний з різних пристроїв.
6. Завдяки широким можливостям сервісу Milanote, його використання покращує процес генерування ідей, в особливості при спільній роботі над одним проектом. Організація створених елементів в дошках які підтримують можливість створення вкладених елементів дозволяє гнучко створювати власну структуру збереження цих даних.
7. Тестування веб-доступності дозволяє перевірити, чи можуть всі користувачі використовувати веб-сайт без перешкод і наскільки він задовольняє їхні потреби. Це важливий процес у розробці веб-сайтів для створення інклюзивного та доступного середовища для всіх користувачів.
8. Тестування функціональності дозволяє виявити помилки, дефекти та проблеми в роботі продукту, які можуть впливати на досвід користувача. Це можуть бути помилки завантаження сторінок, неправильно працюючі функції. Виправлення цих проблем допоможе покращити задоволення користувачів та зменшити кількість відхилень.

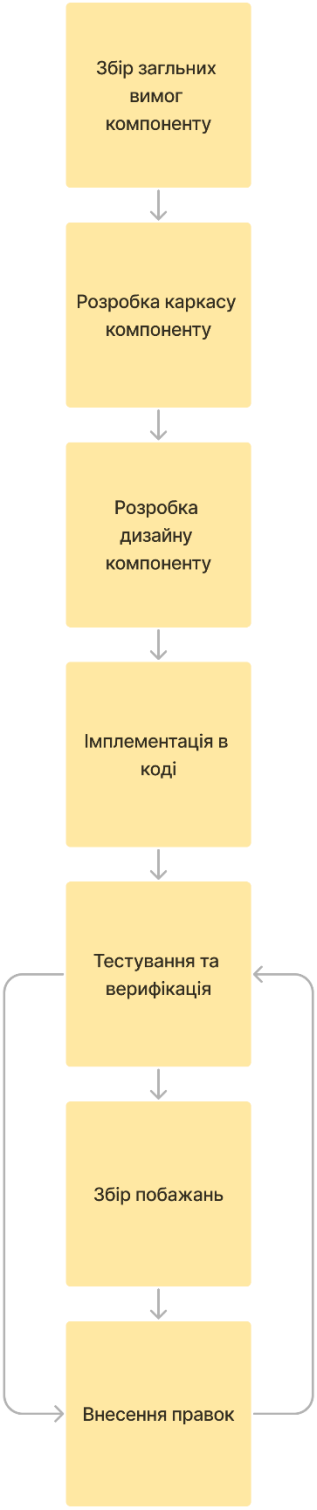
Під час виконання такого комплексного завдання як онлайн-платформа було отримано практичні вміння та грунтовні знання стосовно обраних засобів реалізації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Google Material Design. URL: https://m3.material.io/. (Дата звернення: 01.04.2023 р.)
2. Дональд Норман. Дизайн звичних речей. — Харків, 2019. 320 с. (Дата звернення: 01.04.2023 р.)
3. Dribbble. URL: https://dribbble.com/. (Дата звернення: 06.03.2023 р.)
4. Telegraf.design. User: stories, scenarios, journey, flow. У чому різниця?. URL: https://telegraf.design/user-stories-scenarios-journey-flow-u-chomu-riznytsya/. (Дата звернення: 05.03.2023 р.)
5. ProductPlan. User Flow. URL: https://www.productplan.com/glossary/user-flow/. (Дата звернення: 05.03.2023 р.)
6. Колекція векторних іконок у різних стилях. URL: https://icones.js.org/. (Дата звернення: 05.03.2023 р.)
7. Figma. Figjam. URL: https://www.figma.com/figjam/. (Дата звернення: 08.03.2023 р.)
8. Figma. Guide to components in figma. URL: https://help.figma.com/hc/en-us/articles/360038662654-Guide-to-components-in-Figma. (Дата звернення: 08.04.2023 р.)
9. Figma. Prepare to variants URL: https://help.figma.com/hc/en-us/articles/360055471353-Prepare-for-variants. (Дата звернення: 08.04.2023 р.)
10. Telegraf.design. Коли розробник керує дизайнерами. URL: https://telegraf.design/designer-keruye-rozrobnykamy/. (Дата звернення: 22.03.2023 р.)
11. Webtune. Види тестування сайтів. URL: https://webtune.com.ua/statti/web-rozrobka/vydy-testuvannya-sajtiv/. (Дата звернення: 03.04.2023 р.)

12. BrainLab. Як тестувати веб-сайт: основні етапи і поради. URL: https://brainlab.com.ua/uk/blog-uk/yak-testuvati-veb-sayt-osnovn-etapi-poradi#title_2. (Дата звернення: 05.04.2023 р.)
13. W3C Web Accessibility Initiate. Accessibility Principles. URL: https://www.w3.org/WAI/fundamentals/accessibility-principles/. (Дата звернення: 06.05.2023 р.)
14. W3C. Evaluation, Repair, and Transformation Tools for Web Content. URL: https://www.w3.org/WAI/ER/tools/. (Дата звернення 06.05.2023 р.)
15. W3C. Preliminary Evaluation, Web Accessibility Initiative (WAI). URL: https://www.w3.org/WAI/test-evaluate/preliminary/. (Дата звернення 07.05.2023)

Додаток А. Етапи впровадження компонента на сторінку



Вирішується які функції має виконувати компонент, та які пролеми має вирішувати.

Розробка основної структури без функціональності.

Створюється дизайн, включаючи усі елементи, функціональність, кольори та ін. графічні елементи.

На етапі імплемнтації відбувається перенесення дизайну на функціональну сторінку.

Проводиться тестування для перевірки правильної функціональності компонента.

Збір побажань щодо покращення компонента (при наявності).

Всі знайдені вади та побажання до покращення впроваджуються.

Додаток В. Код компоненту

```
<template>
 <div class="flex flex-col space-y-2 rounded bg-white p-2">
  <div v-if="props.courseTitle">
   <div class="text-sm text-gray-400">Курс</div>
   <div class="text-2xl font-medium" v-text="props.courseTitle" />
  </div>
  <div v-else>
   <SkeletonBone class="mt-1 h-3 w-16 rounded-full" />
   <SkeletonBone class="mt-2 h-6 w-10/12 rounded-full" />
  </div>

  <hr />

  <NuxtLink
   :to="{
    name: 'courses-id-lessons',
    params: { id: props.courseId }
   }"
   active-class="active"
   class="link"
  >
   <Icon name="mdi:notebook-outline" size="24" />
   <span>Уроки</span>
  </NuxtLink>

  <NuxtLink
   :to="{
    name: 'courses-id-students',
    params: { id: props.courseId }
   }"
   active-class="active"
   class="link"
  >
   <Icon name="mdi:account-group-outline" size="24" />
   <span>Студенти</span>
  </NuxtLink>

  <NuxtLink
   :to="{
    name: 'courses-id-rating',
    params: { id: props.courseId }
   }"
   active-class="active"
```

```
    class="link"
   >
    <Icon name="mdi:file-chart-outline" size="24" />
    <span>Рейтинг</span>
   </NuxtLink>

   <hr />

   <NuxtLink
    :to="{
     name: 'courses-id-assignments',
     params: { id: props.courseId }
    }"
    active-class="active"
    class="link"
   >
    <Icon name="mdi:text-box-check-outline" size="24" />
    <span>Завдання</span>
   </NuxtLink>
  </div>
</template>

<script lang="ts" setup>
const props = defineProps<{
 courseId: string;
 courseTitle?: string;
}>();
</script>

<style scoped>
.link {
 @apply cursor-pointer space-x-4 whitespace-nowrap rounded px-1.5 py-3 text-xl transition-colors
hover:bg-gray-200;
}

.active {
 @apply bg-gray-100;
}
</style>
```

Додаток С. User Flow